Formal Language Decomposition into Semantic Primes

Johannes Fähndrich[a], Sebastian Ahrndt[a], Sahin Albayrak[a]
[a] DAI-Laboratory, Faculty of Electrical Engineering and Computer Science, Technische Universität Berlin

KEYWORD

Semantic Decomposition
Semantic Description
AI Planning
Service Matching
#4

ABSTRACT

This paper describes an algorithm for semantic decomposition. For that we surveys languages used to enrich contextual information with semantic descriptions. Such descriptions can be e.g. applied to enable reasoning when collecting vast amounts of information. In particular, we focus on the elements of the languages that make up their semantic. To do so, we compare the expressiveness of the well-known languages OWL, PDDL and MOF with a theory from linguistic called the Natural Semantic Metalanguage. We then analyze how the semantic of the language is build up and describe how semantic decomposition based on the semantic primes can be used for a so called mental lexicon. This mental lexicon can be used to reason upon semantic service description in the research domain of service match making.

1 Introduction

Intelligent environments are made up of multiple pervasive or ubiquitous devices that provide a service to the user. One key feature of such environments is the ability to adapt to changes. The changes are implied by external or internal influences like the introduction or removal of devices or changing application goals [SALEHIE,M. 2009]. We expect that intelligent environments react to such changes and adapt themselves in a way that the services provided are still available for the users.

More than a decade ago *R.J. Sternberg* [STERNBERG,R. 1986] already emphasizes this by specifying that intelligence is the ability to adapt to changes in environments [to distinguish between the environments and the change occurring in it we refer to the current state of the environment as context]. This point of view implies that an environment becomes more intelligent if it can cope with more or bigger changes in the context. The problem at hand is the lack of adaptiveness of such environments.

One can distinguish two types of contextual information: The **defined context** and the **derived context** [HENRICKSEN, K. 2004]. In both cases, the devices making up the intelligent environment have to agree on a language to interpret the data collected by the sensing devices. In a defined context [e.g., a specific application] this language can be given to the environment by a domain model. Another approach is to use semantic languages to annotate contextual information and use reasoner that derive knowledge or facts from these annotations. This approach is called derived context. Here every device has its own local model of the environment, without having to agree on a global context model providing information about all devices.

Derived context is created by finding patterns in raw data from the sensing devices of an intelligent environment and annotating them with a given semantic language. A reasoner then reasons upon this annotated information to transform the information into a local domain model of the device. Sharing information between devices and with that between services emphasizes the requirement of identifying similar concepts in the annotation languages used, because different developers use different conceptualizations because of their different point of views. Furthermore, it underlines why no model of the whole context is needed.

Languages to describe semantics have been subject to research. *Bikakis et al*. [BIKAKIS, A. 2008] survey semantic-based approaches and applicable reasoning methods in the domain of ambient intelligence. Two of the research areas concerned with such languages are the Semantic Web community and the Agent community. Both have developed a quasi-standard language to describe semantics. In the semantic web community the *Web Ontology Language* [1] [MCGUINNESS, D. 2004] [OWL] is being widely used. The agent community uses the *Planning Domain Definition Language* [2] [FOX,M. 2003] [PDDL] to describe their planning problems. This paper will examine the fundamental concepts making up those two languages. Additionally the study includes the *Meta Object Facility* [FAVRE,J. 2004] [MOF] as a meta-language for artificial languages. We compare these approaches with a linguistic theory named the *Natural Semantic Metalanguage* [GODDARD,C. 1994] (NSM). The theory of the NSM states that every naturally developed language is based on 63 semantic concepts. We dive deeper into NSM in Section 2.

One approach to create adaptiveness in an intelligent environment is the use of service matcher which can – given a service request – find a fitting service and with that a device providing this service. By using such a service matcher the environment is enable to find alternative services if a service is no longer available or find new service to be used. By integrating a service matcher the environment thus becomes more failsafe, adaptable and extendable.

To find interchangeable service, we need to have equivalent classes of services. Thus the service matcher needs to find alternative service for a request. This means that the effect and purpose of a service needs to be understood. This leads to the need to understand the description of a service, which again this leads to a need to understand the components of a description, the concepts making up the description. Here we postulate that a reduction of concepts to simple concepts, a so called decomposition, helps to create equivalence classes among concepts and with that the identification of fitting services in an intelligent environment. The emerging dictionary is an Explanatory Combinatorial Dictionary (ECD) which can be part of a meaning-text linguistic model but not for natural rather then for artificial languages [MELCUK,I. 2006].

The approach of semantic decomposition postulates that the semantic primes form NSM do have a predefined meaning [FÄHNDRICH, J. 2013]. This meaning is then used to reason the meaning of more complex concepts, which are decomposed. This decomposition will then be used to classify concepts within their meaning and create equivalence classes.

The paper is structured in the following way: In Section 2 we introduce NSM and the basic concept of semantic primes in a nutshell. Section 3 takes such insights into three different meta languages and compares the languages in a more detailed way. Section 4 compares the categories of the semantic primes with primes used in the languages and analyses the usefulness of the primes in each category for formal language. Furthermore examples are given for the use of those primes in other research areas. Section 6 describes a decomposition algorithm which is the main contribution of this paper. An example of the decomposition is given and the remaining challenges of the algorithm are discussed. Afterwards, Section 7 wraps-up the paper with a discussion of the results.

2 NSM – Natural Semantic Metalanguage

The Natural Semantic Metalanguage (NSM) is a linguistic theory originated in the early 1970s [WIERZBICKA,A. 1996]. The theory states that each meaning of a concept created in a natural language can be represented using a set of atomic terms – so-called universal semantic primes. These primes have an indefinable word-meaning and can be identified in all natural

[1] For further information the interested reader is refereed to: http://lists.w3.org/Archives/Public/www-webont-wg/2001Dec/0169.html

[2] For further information the interested reader is also refereed to: http://www.cs.yale.edu/homes/dvm/

languages looked at [GODDARD,C. 2008]. In conjunction with associated grammatical properties NSM presents a decompositional system able to describe all concepts build in the appropriate language. Here, an expression is decomposed into less complex concepts, where the process ends if the expression is decomposed to the atomic level of semantic primes, which cannot be analyzed further. One can imagine that the decomposition builds a tree, where all leafs are semantic primes [WIERZBICKA,A. 2009].

Category	Semantic Prime
Substantive	I, YOU, SOMEONE, SOMETHING/THING, PEOPLE, BODY
Relational substantives	KIND, PART
Determiners	THIS, THE SAME, OTHER/ELSE
Quantifiers	ONE, TWO, MUCH/MANY, SOME, ALL
Evaluators	GOOD, BAD
Descriptors	BIG, SMALL
Mental predicates	THINK, KNOW, WANT, FEEL, SEE, HEAR
Speech	SAY, WORDS, TRUE
Actions, events, movement, contact	DO, HAPPEN, MOVE, TOUCH
Location, existence, possession, specification	BE (SOMEWHERE), THERE IS, HAVE, BE (SOMEONE/SOMETHING)
Life and death	LIVE, DIE
Time	WHEN/TIME, NOW, BEFORE, AFTER, A LONG TIME, A SHORT TIME, FOR SOME TIME, MOMENT
Space	WHERE/PLACE, HERE, ABOVE, BELOW, FAR, NEAR, SIDE, IN-SIDE
Logical concepts	NOT, MAYBE, CAN, BECAUSE, IF
Intensifier, augmenter	VERY, MORE
Similarity	LIKE, SAME

Table 1. These are the 63 semantic primes found in the English language [GODDARD,C. 2008] .

Consequently, for each natural language there exists a metalanguage that consists of the semantic primes in the specific syntax and their appropriated grammatical properties. About 63 semantic primes exist that are divided into 16 categories [WIERZBICKA,A. 2006]. Table 1 lists the primes for the English language.

3 Artificial Languages and Semantic Primes

As well as natural languages formally defined artificial languages are based on a meta-language (for example, the Meta Object Facility [SMITH,J. 2000] is the meta-language for UML[3]). This leads to the implication that the concepts defined in artificial languages are semantic primes, as they are used to describe more complex concepts by applying construction rules or in other words, grammatical rules/properties. Thus, on the conceptual level, artificial and natural languages can be compared using the primes that exist in the different languages.

Since the bag of semantic primes presented by NSM is empirically well-researched, this work tries to compare three artificial languages utilizing this bag of primes. For this comparison, we take the purpose and concepts of the languages into account and match the available primes with each other as foundation to discuss potentially missing primes in the languages.

Before comparing the primes we want to take a more detailed look at the set of semantic primes presented by NSM and how this set is build up.

First of all we can notice that some of the primes have their antonym included in the set of primes. One could argue that this is redundant because of the prime NOT, which could be used to negate the meaning of every prime and with that semantically describe its antonym. This argument can be contended with a simple example: DEATH and LIVE are at the first glance such antonyms. NOT DEATH could be LIVE. But to describe the difference between a stone that was never alive and an animal that died is not possible without the notion of LIVE.

[3] UML – The Unified Modelling Language – http://www.uml.org/

Thereby a stone is not dead because it was never alive. Or in other words, something that is dead has died sometime before, but something that is not alive is not implicitly dead since it has never died. This can also be traced to the definition of live: the ability to reproduce, the need to metabolize. The other way around dead is defined as 'no longer alive' [WLATER,E. 2008] thus the prime LIVE is needed to define death. This leads to the conclusion that the prime LIVE cannot be removed from the list of primes and be replaced with NOT DEAD and on the other hand DEATH can not be removed, since not everything is dead which is not alive. Further in the meaning of concepts we do not assume a binary logic. Thus not everything that is NOT BAD is automatically good. Examples for antonyms are discussed in **Table 7**. The same argumentation can be applied for the other antonyms found in the bag of primes of NSM.
In the following we will briefly introduce the considered artificial languages and list all equivalents found in comparison with NSM respectively.

The Web Ontology Language (OWL) is a semantic markup language to create structured knowledge representations and to enrich them with semantics. OWL is a W3C standard since 2004 and has been continuously developed since [GRAU, B. 2008]. OWL is an extension of the Resource Description Framework [LASSILA, O. 1999] and has become one of the most used languages to describe knowledge for AI. Since OWL is meant to describe structured knowledge the concepts used are abstract. Table 2 lists all equivalents found in comparison with NSM primes.

Sematic Prime	OWL
I	self.entry
SOMETHING/THING	owl:thing
KIND	rdfs:SubClassOf
PART	owl:topObjectProperty
THIS	owl:entityURI
SOME	owl:someValuesFrom
NOT	owl:complementOf
ALL	owl:allValuesFrom
ONE, TWO	owl:cardinality
THE SAME	owl:equivalentClass

Table 2. List of semantic primes and equivalent found in OWL.

The Planning Domain Definition Language (PDDL) is a first-order logic based language defined as an extended BNF [FOX,M. 2003]. It is commonly used to provide a standardized way to describe planning problems and the associated domains. The syntax allows the users to define among others actions, effects, quantifications and constraints and was intended to enable developers to describe 'the physics' of a domain. Given such a description an automated reasoning/planning process uses a goal also defined in PDDL to search for a plan that satisfies all constraints, requirements and preconditions. The concepts, which are equivalent to semantic primes, are listed in Table 3.

Semantic Prime	PDDL
SOMETHING/THING	:define
KIND	:types
THIS	::=
THERE IS	:exists
BEFORE	:precondition
AFTER	:effect
A LONG TIME	:maintain
A SHORT TIME	:wait-for
NOT	:not
CAN	:action
BECAUSE	:imply
IF	:when/constrained

Table 3. List of semantic primes and equivalent found in PDDL.

The Meta Object Facility (MOF) has been introduced by the Object Management Group and is formally defined, e.g., by Smith et al. [SMITH,J. 2000]. MOF has been developed to model the structure and behaviour of entities in software development and is an international standard.[4] It is intended to be used to model the syntax of domain-specific languages such as UML. Since MOF is quite abstract it introduces mostly structural semantic primes and is thus closer to OWL than to PDDL. Table 4 lists all equivalents found in comparison with NSM.

Semantic Prime	MOF
YOU	uri

[4] ISO/IEC 19502:2005 Information technology – Meta Object Facility (MOF)

SOMETHING/THING	object
BODY	instance
KIND	type, extend
PART	property
THE SAME	element.equals
ONE	multiplicityElement
BE (SOMEWHERE)	link
THERE IS	element
HAVE	classifier
BE SOMEONE/SOMETHING	extend
LIVE	create
DIE	delete
CAN	operation
IF	event

Table 4. List of semantic primes and equivalents found in MOF.

One might already have noticed that not all primes introduced in NSM can be matched with the concepts that exist in the considered artificial languages. The ones that are not matching with any artificial concept are listed in Table 5.

Category	Sematic Prime
Substantive	SOMEONE, PEOPLE
Determiners	OTHER/ELSE
Quantifiers	TWO, MUCH/MANY
Evaluators	GOOD, BAD
Descriptors	BIG, SMALL
Mental predicates	THINK, KNOW, WANT, FEEL, SEE, HEAR
Speech	SAY, WORDS, TRUE
Actions, events, movement, contact	DO, HAPPEN, MOVE, TOUCH
Space	WHERE/PLACE, HERE, ABOVE, BELOW, FAR, NEAR, SIDE, INSIDE
Intensifier, augmenter	VERY, MORE
Similarity	LIKE
Time	WHEN/TIME, NOW, FOR SOME TIME, MOMENT
Logical concepts	MAYBE

Table 5. List of semantic primes with no equivalent found in the other languages.

In the following we will discuss the implications of the comparison.

4 Comparison of Primes

The compared languages introduce additional concepts that are domain-specific and not part of the semantic primes, e.g. <owl:versionInfo>. Depending on the purpose of the language those additional concepts change. OWL, for example, was created to describe shared conceptualizations where versioning and backward compatibility is an important issue. Using NSM, those concepts could be described using the available semantic primes. This also applies for the other domain-specific languages. Hence, such additional concepts are merely domain-specific shortcuts. That means they do not influence the expressiveness of the language at all. In contrast to such shortcuts, there are multiple extensions to those languages for special cases like the Semantic Web Rule Language [SWRL] [HORROCKS, I. 2004], which introduces rules to OWL. Such extensions influence the expressiveness of the language and thus are exactly what we are looking for during this work. To identify which concepts are added to a language by an extension, we first analyze which concepts are available in the basic language before considering extensions.
We now discuss the 16 categories of semantic primes to analyze why the different concepts do or do not exist in the artificial languages and some of their extensions.

Substantives are the first category. In natural language these semantic primes are used to distinguish actors and to separate humans from other things. To describe meaning, humans often reduce a description of properties of things to the relation to humans, or more precisely to the relation to themselves [WIERZBICKA,A. 2009]. In formal ontologies, to compare this to a standard engineering task, one of the main objectives is to describe substantives (entities, facts, processes) using their properties, nature and relationship to each other, to create a theory for the explanation of nature [GRUBER,T. 2008]. It is not surprising that all three analyzed languages have a concept for ‘THING’ to start a

description from. Furthermore the 'THING' is seen at the top of each taxonomy (a partial order) introduced by an ontology, meaning that everything is a 'THING'. Additionally, 'PEOPLE' or humans are modeled as such and described explicitly.
For example, to describe the concept 'mouse' a semantic description that is context-independent most likely relies on a degree in Biology regarding the taxonomy of a mouse as illustrated in Figure 1 This objective description will contain a vast amount of information that is irrelevant to most contexts. We argue that a domain-specific description is easier to create and to reason upon. More details about a semantic decomposition are discussed in Section 6.
Describing a mouse so that a naïve[5] reader of the description can understand it, the description must refer to the potential context of the reader. In natural languages these readers are other humans, which implies that most descriptions in NSM are in the context of a human and their relation to things. This leads to the requirement to ground descriptions, which is also named 'Common Ground' [BRENNAN, S. 1998] and presumes mutual knowledge, mutual beliefs and mutual assumptions between the provider of the description and the recipient.

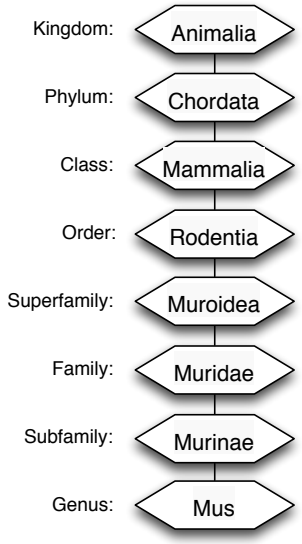

Figure 1 Taxonomy of a mouse

[5] By naïve we refer to a reader with only special knowledge of the domain he acts in.

We postulate that the grounding can be established by the decomposition into semantic primes from NSM (see section 6 for more details).
However, in the introduced formal languages, which are usually applied for communication between software agents, the descriptions have little common ground. Indeed the missing grounding is one of the major obstacles for interoperability in this area.
For artificial agents, the common ground has to be established by sharing contextual and semantic information. Sharing ontologies of beliefs, knowledge and contextual information can do this. Such information can be conceptualized in ontologies describing substantives and their relations in a domain-specific manner. This leads to different descriptions of the same entity in different domains, which leads to mismatches in the common ground [GAL,A. 2005] .
Thus a description should take into account the context of the agent using the description. This can be illustrated by the concept 'I': except in passive voice and citations, 'I' refers to the speaker themselves. This works fine as long as the identity of the speaker is known. For the identification most humans have names, virtual resources have IRIs and communication devices have IPs. Identifying in this case means to resolve the reference introduced by the 'I' with a concrete instance of the entity to identify.
The distinction of 'YOU' and 'I' is required if roles are described, e.g. in negotiation or contracting. None of the reviewed languages has a concept for 'PEOPLE'. On the one hand semantic descriptions in these languages are thought for artificial reasoners and there a concept of 'PEOPLE' is not needed to describe most concepts. On the other hand, in the area of HCI the distinction of artificial agents and human agents can be of some concern and with that the concept of 'PEOPLE' and 'SOMEONE' might be required.

The category of **relational substantives** is well represented in two of the languages, except PDDL as it does only use a type hierarchies based on 'KIND' to define domains. The relationship 'PART OF' is described in OWL as '<owl:topObjectProperty >". In MOF on the other hand the parthood relation is defined in

'Properties' which can be separated into necessary, properties (compositions) and optional ones (aggregation). Being of one 'KIND', on the other hand, defines the hierarchy of the entities, which is handled in the same way in OWL and MOF: through inheritance. In OWL this is denoted by '<rdfs:SubClassOf>', in MOF by the 'extend' or 'typed' relation. In consequence, this means that PDDL does not semantically aggregate all instances of one 'KIND'.
In semantic descriptions, 'KIND' is for example used to describe water: 'something of one kind' [GODDARD,C. 2010] .

Quantifiers are represented in all three languages in the form of universal quantifier ('ALL') and existents quantifier ('SOME') as well as cardinality of 'ONE'. In OWL for example we have the universal quantifier <owl:allValuesFrom> and existents quantifier <owl:someValuesFrom>.
The exception is the fuzzy representation of 'MUCH/MANY'. However, there is a need to enable fuzziness in semantic description languages as motivated by [STOILOS,G. 2005].

The category of **evaluators** is not represented in any of the analyzed language. PDDL from version 2.1 features the concept of numeric fluents to describe, e.g., cost of actions, which can be interpreted depending on the metric. Here an implicit metric is given to the reasoner, e.g. the plan minimization metric [FOX,M. 2003] . We argue that such a metric can be explicitly formalized in the description of a query or request itself and to define what is 'BAD' or 'GOOD' those concepts need to be part of the description language.
Another reason why evaluators should be part of the description language is that in the research area of Hyper-Heuristics [BURKE,E. 2013] an explicit description of properties which are good or bad given a specific goal might help.

Descriptors are represented in none of the three analyzed languages 'BIG' and 'SMALL' for example are fuzzy values and need to be defined in a description. For example, Wierzbicka describes mice as small in the following way: 'They are very small. A person can hold one easily in one hand' [WIERZBICKA,A. 2009] p. 9. Giving an example on what small means and a relation to something every reader of the explanation knows: the size of a hand. It follows that a reference is needed to defuzzify such descriptors. Furthermore, descriptors are concept dependent, which means that the meaning changes depending on which concept is described with them. Thus, for each concept described we need some kind of measure for the qualities like size. Explicitly specifying what the descriptors mean in the context of a given concept. How to handle fuzzy primes is analyzed in detail in section 5.

In relation to some reference or as constants the semantic primes 'A LONG TIME' and 'A SHORT TIME' can be used to describe relations in e.g. size, intensity, power or time. This makes the descriptors part of the Quality category of Aristotles.

We can imagine that for example a timeout can be explained by defining 'A LONG TIME' as the maximum timeout. We argue that if the semantic description is used by a reasoner to create a heuristic, a metric needs to be defined and with that semantic descriptors are needed that classify the values which is subject to the metric. Furthermore we argue that 'LONG' and 'SHORT' should be part of the descriptors since they are descriptors that could be used in addition to time with other concepts like spatial distances.

Mental predicates are represented in non of the three languages due to the fact that these predicates are based on the senses of human beings. We separate mental predicates in two groups: The first group is based on human cognition, containing: 'FEEL', 'SEE' and 'HEAR'. First of all, there are two of the senses missing: 'SMELL' and 'TASTE'. Additionally for the domain of intelligent environments and in other domains where sensors need to be described, such predicates can be used. This is not specific for humans, since every agent involved in an intelligent environment could have such cognitive functions (i.e. gas sensors, heat sensors). Additionally there could be sensors which extend the human cognitional like a barometer, altimeter or localization like GPS, which should then be incorporated in the

semantic primes as well. Even though these concepts are not used in the analyzed modelling languages, they are part of the semantic description of many fundamental concepts like 'HANDS' [WIERZBICKA,A. 2009]. The second group of mental predicates is the mental state of mind: 'THINK', 'KNOW', 'WANT'. These are philosophical terms and rarely used in artificial languages. Braubach et al. [BRAUBACH, L. 2005] e.g. describe a Believe, Desire, Intent [BDI] paradigm for agents. Here 'I believe' is considered a subset of 'I think' [WIERZBICKA,A. 2006]. In the BDI paradigm 'believe' can be mapped to the semantic prime 'KNOW', since it represents the knowledge of the agent, and 'desire' can be mapped to 'WANT', since it describes the internal goals of the agent. But in our analyzed languages, all of these concepts are missing.

Speech is – at first – the category, which holds one of the basic logical operators 'TRUE'. All three languages use an implicit representation of the concept 'TRUE' since they assume that a reasoner interprets an axiom as fundamentally true. PDDL for instance 'define a model to be an interpretation of a domain's language that makes all its axioms true' [MCDERMOTT,D. 2003] p. 5. Here again it can be argued to explicitly describe such truth values and with that add a semantic prime to the metalanguage. But we think that 'TRUE' should be keept in the category 'Logical concepts'.
Further we use 'WORDS' as basic building blocks for our description, and thus need a semantic prime for them. 'SAY' has been represented in a formal way as agent communication speech acts [COLOMBETTI,M. 2002] and could be directly part of the metalanguage.

Semantic primes in the category **Action** are often defined in a context-dependent manner, where the semantic is given by the reasoner that evaluates the axioms. In PDDL for example the blocks world defines 'MOVE[A,B,T]' [GUPTA,N. 1992]. NSM proposes to add such primes to the metalanguage, to be able to describe events, movement and actions with the primes: HAPPEN, MOVE and DO.

The semantic primes in the category **possession** (e.g., 'HAVE') can be seen as the specialization of composition and aggregation of the semantic prime 'PART'. The specification 'BE' denotes a location where something is located and at the same time to be of a certain type.

Life and death is a category, which is not subject to research in formal languages because computer systems rarely need a concept of death or living. Semantically there are many concepts that can not be described without the concept of 'LIVE' and 'DEATH'. In agent communication for example agents send 'alive messages' to other agents, where the interpretation is left to the programmer of the agent. But if one would want to describe the meaning of an alive message, one would need a notion of 'LIVE'. With a broad definition of live [RUIZ-MIRAZO, K. 2004] the termination of a software and with that for instance the non-functional requirement of computer systems for availability could be described as a goal of the system to stay alive. Formally death has been modeled in the SUMO ontology[6]. Since there is no use case of those two concepts known to the authors we postpone the in-depth analysis of the concepts of LIVE and DEATH.

Time has found its way into almost every formal language. Even an own logic – the temporal logic – which is a kind of modal logic has been created to model something like: 'I am hungry until I eat something'. In formal languages time has often been included into the language; e.g., PDDL from version 2.1.

The category **space** is subject to research and is formulated in contextual models like the CORBA-ONT [CHEN.H. 2003]. Nevertheless none of the surveyed languages presents primitive elements to describe special properties. The fact that an OWL ontology is required, shows that such semantic primes are necessary for the modeling of contexts. The same can be argued for the semantic primes 'BE [SOMEWHERE]' of the category location.

[6] see: http://virtual.cvut.cz/kifb/en/concepts/_killing.html last visit: 2014.08.11

Other fuzzy primes like 'NEAR' or 'FAR' are again hard to grasp in a formal language.

Logical concepts 'NOT' and 'IF' are part of most formal languages including PDDL and MOF, since they build the foundation of most binary logics. The hurdle primes are again the fuzzy ones: 'MAYBE' and 'CAN'. Those two primes model uncertainty. To describe the meaning of probability, those primes could be part of a meta language like they are in epistemic logic [VAN DER HOEK, W. 2004].

Intensifiers can be modeled as lexical functions [MELCUK,I. 1996]. Fuzzy decision systems have been subject to research [MENGEN,G. 1994] and thus to make their semantic explicit intensifier should be part of the metalanguage.
The semantic primes 'MORE' for example is modeled as lexical function 'MAGN' by Uson et al. [USÓN,R. 2005]. Which can then be used in the way like 'MAGN(KNOW)[Mice]' which is a more formal representation of the meaning of: to know more about mice.

Similarities are a huge research area and measures have been studied in depth [DUDA,R. 2001]. The developed methods like recommender systems try to find entities that are alike. Those methods try do define the prime 'SAME' and 'LIKE' for different domains by defining metrics.
The act of finding entities, which are similar, is called classification. The methods reach from Bayesian methods, Markov Models and Parzen Windows to C.45, Support Vector Machines and Artificial Neuronal Networks [DUDA,R. 2001].
The act of determining similarity as in being the 'SAME' is called clustering. Here one main research point is feature extraction. Feature extraction determines, which properties of an entity is well suited for its discrimination. The features are selected with the goal of reducing the 'Withing-Class-Scatter' and maximizing the 'Between-Class-Scatter'[DUDA,R. 2001] p. 120. Which means that the properties that are used to decide if two entities are the same have to be chosen in a way that similar entities have similar values in those properties.
By analyzing which property is characteristic for an entity we can decide how to describe the entity in a mental lexicon like Wierzbicka did [WIERZBICKA,A. 2009].

In this section we have seen that most of the 65 primes can be interpreted by an artificial reasoner. We have analyzed the need of formalizing the primes and come to the conclusion that some of them are already used in artificial languages in computer science and some of them are not as important yet. If a prime is needed in an artificial language depends heavily on the domain as we can imagine by looking at the vast amount of OWL-derivations and domain-specific languages in use.

5 Fuzzy Primes

As we have seen; some of the primes have no true or false evaluation but rather a degree of some quality. This leads to the problem of not being able to define a sharp meaning. Such concepts are called Fuzzy Primes and are based on an extension of set theory, the so-called fuzzy sets, probability or a theory of evidence.
Since those qualities are fuzzy we can utilize fuzzy descriptions like fuzzy sets including a membership function [KLIR, G. 1995] to describe the extend of the quality and help to defuzzify. An example membership function is presented in **Figure 2**.

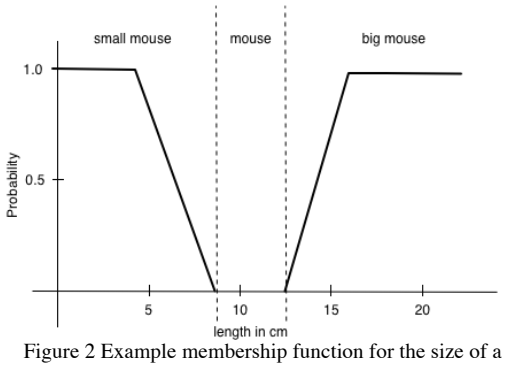

Figure 2 Example membership function for the size of a mouse.

Here the Primes BIG and SMALL are explained with the property size. This is simplified since the **Figure 2** uses a length measure, which of course must be grounded. In this case the common ground comprises the knowledge about a cm. This example shows how fuzzy sets can

be used to describe a quality that cannot be defined crisply.

The list of fuzzy primes can be found in Table 6.

Category	Semantic Prime
Quantifiers	MUTCH/MANY, SOME
Evaluators	GOOD, BAD
Descriptors	BIG, SMALL
Time	A LONG TIME, A SHORT TIME, FOR SOME TIME, MOMENT
Space	FAR, NEAR
Logical	MAYBE

Table 6 List of fuzzy primes as part of the set of primes of NSM.

For more formal details the interested reader is advised to read the work of Novák [NOVAK,V. 2001].

6 Semantic Decomposition

In this section we will propose a semi-automatic decomposition of meaning starting from a more complex concept and ending up with a description build upon semantic primes.
The idea behind this decomposition is to structure the creation of ontologies and to specify a granularity of detail to which a concepts needs to be decomposed to be fine-grained enough for an artificial reasoner.
The description is build up on less complex concepts which themselves are decomposed in less complex concepts until the level of semantic primes is reached. Here an inherent assumption from the theory of NSM is that those semantic primes are the most basic concepts which build up meaning of the words in an language (its concepts). This is how monolingual dictionary work, meta languages like MOF and OWL and how Wierzbicka [WIERZBICKA,A. 2009] describes a mental lexicon. Further Mel'čuck calls this kind of dictionary an Explanatory combinatorial dictionary (ECD) [MELCUK,I. 2006].
We will build a hierarchical structure made up from concepts also referred to as lexical units. Those concepts include a lexical representation, the textual representation of a lexeme and a decomposition. Furthermore a concept includes all its inflections (all concepts which can be created by applying grammatical forms to a concept like eating, ate, eaten), all lexical paradigms for this concept (all concepts rooting from the same word stem like to dine, dinner) and all sub-categorization frames (like the valence which is the amount of parameters like ask, ask X, ask X for Y).
It is important to notice that this is not meant for natural language processing. The languages decomposed here are formal languages like OWL. Thus the ECD only contains concepts of this language and thus has the goal to be computer readable.
The task of building a concept even if it is composed of multiple words and the search for lexical paradigms and inflections is leaved to linguists and will not be subject to our work.
Since the semantic primes are part of the metalanguage, an interpretation of those primes is not necessary as we have argued in a previous publication [FÄHNDRICH, J. 2013].
The prime 'KIND' for example is part of OWL and MOF. In both languages such a prime has the same interpretation [7] of tagging the inheritance of an object. Inheritance means that all properties, attributes and methods of the parent object are available to the child objects. This means that if we tell a reasoner that object B is a sub class of object A, the reasoner knows what to do, since the abstract prime 'KIND' is part of its language.

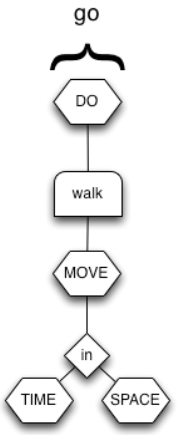

Figure 3 Decomposition of the verb 'to go'.

[7] Both languages give this prime the same meaning.

With decompositions we want to decompose every concept into those entities, which are known to the reasoner in a way that the meaning of the concept can be distilled from the meaning of the primes. One example of a decomposition could be the explanation of 'to go'. We take the English dictionary [WLATER,E. 2008] as reference for this decomposition: the meaning of 'to go' is defined as: 'to move from one place to another'. Figure 3 illustrated this decomposition as a schematic depiction. Here the hexagons are semantic primes from the list of NSM primes, round cornered tetragons are intermediate concepts and the rhombus are auxiliary words. The decomposition is done by taking the definition out of a dictionary, where such a complex concept is defined using less complex concepts. The intermediate concepts can provide different interpretations for different contexts.

Algorithm 1 A decompositonal algorithm

Name: Decompose **Input:** concept, PRIMES **Output:** decomposition

```
1: Syn ← findSynonyms(concept)
2: Ant ← findAntonyms(concept)
3: Hyper ← findHypernyms(concept)
4: Hypo ← findHyponyms(concept)
5: Definition ← lookUpDefinition(concept)
6: if ∃ s ∈ Syn : isSynonymOfPrim(s) then
7:    concept.decomposition + PrimeOfConcept(s)
8:    return concept.decomposition
9: end if
10: if ∃ s ∈ Syn : hasDecomposition(s) then
11:    concept.decomposition + DecompostionOfConcept(s)
12:    return concept.decomposition
13: end if
14: if ∃ a ∈ Ant : isAntonymOfPrim(a) then
15:    concept.decomposition + NOT + PrimeOfConcept(a)
16:    return concept.decomposition
17: end if
18: if ∃ a ∈ Ant : hasDecomposition(a) then
19:    concept.decomposition + NOT + DecompostionOfConcept(a)
20:    return concept.decomposition
21: end if
22: if ∀ hy ∈ Hyper : hasDecomposition(hy) then
23:    concept.decomposition + ALL + DecompostionOfConcept(hy)
24: end if
25: if ∀ ho ∈ Hypo: hasDecomposition(ho) then
26:    concept.decomposition + SOME + DecompostionOfConcept(ho)
27: end if
28: for all def in Definition do
29:    if def in PRIMES then
30:       concept.decomposition + def
31:    else
32:       concept.decomposition + Decompose(def)
33:    end if
34: end for
35: return concept.decomposition
```

We now have a look on how such decomposition can be created and how automatisms might help. We identified the steps for a decomposition as described in the recursive Algorithm 1. The algorithm takes as input the concept that is subject to the decomposition and the set of primes. The decomposition is held in a knowledge base, which can be queried and is denoted *'concept.decomposition'*. As a successful decomposition will always build a tree, the semantic primes are the leaves of such a tree and at the same point the termination criterion for the recursion.

The algorithm 1 reads as follows:

Lines 1 to 4 prepare the used external knowledge for the decomposition: First synonyms of the decomposed concept are stored in the set *Syn*. Synonyms are semantically equivalent concepts like stationary and fixed, motionless, halted, stopped, immobilized, immobile, unmoving, still, static, stock-still or at rest. Then Antonyms for the concepts are stored in the set *Ant*. Antonyms are words with an opposite meaning of the concept like stationary and travel, go, move or locomote. Table 7 shows examples of different types of antonyms.

Name	Example
Gradable	hot vs. cold
Complementary	off vs. on
Convers Relational	seller vs. buyer
Revers relational	adding vs. subtracting
Incompatible	Dog vs. banana

Table 7 Types of antonyms with examples

Further discussions on antonyms for artificial reasoning can be found for instance in [HEIM,I. 2008] and [NOVAK,V. 2001].
The next step is to find the hypernyms of the concept which are then stored in the set *Hyper*. Hypernyms are more general terms, so-called umbrella terms. 'Animal' for instance is a hypernym for 'dog' and 'human'.
Next the Hyponyms of the given concept are stored in the set *Hypo*. Hyponyms are concepts which have the Hypernym relation with the given concept. 'Dog' and 'human' are thus hyponyms of 'Animal'.
For those first four statements external sources like WordNet[8], Dictionaries and DBPedia[9] are used.
Line 5 stores the definition of a word. Here we use a monolingual dictionary with additional intelligence, where the same definition is not used twice since, if multiple definitions exist, and on can not be decomposed another one is used. This is done until one definition is successfully decomposed or all fail and the algorithm asks for a manual decomposition. The choice of a definition can be left to the user. The manual choice of the definition to use for the decomposition overcomes the problem of ambiguity and is not modeled in this algorithm due to complexity reasons but is considered as part of the indeterministic choice of the definition.
Such a dictionary can be an online dictionary like [WLATER,E. 2008] or DBPedia or some domain-specific dictionary like a medical encyclopedia.

Lines 6 to 9 validate whether one of the synonyms is a semantic prime or whether the concept itself is a prime (the set of synonyms always contains the concept itself). If so, the prime is added to the decomposition and the algorithm terminates.

Lines 10 to 13 checks if the concept of a synonym has already be decomposed. If so we add this decomposition instead of creating a new one. In this way we optimize the calculation effort by the means of memory and prevent to ask the user for a manual decomposition over and over again.

Lines 14 to 17 is similar to line 6 to 9 for synonyms but instead with antonyms.
If there is no synonym with a decomposition or which is a synonym of a prime, we can have a look at the antonyms. If we find an antonym which has a decomposition, or is itself a prime, we can add this prime or the decomposition of the antonym with a negation. In this way we

[8] http://wordnetweb.princeton.edu/perl/webwn
[9] http://dbpedia.org/

have decomposed the concept by explaining an antonym.
Since this description of the algorithm is a simplification, we neglect the fact that continuously decomposing all synonyms or antonyms could create better decomposition for the sake of simplicity.

Lines 17 to 21 are similar to line 10 to 13 but with antonyms.

Lines 22 to 27 complete the decomposition with additional facts. We can take the hypernyms and hyponyms as additional information sources.
If we have a generalization, we can count on everything being true for the more general concept, on being true for our concept as well.
Meaning that if a hypernym of the concept that is subject to decomposition has a decomposition we might add this decomposition since it applies to our concept as well. Thus, in the algorithm (lines 22 to 23) we add the decompositions of the hypernyms to our decomposition with the preamble 'ALL' since our 'KIND' relation is symmetric and with that, for example, all mice are rodents.
If we have a specialization of a concept, which has a decomposition, we might add this decomposition to ours, with the preamble 'SOME' since this adds additional information for the reasoner regarding the classification of a concept (lines 24-27).
In the example of our mouse we could add something like 'some rodents are mice' to the decomposition of rodents.
The same argumentation can be done with hyponyms.
If we have no decomposition yet, we need to start to create one by using a definition of the concept. Here we see a definition of a concept 'as an statement of the exact meaning of a [concept]' [WLATER,E. 2008] which we interpret in the following as: describing a complex concept with multiple less complex onse. By this interpretation at one point the decomposition becomes highly find-grained, consisting of very simple concepts, which can be decomposed using semantic primes.

Lines 28 to 34 describe the recursion of the algorithm. The use of the definition is done by the recursion in lines 28 to 34, where we again look at every concept of the definition and check if the concept is a prime or if we need to decompose it further.
The main point to notice here is that the order of words of the definition is not change. The grammar is thus untouched and does not yet influence the decomposition. Further this means the decomposition is highly influenced by the quality of the dictionary.

Manual Decomposition
Now we want to analyze the manual parts of the algorithm: Those concepts which cannot be decomposed automatically need a manual decomposition. This part is not modeled due to simplicity reasons of this paper.
There are multiple points, which can be handled semi-automatically.
First the selection of the concept type. Here the word type needs to be specified. This parameter influences the definition selection as well as the search for synonyms, antonyms hpernyms and hyponyms.
Second the selection of the definition. Here the sense of a word is selected. The user can select here which definition to use. In the example of 'well' this could be the meaning of 'in an satisfactory way' or 'a hole in the ground with water in it'.
Third the decomposition of a concept which has no definition or where the definition is circular. This can be the case, if the word is not described in the dictionary. In this case, the decomposition stops at this prime and might only continue with the decomposition if a manual definition or decomposition is introduced.
Fourthly the breadth of the recursion needs to be defined. This parameter specifies how many synonyms and antonyms are searched for in each decomposition step.
Fifthly the amount of additional information included. This parameter specifies how many examples of hypernyms and hyponyms are included in the decomposition.
Further the information sources need to be specified which includes the used dictionary and a mapping of the information sources to the information retrieval parts of the algorithm (line 1 to 5).

Termination conditions
The termination of the algorithm 1 has been chosen in a way that the decompositions are kept as short and precise as possible. For some tasks this might be advantageous and the order of steps in the algorithm need to be changed appropriately.
Algorithm 1 does not terminate if there are cyclic definitions in the used dictionary. Terminal behavior and a recognition when to ask for a manual decomposition will be discussed later.
The manner in which a decomposition is created depends - as always in ontology engineering - on the skills of the engineer, his domain and cultural background. Thus even with this strict framework of decomposition we might get different results for the same concepts. We postulate that those results differ in such a way that an artificial reasoner can create equivalence classes from those decompositions. This means that the decompositions might vary in different domains, by syntax and order of primes, which is irrelevant to the meaning the decomposition represents.

Example Decomposition
Now we will have a look at an example. In Figure 4 we illustrate an example of a decomposition of the concept **to use** in the sense of *putting something to a purpose*.
In this example we decompose the concept **use** as verb and selected a definition out of the [WLATER,E. 2008].Here we neglect the modal particles like 'to'.
The decomposition has been created by recursively applying the decomposition algorithm. The result is a relative short decomposition, because we did select the definitions which are used for the decomposition. Further we selected the words to use from the definition by hand.
The example from Figure 4 shows the feasibility of the decomposition. But there are still challenges, which need to be addressed.
Next we will have a look at the remaining challenges:
The **selection of the definition** to be used, for the decomposition influences the meaning which is decomposed. Here the different information sources must use the same sense of the word, for the decomposition. In this way ambiguities are neglected.
Circular definition in the dictionary lead to a non terminating behavior.
There are many **parameters**, which need specification reaching from the depth of the search for i.e. synonyms and the recursion depth for crawling through the dictionary.
Further **manual decompositions** are needed in case of a deadlock in the recursion of the algorithm. Here we need to find a way on how a human can be queried for his input.
Integration **of context dependent meaning** is not yet considered by our algorithm. If context dependent meaning needs to be described, or if the decomposition is context dependent, then context needs to be integrated into the recursion.
Finally, we want to stress the need for an extension of a reasoner using those concepts to be able to interpret the primes. Here a context dependent meaning can be defined for the primes so that the decomposition can be reasoned upon.

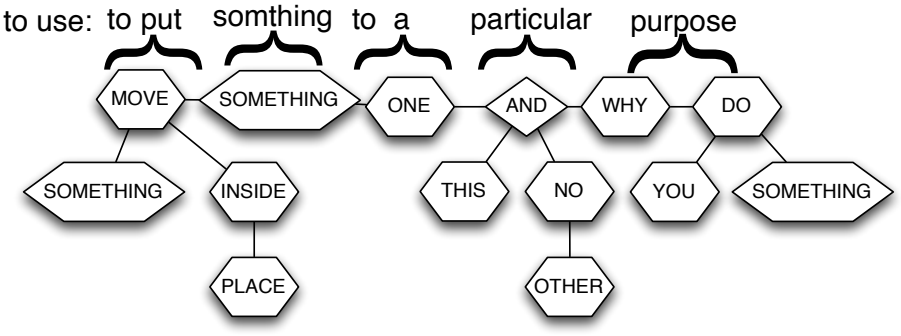

Figure 4 Example of decomposition using a definition out of the Cambridge Dictionary [WLATER,E. 2008]

7 Conclusion

We have analyzed three common semantic description languages and compared their meta languages with the set of semantic primes taken from NSM. We have found that already many of the semantic primes are part of the three formal description languages depending on their focus. The semantic primes that are not yet part of the description languages have been collected in **Table 5**. We have discussed the usefulness and potential interpretations of the different categories of the semantic primes, which broad forth that most categories are subject to research for artificial intelligence. Even so the languages for semantic descriptions do not include concepts for those categories like for instance probability. Then we introduced an algorithm for a semantic decomposition and illustrated in an example how the decomposition can be used to create structured ontologies.

The algorithm uses external data sources like domain specific dictionaries or DBPedia that are used to enrich a concept used in the description with additional semantic information.
This can be seen as a step towards a language in which an artificial intelligence might 'think' to process for instance the planning step of an agent planner.
Future work will include an in-depth analysis of those primes and their influence on the Description Logic of the language OWL. Here we want to examine which primes are useful for formal languages and define a set-theoretic semantic for each of them. Further, a practical evaluation of the decomposition algorithm will be analyzed to determine fitting parameters.

Acknowledgment
Research in this paper has been financed by the Schaufenster Elektormobilität http://www.schaufenster-elektromobilitaet.org in the project EMD (16SBB007A).

8 References

[BIKAKIS, A. 2008] Bikakis, A. et al., 2008. A Survey of Semantics-Based Approaches for Context Reasoning in Ambient Intelligence,

[BRAUBACH, L. 2005] Braubach, L., Pokahr, A. & Moldt, D., 2005. Goal representation for BDI agent systems. multi-agent systems, pp.44–65.

[BRENNAN, S. 1998] Brennan, S.E., 1998. The grounding problem in conversations with and through computers. Social and cognitive approaches to interpersonal …, pp.201–225. Available at: http://www.psychology.stonybrook.edu/sbrennan-/papers/brenfuss.pdf.

[BURKE, E. 2013] Burke, E.K. et al., 2013. Hyper-heuristics: a survey of the state of the art. Journal of the Operational Research Society, 64(12), pp.1695–1724.

[CHEN. H. 2003] Chen, H., Finin, T. & Joshi, A., 2003. An ontology for context-aware pervasive computing environments. The Knowledge Engineering Review, 18(3), pp.197–207.

[COLOMBETTI, M. 2002] Colombetti, M. & Verdicchio, M., 2002. An analysis of agent speech acts as institutional actions, New York, New York, USA: ACM.

[DUDA, R. 2001] Duda, R.O., Hart, P.E. & Stork, D.G., 2001. Pattern classification Second, Wiley-Interscience, New York.

[FAVRE, J. 2004] Favre, J.M., 2004. Towards a basic theory to model model driven engineering, 3rd Workshop in Software Model Engineering.

[FÄHNDRICH, J. 2013] Fähndrich, J., Ahrndt, S. & Albayrak, S., 2013. Self-Explaining Agents. Jurnal Teknologi (Science & Engineering), 63(3), pp.53–64.

[FOX, M. 2003] Fox, M. & Long, D., 2003. PDDL2.1: An Extension to PDDL for Expressing Temporal Planning Domains. 20, pp.61–124.

[GAL, A. 2005] Gal, A. et al., 2005. Automatic Ontology Matching Using Application Semantics. pp.1–12.

[GODDARD, C. 2008] Goddard, C., 2008. Cross-linguistic Semantics, John Benjamins Publishing.

[GODDARD, C. 2010] Goddard, C., 2010. Semantic molecules and semantic complexity:(with special reference to“ environmental” molecules). 8(1), pp.123–155.

[GODDARD, C. 1994] Goddard, C. & Wierzbicka, A., 1994. Semantic and Lexical Universals, John Benjamins Publishing.

[GRAU, B. 2008] Grau, B.C. et al., 2008. OWL 2: The next step for OWL. Web Semantics: Science, Services and Agents on the World Wide Web, 6(4), pp.309–322.

[GRUBER, T. 2008] Gruber, T., 2008. Encyclopedia of database systems L. Liu & T. Ozsu, eds. Encyclopedia of Database Systems.

[GUPTA, N. 1992] Gupta, N. & Nau, D.S., 1992. On the Complexity of Blocks-World Planning. 56(2--3), pp.223–254.

[HEIM, I. 2008] Heim, I., 2008. Decomposing antonyms. In Proceedings of Sinn und Bedeutung.

[HENRICKSEN, K. 2004] Henricksen, K. & Indulska, J., 2004. Modelling and using imperfect context information. Audio, Transactions of the IRE Professional Group on, pp.33–37.

[HORROCKS, I. 2004] Horrocks, I. et al., 2004. SWRL: A Semantic Web Rule Language Combining OWL and RuleML, W3C Member Submission.

[KLIR, G. 1995] Klir, G.J. & Yuan, B., 1995. Fuzzy Sets and Fuzzy Logic: Theory and Applications, Upper Saddle River, NJ, USA: Prentice-Hall, Inc.

[LASSILA, O. 1999] Lassila, O. & Swick, R.R., 1999. Resource description framework (RDF) model and syntax specification.

[MCDERMOTT, D. 2003] McDermott, D.V., 2003. The Formal Semantics of Processes in PDDL.

[MCGUINNESS, D. 2004] McGuinness, D.L., Van Harmelen, F.others, 2004. OWL web ontology language overview. W3C recommendation, 10(2004-03), p.10.

[MELCUK, I. 1996] Mel’čuk, I. & Wanner, L., 1996. Lexical functions and lexical inheritance for

emotion lexemes in German. Lexical functions in lexicography and natural language processing. Amsterdam/Philadelphia. John Benjamin, pp.209–278.

[MELCUK, I. 2006] Mel'čuk, I. (2006). Explanatory combinatorial dictionary. *Open problems in linguistics and lexicography*, pp. 225-355.

[MENGEN, G. 1994] Mengen, G.D.T.U., 1994. Fuzzy Decision Support-Systeme.

[NOVAK, V. 2001] Novák, V., 2001. Antonyms and linguistic quantiers in fuzzy logic. Journal of Web Seamantics, 124(3), pp.335–351. Available at: http://www.sciencedirect.com/science/article/pii/S016501140100104X.

[RUIZ-MIRAZO, K. 2004] Ruiz-Mirazo, K., Peretó, J. & Moreno, A., 2004. A Universal Definition of Life: Autonomy and Open-Ended Evolution. Origins of life and evolution of the biosphere, 34(3), pp.323–346.

[SALEHIE, M. 2009] Salehie, M. & Tahvildari, L., 2009. Self-adaptive software: Landscape and research challenges. 4(2), pp.1–42.

[SMITH, J. 2000] Smith, J. et al., 2000. Category theoretic approaches of representing precise UML semantics. Available at: http://www.ccs.neu.edu/home/kenb/pub/2000/03/public.pdf.

[STERNBERG, R. 1986] Sternberg, R.J. & Clarke, A.M., 1986. Beyond IQ: A Triarchic Theory of Human Intelligence. British Journal of Educational Studies, 34(2), p.205.

[STOILOS, G. 2005] Stoilos, G. et al., 2005. Fuzzy OWL: Uncertainty and the Semantic Web. OWLED.

[USÓN, R. 2005] Usón, R.M. & Faber, P., 2005. Decomposing semantic decomposition: Towards a semantic metalanguage in RRG. pp.1–28.

[VAN DER HOEK, W. 2004] van der Hoek, W., 2004. Epistemic logic for AI and computer science, Cambridge University Press.

[WLATER, E. 2008] Walter, E., 2008. Cambridge advanced learner's dictionary, Ernst Klett Sprachen. Available at: http://dictionary.cambridge.org/dictionary/british/.

[WIERZBICKA, A. 2006] Wierzbicka, A., 2006. English : Meaning and Culture. pp.1–363.

[WIERZBICKA, A. 2009] Wierzbicka, A., 2009. Mental Lexicon. In Berlin: Mouton de Gruyter.

[WIERZBICKA, A. 1996] Wierzbicka, A., 1996. Semantics: Primes and Universals, Oxford University Press, USA.