

Priming-based Disambiguation in Ontological Semantics

Max Petrenko

Princess Ekaterina R. Dashkova Moscow Humanities Institute
and RiverGlass, Inc.

Moscow, Russia, and Champaign, IL, USA

maxpetrenko@alumni.purdue.edu

Abstract – *The paper explores the possibilities of emulating the mechanism of disambiguation by priming within the framework of Ontological Semantics. Based on rich static knowledge resources and powerful parsing capabilities, a general principle of priming-based disambiguation is offered which allows the Ontological Semantic parser to compute semantic primes through ontological properties.*

Keywords: Ontology, computational semantics, priming, ambiguity.

Based on pertinent examples, the paper illustrates how the mechanism of priming, used by human agents for disambiguation, can be described and emulated using the Ontological Semantic platform. First, a brief discussion of priming and its disambiguating role will be provided, and ways of resolving standard cases of ambiguity by priming will be considered. Second, it will be shown how priming-based disambiguation can be translated into the language of Ontological Semantic technology, and general principles of modeling priming within the Ontological Semantics framework will be outlined.

1 Priming-based disambiguation

Priming is described as the co-occurrence of linguistic items (mostly lexical entries), whose relatedness is based on semantic proximity (i.e. “car” – “driver”), syntactic (“spider” – “web”) or collocational (“run” – “Forrest” – “run”) frequency [7]. Based on the psychological research, priming is known to develop at an early age, operates at the subconscious level, is susceptible to repetition-based memorization, and is widely accessible by systems of recognition, perception, and conceptualization [8].

At the level of semantic competence, whose most notable feature is the ability to resolve ambiguity (first defined in [1] as the core component of semantic competence), priming plays a role of a

“fine-tuning” mechanism, which assigns saliency (and thus “sensitizes” the speaker) to items required for disambiguation. The salient element then determines further selection of semantic, categorical and syntactic properties of the text. To illustrate, the ambiguous status of the noun “bank” in the sentence “He waited by the bank” can be eliminated by priming each of the two possible readings in the prior context:

1 He waited by the bank.

(a) He enjoyed rivers, so he waited by the bank.

(b) He needed to deposit the money, so he waited by the bank.

By “attuning” the speaker to the sense “landscape entity” in (a) by evoking the related sense of “body of water”, and to the sense of “financial organization” in (b) by evoking the related sense of “currency”, priming effectively eliminates the competing sense of “bank” in each reading.

Note that the relatedness of the “prime” (i.e. the evoking sense) to the “target” (i.e. the evoked sense) can vary from direct (i.e. duplication of senses) to remote (i.e. mentioning of a related sense or common property). Consider options for 1(a):

(i) He enjoyed [river banks / natural sites /

*physical objects], so he waited by the bank

(ii) He enjoyed [rivers / fresh water/ ducks / fishing],

so he waited by the bank

Assuming that the candidates are ranked on the remoteness scale, where the sense “river bank” is evoked either by duplication or via hypernyms in (i) or via various types of spatial relationship in (ii), the ineligible sense “physical objects” in (i) indicates the greatest semantic distance and reinstates ambiguity. A legitimate question, which deserves a separate inquiry, is whether the disambiguating potential of the prime is commensurate with its semantic proximity to the target.

2 Priming-based disambiguation and ontological semantics

For a computational system that aims at emulating semantic competence at the human-like level, the description above translates into the need to model this priming-based disambiguation faculty in its full complexity. This section will briefly outline the Ontological Semantic technology (OST) and offer general principles for emulating priming.

2.1 OST in a nutshell

OST focuses on a full, comprehensive and ambiguity-free representation of natural language. The static knowledge resources reproduce extra-linguistic (i.e. world view) and linguistic (syntactic, morphological and lexical) knowledge of human agents in a comprehensible, versatile and robust fashion.

The ontology is a parsimonious language-independent multi-level hierarchy branching into concepts OBJECT, EVENT interconnected through hundreds of PROPERTYIES. The mechanism of slot filling conforms to the “slot/facet/filler” format, where facets DEFAULT, SEM, RELAXABLE-TO prioritize fillers according to their prototypicality (see [2] on the function of facets and [7] on their role in computing inferences). Inherited and transitive slot fillers are also computed and blocked through the facet NOT where needed. A typical ontological template for a concept within OST has the form of (concept(property(facet(filler)))) for OBJECT’s EVENT’s and the form of (property(domain(facet(filler)))(range(facet(filler)))) for PROPERTYIES (see [6] for a detailed discussion and formal description of the ontology). The concept BLOOD-VESSEL serves as an illustration:

```
(blood-vessel
  (definition (value("an elastic tubular
channel, e.g. artery, vein, etc., through which blood
circulates"))))
  (is-a
    (hier(circulatory-system)))
  (path-of(sem(flow
    (theme(default(blood))))))
  (contains
    (default(blood)))
  (connected-to
    (sem(blood-vessel heart)))
  (inside-of
    (sem(body-part))))
```

Besides the IS-A relation indicating the parental concept CIRCULATORY-SYSTEM, the definition of the concept BLOOD-VESSEL is enriched by the property PATH-OF (filled by the concept FLOW with its own property THEME filled by BLOOD via the DEFAULT facet), the property CONTAINS (filled by BLOOD via the DEFAULT facet), the property CONNECTED-TO (filled by concepts BLOOD-VESSEL and HEART), and the property INSIDE-OF (filled by the concept BODY-PART). The concept BLOOD-VESSEL is also available for computation through PROPERTIES that have it or any of its ancestors in their DOMAIN (i.e. a set of concepts they define) or RANGE (i.e. a set of their fillers). For instance, properties PART-OF-OBJECT (and its inverse HAS-OBJECT-AS-PART), HAS-LOCALE (and its inverse LOCALE-OF) and END-LOCATION (and its inverse LOCATION-OF) will have the concept PHYSICAL-OBJECT in their DOMAIN and RANGE. This will make BLOOD-VESSEL accessible by the parser through these PROPERTIES because PHYSICAL-OBJECT is the ancestor of BLOOD-VESSEL.

The lexicon is a language-specific repository of lexical entries. The semantic structure of a lexical entry (1) anchors the entry to a pertinent concept in the ontology with relevant properties, (2) captures the multiplicity of meanings in separate senses, and (3) represents synonymic relations. If no directly pertinent concept is available, the most immediate parent and relevant properties are provided in the semantic structure, so that the potential concept is “emulated” in the entry. The syntactic structure of an entry (1) captures morphological information and syntactic properties of a sense and (2) comprises possible syntactic variations (transitivity/intransitivity; case-role alternation, etc.) Ref. [2] provides an exhaustive discussion of the lexicon structure and acquisition within OST, [4] outlines techniques of lexicon management, and [6] offers insights into the development of an automatic lexicon acquisition toolbox.

Two senses for the entry “drink” below illustrate a noun and verbal entries in the lexicon:

```
(drink
  (drink-n1(anno(def "a beverage")
  (comments ""))(ex "she ordered a drink")
  (synonyms ""))
  (cat(n))
  (syn-struct((root($var0))(cat(n))))
  (sem-struct(drink))
  )
  ...
  (drink-v2
  (anno(def "to ingest any liquid")
  (comments ""))(ex "he drank some water")
```

```
(synonyms "" )(cat(v))
(syn-struct
((subject((root($var1))(cat(np))))
(root($var0))(cat(v))
(directobject((root($var2))(cat(np))))))
(sem-struct(swallow
(agent(value(^$var1(should-be-
a(sem(animal))))))
(theme(value(^$var2(should-be-
a(default(drink))))))))))
```

The noun sense “drink-n1” presents a simple case where a concept is available in the ontology onto which the entry can map: the semantic structure contains the concept DRINK. This tells the parser that all other pertinent properties and their fillers should be extracted from the ontological definition of DRINK and other properties that have DRINK in their DOMAIN or RANGE. The verb sense “drink-v2” presents a more complex case where no one-to-one mapping between a sense and a concept is available. Following the considerations of parsimony, application requirements and granulation level in the ontology (see [2] for a more detailed discussion of ontological acquisition), the event of drinking has not been acquired in the ontology; therefore its nearest parent SWALLOW is selected. The two variables \$var1 (indicating a subject) and \$var2 (indicating a direct object) in the syntactic structure restrict the position of the head concept SWALLOW in the sentence. Their semantic counterparts, variables ^\$var1 and ^\$var2, indicate the case roles of AGENT and THEME and link them to the syntactic positions of the subject and the direct object. The facet SHOULD-BE-A restricts the case role fillers to the concepts ANIMAL (for the AGENT case role) and DRINK (for the THEME case role).

The OST text parser is a multi-modular dynamic resource which translates natural language input (clauses, sentences, paragraphs and texts) into a Text Meaning Representation (TMR) – a configuration of ontological concepts with respective property fillers. When processing a sentence, the parser starts building a TMR by identifying an EVENT and then fills its case roles with other concepts from the input by reading the information from the tagger, the stemmer, and the syntactic structure and the semantic structure of the entries available in the lexicon. Prepositional phrases (e.g. “the car of my brother”, compound noun phrases (e.g. “history lecture”), adjectival phrases (e.g. “a dangerous animal”), adverbial phrases (e.g. “run fast”), unattested input (e.g. “Georhe Bush had a meeting”, which contains a typo) and number expressions (e.g. “I bought two books”) are computed by special modules. Parametric (i.e. defined through an ATTRIBUTE value) verb

and noun senses (e.g. “he finished talking”, “the speed of light”) are processed by a separate module. Additional EVENT’-s are treated as clause-forming and are assigned separate TMR’-s, which are then merged with other TMR’-s from the sentence by an event-merging module. The produced TMR’-s are combined in larger entities and then further processed by higher-level modules.

2.2 Modeling priming-based disambiguation in OST

The current functionality of OST enables the system to determine polysemy, compute deixis and reconstruct a number of inferences (see [7] on the computation of unintentional inferences). A number of works illustrates the capability of the OST system to emulate abductive reasoning by applying inference rules ([3], [5]).

Given the descriptive power of the static resources and the computational capacity of the dynamic resources, it is also possible for the OST parser to employ priming-based ambiguity resolution. Based on the description of priming and its conditions provided in previous sections priming can be defined in terms of OST as the mechanism of evoking a disambiguating concept via a lexical item in the immediately preceding context.

It is important to specify conditions for the activation of the priming-based disambiguation mechanism. The mechanism would need to be called for when ambiguity is detected. In terms of OST, ambiguous input is defined as two (rarely more) identically ranked TMR’-s.

Let us consider how the priming-resolved ambiguous example in 1(a) would be computed within OST. For the sentence, two identically ranked TMR’-s would be produced:

1(a) *He enjoyed rivers, so he waited by the bank.*

```
TMR1 (awe
(experiencer(value(human1)))
(theme(value(river1)))
(effect(value(wait1
(agent(value(human2
(gender(value(male))))))
(location(value(land1(beside
(default(river2))))))))))

TMR2 (awe
(experiencer(value(human1)))
(theme(value(river1)))
(effect(value(wait1
(agent(value(human2
(gender(value(male))))))
(location(value(commercial-
building1(occupied-by(sem(bank1))))))))))
```

The two TMR's result from two lexical senses for "bank" in the lexicon, whose semantic structures contain concepts BANK and LAND, which can act as fillers for the same case role LOCATION:

```
(bank-n1
  (cat(n))(synonyms "")
  (anno(def "financial organization")
  (comments "")(ex "he worked at a bank"))
  (syn-struct((root($var0))(cat(n))))
  (sem-struct(bank))
)
```

```
(bank-n2
  (cat(n))(synonyms "")
  (anno(def "a waterfront")
  (comments "")(ex "he stayed on the bank"))
  (syn-struct((root($var0))(cat(n))))
  (sem-struct(land(beside(sem(river))))))
)
```

Given the way the input is presented, the parser correctly processes the disambiguating concept RIVER and its relation to the concept LAND from the sense "bank-n2", but does not "see" it as primed in the previous clause. A rule therefore is needed that would enable the parser to identify the relation between RIVER and LAND via the property BESIDE and re-rank the TMR's on that basis.

Let us now turn to the example 2: "He was knocked over by the punch".

Unlike (1), where ambiguity resides within the OBJECT branch, in example 2, the noun senses "punch-n1" and "punch-n2" evoke EVENT and OBJECT, which defines different case roles and alters the whole reading.

```
(punch-n1
  (anno(def "a thrusting blow")
  (comments ""
  (ex "a nice punch of the ball by the batter"))
  (synonyms "")(cat(n))
  (syn-struct((root($var0))(cat(n)))
  (pp-adjunct((root(of))(opt(+))(cat(pre))
    (obj((root($var1))(cat(np))))))
  (pp-adjunct((root(by))(opt(+))(cat(pre))
    (obj((root($var2))(cat(np))))))
  (sem-struct(punch
  (theme(value(^$var1)))
  (agent(value(^$var2))))))
)
```

```
(punch-n2
  (cat(n))(synonyms "")
  (anno(def "an alcoholic beverage")
  (comments "")(ex "he drank the punch"))
  (syn-struct((root($var0))(cat(n))))
  (sem-struct(alcoholic-drink))
)
```

Sentences 2 (a-b) illustrate how the sentence can be disambiguated by priming:

- 2 He was knocked over by the punch
- He picked a fight and was knocked over by the punch
 - He ordered a strong drink and was knocked over by the punch

For 2(a), the OST parser would produce the following TMR's:

2 He picked a fight and was knocked over by the punch

```
TMR 1: (fight1(phase(value(begin)))
  (agent(value(human1(gender(value(male))))))
  (effect(default(punch1(effect(sem(drop)))
    (experiencer(value(human2
      (gender(value(male))))))))))
  ))
```

```
TMR 2: (fight1(phase(value(begin)))
  (agent(value(human1(gender(value(male))))))
  (effect(value(be-intoxicated1
    (effect(sem(drop1)))
    (instrument(sem(alcoholic-drink1)))
    (experiencer(value(human2
      (gender(value(male))))))))))
  ))
```

Based on the semantic structures of the lexicon entries "punch-n1" and "punch-n2", the parser will compute the relation of the anchored concepts PUNCH and the BE-INTOXICATED (which has been elliptically reconstructed from "punch-n2") to the head event FIGHT through the available property EFFECT, whose RANGE filler EVENT is an ancestor of both. Based on the ontological definition of FIGHT, the system has correctly prioritized PUNCH as the default filler of EFFECT of FIGHT. Similarly to example 1, however, the parser is still "blind" to the correctly extracted disambiguating clues.

A general rule can therefore be formulated that would instruct the parser to resolve ambiguity by checking for semantically related primes in the immediately preceding clause.

1. For two equally ranked TMR's of one clause, check if they:

- have identical head EVENT but different case role fillers, or
 - have different head EVENT's
- // call the differing case role filler or head EVENT A
 // call any other concept evoked by the lexical item C
2. In the preceding clause:

- (a) find a lexical item evoking A or its ancestor // do not go higher than three levels; or
 - (b) find a C for which there is a PROPERTY P so that (C(P(default(A)))) = true
 - (c) find a C for which there is a PROPERTY P1 that has a range filler R, which is also in the domain of (P(default(A))) so that (C(P1(R(P(default(C)))))) = true
3. If 2(a) or 2(b) or 2(c) return true, assign greater value to the TMR in question.

When applied to each of the competing TMR's, the rule allows establishing semantic relations between the prime and the target based on ancestral relations and the information about default property fillers in the ontology.

Example 1(a) would return true for 1(a) of the rule because (land(beside(default(river)))) and (commercial-building(occupied-by(sem(bank)))) act as fillers for the case role LOCATION for the same event WAIT. It would also return true for 2(b) of the rule because the sem-struct of the lexical entry "bank-n2" (which, in the absence of a concept for "riverbank", is meant to emulate it) contains the concept LAND filling the property BESIDE by default:

He enjoyed rivers, so he waited by the bank.
 (C(P(default(A))))
 (river(beside(default(land))))

Similarly, example 1(b) will return true for 2(b) of the rule because the concept MONEY in the ontology will have its default filler of the property HAS-LOCALE restricted to (commercial-building(occupied-by(sem(bank)))):

He needed to deposit the money, so he waited by the bank.
 (C(P(default(A))))
 (money(has-locale(default(commercial-building(occupied-by(sem(bank))))))

Example 2(a) will conform to 1(b) of the rule because the same property EFFECT is filled by different events PUNCH and BE-INTOXICATED in two TMR's. This example will also return true for 2(a) of the rule because the concept FIGHT in example 2(a) will have PUNCH as the default filler for the property EFFECT:

He picked a fight and was knocked over by the punch.
 (C(P(default(A))))
 (fight(effect(default(punch))))

Example 2(b) will return true for 2(a) of the rule because the entries "strong-drink-n1" and "punch-n2" are anchored in the same concept ALCOHOLIC-DRINK:

He ordered a strong drink and was knocked over by the punch.
 (C(P(default(A))))
 (alcoholic-drink(theme-of
 (sem(buy(effect(sem(get-intoxicated(instrument(default(alcoholic-drink))))))))))

The function of computing more remotely related concepts is expressed in 2(c) of the rule. This function allows the parser to detect priming in cases like 1(ii), where WATER, DUCK, FISHING would not have properties immediately connecting them to (land(beside(river))). The procedure evoked is similar to resolving cases of remotely related compound nouns like "theater ticket" and involves searching for properties whose domain and range are controlled by other properties.

In the sentence "He enjoyed the ducks so he waited by the bank", DUCK would have the default filler for HAS-LOCALE restricted to BODY-OF-WATER (an ancestor of RIVER), which is in the domain of the property BESIDE, which, in turn, is listed in the lexical definition of a riverbank in "bank-n2":

He enjoyed the ducks so he waited by the bank.
 (C(P1(R(P(default(C))))))
 (duck(has-locale
 (default(river -is-a- body-of-water
 (beside(default(land))))))

To summarize, priming-based semantic disambiguation is part of the human ability to resolve ambiguity and its emulation is an necessary prerequisite for any NLP system that strives to function at the human-like complexity. Priming-based disambiguation can be efficiently employed by the Ontological Semantic technology. Within the framework of OST, the mechanism of disambiguation by priming translates into the evocation (mostly with a lexical entry) of semantically related concepts across adjacent clauses. Semantic relatedness between the prime (i.e. the evoking concept from prior context) and the target (i.e. the evoked concept) can be captured with an inference rule. The rule operates on rich ontological and lexical knowledge resources, particularly the DEFAULT facet for ontological property fillers, and enables the parser to detect relatedness through class-subclass relation or by finding common property fillers or properties in the immediate context.

3 References

- [1] J. Katz, and J. Fodor, "The Structure of a Semantic Theory." In *Language*, vol. 39, No. 2, pp. 170-210, April-June 1963
- [2] S. Nirenburg and V. Raskin, *Ontological Semantics*. Cambridge, MA: MIT Press, 2004.
- [3] M. Petrenko, "Ontological semantics and abduction: parsing ellipsis," in *Papers from the Annual international Conference "Dialogue 2009"*, vol. 8(15), pp. 598-604, Moscow, Russia, May 2009.
- [4] M. Petrenko, "Lexicon Management in Ontological Semantics" in *Papers from the Annual international Conference "Dialogue 2010"*, Moscow, Russia 2010, in press.
- [5] M. Petrenko and V. Raskin, "Modeling Abduction within Ontological Semantics" in *Proceedings of Midwestern Computational Linguistics Colloquium 5*. East Lansing: Michigan State University, May 2008.
- [6] J. M. Taylor, C.F. Hempelmann, and V. Raskin, "On an automatic acquisition toolbox for ontologies and lexicons" In ICAI'10, in press.
- [7] J. Taylor, V. Raskin, C. F. Hempelmann and S. Attardo. "An Unintentional Inference and Ontological Property Defaults". 2010, in *SMC 2010: IEEE International Conference on Systems, Man, and Cybernetics*, Istanbul, Turkey, October 2010, in press.
- [8] E. Tulving and D. L. Schacter. "Priming and Human Memory Systems", in *Science*. New Series, vol. 247. No. 4940, pp. 301-306, January 1990.