Advances in Cognitive Systems 4 (2016) Submitted 4/2016; published 6/2016

Decisions for Semantic Analysis in Cognitive Systems

Marjorie McShane MARGEMC34@GMAIL.COM
Sergei Nirenburg ZAVEDOMO@GMAIL.COM
Cognitive Science Department, Rensselaer Polytechnic Institute, Troy, NY, 12180

Abstract

Over the past couple of decades, semantic analysis has been outside of the purview of mainstream natural language processing, but the recent surge of interest in cognitive computing has returned it to center stage. However, what exactly is semantic analysis? What methods can be used to achieve it? What counts as a useful result? How does one measure progress? All of these, and many more, issues define the choice space for building semantic analysis systems. This paper presents an inventory of parameters and values that contributes to this choice space. It then shows how different combinations of parameter values lead to different system profiles using three implementations of the theory of Ontological Semantics. We argue that published descriptions of systems should explicitly state both the choices made and the positive and negative consequences of those choices. This will counter the current tendency for system after system to tacitly reflect the same choices without overtly motivating or justifying them.

1. Introduction

When selecting problems to work on and methods to use in solving them, AI researchers face a plethora of scientific, technical, sociological and strategic considerations. In the best case, this selection process should be fully overt and conscious. Experience shows, however, that too often important decisions are made after less than a thorough reckoning about their prospects and consequences. This is true both at the "macro" level of entire fields of study[1] and at the "micro" level of a single research program. In the latter case, reckoning about choices should take place regularly; otherwise, a research program runs the risk of becoming obsolete.

This paper describes historical shifts within a research program called Ontological Semantics (OS) (Nirenburg and Raskin, 2004). Although the basic goals and tenets of the theory of OS have remained stable since its inception in the 1980s, the actual program of research, as well as the systems implementing the theory, have changed markedly over time. In fact, just recently we undertook a new implementation of OS text understanding to reflect two new priorities: incremental rather than sentence-level text analysis, and a concentration on dialog rather than mainstream NLP tasks (machine translation, question-answering, etc.). Deciding to fully retool an approach 30 years in the making was not easy or risk-free. As part of the decision-making process, we formulated the parameters that have affected development choices over the years and traced how the relative importance of these parameters evolved over time. As a result, this paper has a two-fold utility: a) it proposes some parameters and values that can contribute to a checklist

[1] See, e.g., Nirenburg and McShane (forthcoming) for a discussion of field-level preferences in natural language processing.

Table 1. Parameters affecting the development of semantic analysis systems, along with their values in three implemenations of OS: Mikrokosmos (1993-2003), OntoSem (2003-2015) and OntoSem2 (2015-). Thick borders and shading indicate the same value for a parameter across implementations.

<table>
<tr><th></th><th>Parameters</th><th>Mikrokosmos</th><th>OntoSem</th><th>OntoSem2</th></tr>
<tr><td>1</td><td><b>Analysis vs. Generation</b></td><td colspan="3">Primary focus on analysis</td></tr>
<tr><td>2</td><td><b>R & D balance</b></td><td colspan="3">Approximately equal R&D</td></tr>
<tr><td>3</td><td><b>Time frame</b></td><td colspan="3">3-5-year funding horizons</td></tr>
<tr><td>4</td><td><b>Priorities</b></td><td colspan="3">Dictated by funded projects, which reflect research interests</td></tr>
<tr><td>5</td><td><b>Workforce</b></td><td colspan="3">Academic, average ~ 6 individuals at a time</td></tr>
<tr><td>6</td><td><b>Depth of analysis</b></td><td>The meaning of overt strings</td><td colspan="2">Beyond the meaning of overt strings</td></tr>
<tr><td>7</td><td><b>Analysis end point</b></td><td>Text meaning representation</td><td colspan="2">Meaning grounded in agent memory</td></tr>
<tr><td>8</td><td><b>Quality needs</b></td><td>The best quality that could be achieved</td><td colspan="2">Very high quality and confidence</td></tr>
<tr><td>9</td><td><b>Languages</b></td><td>Strongly multi-lingual</td><td colspan="2">English focus (to date)</td></tr>
<tr><td>10</td><td><b>Role of ontological scripts</b></td><td>Limited; more theoretical than practical</td><td colspan="2">Extensive: for simulation and reasoning</td></tr>
<tr><td>11</td><td><b>Learning of lexicon and ontology</b></td><td>None</td><td colspan="2">Learning of both lexicon and ontology</td></tr>
<tr><td>12</td><td><b>Level of psychological inspiration/plausibility</b></td><td colspan="2">Moderate</td><td>High</td></tr>
<tr><td>13</td><td><b>Architecture: incrementality & pipeline</b></td><td colspan="2">Sentence-level processing; pipeline</td><td>Incremental; not a strict pipeline</td></tr>
<tr><td>14</td><td><b>Confidence metrics</b></td><td colspan="2">Limited</td><td>Extensive</td></tr>
<tr><td>15</td><td><b>System Status</b></td><td colspan="2">Prototype</td><td>Aiming for deployed systems</td></tr>
<tr><td>16</td><td><b>Text genres</b></td><td>Formal texts</td><td>Formal & dialog</td><td>Primarily task-oriented dialogs</td></tr>
<tr><td>17</td><td><b>Applications</b></td><td>Machine translation, question-answering, etc.</td><td>Various</td><td>Human-agent collaboration</td></tr>
<tr><td>18</td><td><b>Inspectability of processing</b></td><td>Limited</td><td>Moderate</td><td>Extensive</td></tr>
<tr><td>19</td><td><b>Imported engines</b></td><td>Preprocessor</td><td>Preprocessor & syntactic parser</td><td>All CoreNLP engines</td></tr>
<tr><td>20</td><td><b>Support of non-linguistic perception</b></td><td>None</td><td>Interoception</td><td>Interoception; planned integration with other sensors</td></tr>
<tr><td>21</td><td><b>Interaction of linguistic and non-linguistic reasoning</b></td><td>Minimal</td><td>Moderate</td><td>Extensive</td></tr>
</table>

to support conscious choices about *what* to do and *how* to do it; and b) it discusses specific choices – and their consequences – in the subarea of language processing in cognitive systems.

The time is right for the field of cognitive systems to engage in a meta-analysis of this type due to the recent surge of interest in cognitive computing, both scientifically and in the public sphere at large. The buzz around cognitive computing represents a thaw in the alleged winter of artificial intelligence, but history warns us not to overpromise. Cognitive computing requires human-level reasoning, and human-level reasoners require as input formal, unambiguous meaning representations that address all of the challenges posed by natural language, such as ambiguity, ellipsis and implicature. However, the challenges of human-level semantic analysis have not magically become any easier, so the question is, where do we actually stand in attaining this goal? Our community is poised to make fundamental progress on comprehensive language understanding within embodied cognition. However, planning a realistic path of progress requires understanding all of the challenges posed by natural language, having a plan to solve all of these challenges, and delivering systems on a staged timeline that shows useful progress.

2. The Spoiler

In service of clarity – though to the detriment of compelling storytelling – we start with a summary (Table 1) of key parameters affecting the development of semantic analysis systems, along with their values in three successive implementations of OS: Mikrokosmos (1993-2003), OntoSem (2003-2015) and OntoSem2 (2015-). In the text of the paper, parameters are referred to by their numbers in curly brackets. Readers well acquainted with computational semantics should find the table largely self-explanatory. Others will learn about the parameter space in the course of the system descriptions to follow. However, first let us briefly introduce the theory of OS underlying all three implementations.

3. The Theory of Ontological Semantics (OS)

The theory of Ontological Semantics (OS) describes an approach to automatically extracting the full, ontologically-grounded, contextual meaning of natural language input, which is then used as interpreted knowledge to support language processing applications. The concepts used in text meaning representations (TMRs) are drawn from a property-rich ontology currently containing around 9,000 concepts. The translation from text strings to concepts is supported by a lexicon currently containing around 30,000 word senses. Each word sense features linked syntactic and semantic structure zones, with the latter describing the meaning of words and phrases using the ontological metalanguage. For example, the lexical sense for the multiword expression *apply pressure to*, recorded as the fifth verbal sense of *apply* is shown below. We use a semi-formal representation for readability's sake.

```
apply-v5
   def      "phrasal: 'apply pressure to'; press in a physical sense"
   ex       "She applied pressure to his chest."
   syn-struc
      subject        (root $var1) (cat np)
      v              (root $var0) (cat v)
      directobject   (root $var2) (cat n) (root pressure)
      pp             (root $var3) (cat prep) (root to) (obj ((root $var4) (cat np)))
   sem-struc
      PRESS
         AGENT     ^$var1
         THEME     ^$var4
      ^$var2    null-sem+
      ^$var3    null-sem+
```

The syn-struc zone asserts that, in the active voice, the verb *apply* takes a subject, a direct object headed by the word *pressure*, and a prepositional phrase headed by *to*. Each syntactic element is associated with a variable. The sem-struc asserts that the event in question is the ontological concept PRESS, whose AGENT is the meaning of $var1 (^ indicates "the meaning of") and whose THEME is the meaning of $var4. The ontological description of PRESS – which is consulted during lexical disambiguation – includes the information that the AGENT must be ANIMATE, whereas the THEME must be a PHYSICAL-OBJECT that is NOT ANIMATE. (So, an input like *He applied pressure to his employees* will be treated by a different lexical sense whose prepositional object is constrained to HUMAN.) The descriptor "null-sem+" indicates that the meaning of these elements should not be computed separately: it has already been taken care of in the overall meaning representation.

Below is a the pretty-printed text meaning representation (TMR) for the input *You need to apply pressure to the wound,* excluding metadata and frames showing inverses:

```
REQUEST-ACTION-1
    AGENT          HUMAN-1             ; the speaker
    THEME          PRESS-1
    BENEFICIARY    HUMAN-2             ; the interlocutor
PRESS-1
    AGENT          HUMAN-2
    THEME          WOUND-INJURY-1
```

The TMR is headed by a numbered instance of the concept REQUEST-ACTION, which is the interpretation of *you need to* when uttered in a task-oriented dialog.[2] The AGENT of this action is the HUMAN speaker and its THEME (what is requested) is an instance of PRESS. This instance, PRESS-1, is further specified in its own frame as having the HUMAN interlocutor as its AGENT and an instance of WOUND-INJURY as its THEME. The ontology provides further information about each concept. For example, PRESS is the child of APPLY-FORCE. Among its property-value pairs are case-roles that support lexical disambiguation and agent reasoning, including (AGENT ANIMATE), (INSTRUMENT LIMB, DEVICE), (THEME PHYSICAL-OBJECT). In short, the meaning of ontological concepts is defined by their inventory of property values and how they participate in the scripts that drive simulated agent action and reasoning.[3]

TMRs represent knowledge structures of the type that the reasoning community has been needing – and handcrafting – for decades. The OS approach to meaning analysis is the antithesis of so-called "upper-case semantics," which refers to the tradition of sidestepping natural language challenges like ambiguity and semantic non-compositionality by asserting that strings written using a particular typeface (often, uppercase) have only the single, intended meaning. By contrast, OS takes responsibility for all challenges of natural language including lexical and referential ambiguity, multi-word expressions, nominal compounds, ellipsis, fragments, indirect speech acts, and more.

OS is a language-independent theory of meaning, so the ontology, the fact repository, the TMRs, and all agent reasoning rules are written in the same language-independent, ontologically-grounded metalanguage (McShane & Nirenburg, 2012).[4] Even many aspects of the lexicon – such

[2] Outside of a task-oriented dialog, *one must* conveys only a high value of obligative modality. Within a task-oriented dialog, *you must* has the further implication "so do it".

[3] For scripts in OS, see McShane et al., 2007. For reasoning, see Nirenburg et al., 2008.

[4] Individual agents have their own ontologies, which contain a combination of universal knowledge and agent-specific knowledge: e.g., all agents know what a STOMACH is, but only medically informed agents know what a LOWER-

as the semantic descriptions of many words – are language-independent, leading to the time-efficient porting of lexicons across languages (McShane et al., 2005a). We leave additional details about OS to the descriptions of implementations to follow.

4. The Implementations

Over the 25+ years of its development, OS has been the theoretical substrate for three markedly different systems. The first two, Mikrokosmos and OntoSem, represent the evolution of a single code base, whereas the third, OntoSem2, is being developed from scratch. Throughout the history of OS, the values for five parameters in our inventory have remained stable: analysis has been pursued more rigorously than generation {1}; approximately equal effort has been spent on research and development, with the academic orientation pushing the research angle but the need for system-based validation driving development {2}; the funding horizon has been 3-5 years {3}, with the focus of work at any given time reflecting projects goals {4}; and the workforce has been mainly academic personnel, averaging around 6 individuals at a time {5}. Here the similarities across all implementations end, and we switch to top-down descriptions of each system, ordered historically.

4.1 Mikrokosmos: 1993-2003

The Mikrokosmos language analyzer (Beale et al., 1995) was initially developed in service of interlingual machine translation (MT) {9, 17}.[5] Language analysis resulted in sentence-level TMRs that could be used as input to generators for any language. Text analysis was prioritized over generation {1} since the automatic creation of knowledge structures is a prerequisite to any scalable approach to generation: that is, generation systems have to generate language strings from *something* and, to date, that something is manually crafted for most systems.

Although all language phenomena were, in principle, in purview, the MT application suggested certain priorities, such as lexical disambiguation, the establishment of the semantic dependency structure, and the processing of long and syntactically complex sentences. Of lower priority were phenomena that could typically be carried over in translation without deep analysis, such as ellipsis, implicatures, and referring expressions {6, 7}.[6] The goal was to achieve the highest-quality translations possible within a largely research-oriented effort {8}. Although script-based reasoning was included in the theory of OS, it was operationalized only for demonstration examples {10}. There was no explicit learning of lexicon or ontology, but meanings for unknown words could be hypothesized on the fly using unilateral selectional constraints: e.g., *quood* could be understood to be an INGESTIBLE from the input *John ate some quood* {11}.

ESOPHAGEAL-SPHINCTER is. As concerns language- and culture-specific interpretations of basic ontological concepts, these can be conveyed by property values added to the relevant word senses in the lexicon.

[5] This was before the ascendancy of statistical methods supported by large parallel corpora. MT is among the most suitable applications for statistical processing since many ambiguities, implicatures, ellipses, etc., can be directly passed from one language to the other, with the human consumer responsible for resolution. Language understanding by task-oriented intelligent agents, by contrast, is far more demanding: the agent must resolve all of these reasoning-heavy ambiguities and underspecifications.

[6] Of course, there are exceptions: e.g., plural pronouns in Hebrew have gender, introducing an extra choice in English-to-Hebrew translation.

In Mikrokosmos, sentences were processed as a whole, using knowledge-based strategies supplemented by computational methods for reducing combinatorial complexity. A program called Hunter-Gatherer (Beale et al., 1996) leveraged two tools for achieving higher efficiency in problem solving: hunting (reducing the search space by looking for suboptimal or impossible solutions and removing/killing them) and gathering (efficiently extracting answers to subproblems; collecting and combining satisfactory answers). Weighted preferences were also combined to decide, for example, whether it was preferable to satisfy syntactic or semantic constraints when both could not be simultaneously satisfied. The contribution of these computations made some aspects of language-oriented decision-making uninspectable {18}. In addition, effort was not devoted to creating visualization tools to trace system processing – an enhancement that was introduced later on.

Although the theory of OS is strongly psychologically motivated, emulating human processing {12} was not a first-order goal of Mikrokosmos. Thus, inputs were analyzed at the level of sentences, and the architecture followed a strict pipeline (preprocessing → syntactic analysis → semantic analysis → pragmatic analysis), which is not how people process language {13}.

All Mikrokosmos processing engines, apart from various preprocessors {19}, were developed in-house. Also developed in-house were the Mikrokosmos ontology and ontologically-linked lexicons of varying sizes for languages including English, Russian, Spanish, Turkish, Chinese and Korean.

Once the basic language processing resources and engines were developed, they were applied to various applications beyond MT, such as information extraction and question answering (e.g., Cowie et al., 2000) {17}. All of these applications were prototypes that worked on open text or within a specific but non-toy domain, such as the Olympic Games or bankruptcy {15}.

The goal of semantic analysis was the production of correct TMRs for individual sentences {7}, sometimes merged into a fact repository. The actual implementation of fact repositories during the Mikrokosmos period was not as advanced as the corresponding memory modeling of intelligent agents in subsequent implementations.

Let us return to the historical context. This was the 1990s, before the ascendancy of the internet, before social media, before vast increases in computing power, before "ontology" became a watered-down term used to describe any hierarchical knowledge base, before the push toward open-source resources, and during the meteoric surge of interest in machine learning in support of knowledge-lean NLP. The most noteworthy point about knowledge-lean NLP for this discussion is that it offers faster results for applications in the near term but it does not address the difficult language phenomena that will be necessary in the long run. In fact, the success of knowledge-lean NLP can be traced to the strategic choices of applications: machine translation systems can leave many ambiguous aspects of text unresolved, shifting that work to the human reader; question-answering systems can rely on redundancies in a large corpus, opting to target simpler formulations of information; and search engines are readily forgiven errors, being the ultimate no-risk application. Even IBM's Jeopardy-playing *Watson* system, though not knowledge-lean, could selectively treat inputs, choosing not to respond at all in cases of low confidence.

By the end of the 1990s it became clear that these mainstream applications would not be the best showcases for systems pursuing full text understanding. Therefore, OS development began to shift focus to reasoning-heavy applications for which knowledge-lean approximations would not suffice.

4.2 OntoSem: 2003-2014

OS expanded into agent applications thanks to the Maryland Virtual Patient (MVP) project. MVP is a clinician-training system in which a cohort of simulated virtual patients can be diagnosed and treated using interactive simulations (Nirenburg et al., 2008). A virtual tutor, along with other supporting medical agents, rounds out the team of intelligent agents. This R&D effort had three foci: a) creating functionally realistic *double agents*, defined as integrated, interactive physiological and cognitive simulations; b) creating a knowledge environment in which the same structured knowledge supported cognitive and physiological simulations (McShane & Nirenburg, 2012); and c) approaching language understanding as a situated, task-oriented problem, affected by the plans, goals and expectations of the dialog participants.

In the context of developing this agent environment, which would come to be known as OntoAgent, the team's attention became widely distributed. We dove into domains ranging from physiological simulation (McShane et al., 2007) to decision-making biases (McShane et al., 2013) {21}. To help speed progress on language understanding with less in-house effort, we incorporated the Stanford CoreNLP dependency parser (Manning et al., 2014) into our processing pipeline {19}. Although this was not without significant initial and ongoing overhead (see McShane et al., Forthcoming, for details), it permitted us delegate responsibility for syntactic analysis so that we could more squarely focus on semantics within the larger model of agent cognition.

OntoSem, like Mikrokosmos, processed full sentences at a go, used a pipeline architecture, and relied on the Hunter-Gatherer engine to manage complexity {12, 13}. The results of language understanding were incorporated into agent memory along with memories generated by interoception (the experiencing of bodily signals), reasoning, and simulated action {20}. Memory thus augmented served as input to the agent's decision functions about its lifestyle and medical treatment.

A core feature of MVP was its extensive "under the hood" panes, which showed real-time traces of many types of agent functioning (Nirenburg et al., 2010). In addition, we developed interfaces that offered both programmers and knowledge engineers better access to intermediate results of language processing, which fostered testing and debugging of both the language processing engines and the supporting knowledge bases {18}.

As concerns learning, OntoSem introduced methods for agents to learn elements of lexicon and ontology through interaction with a human user {11}. For example, virtual patients in MVP could learn the names and properties of diseases, tests and treatments through interaction with the human playing the role of attending physician.

An essential component of MVP was the ontological scripts used to model the interactive, physiological simulations. This same script-based knowledge also supported language understanding and agent reasoning {10}.

During the OntoSem period, quite separately from the work on MVP, we continued to enhance the microtheories devoted to such issues as lexical disambiguation (McShane et al., Forthcoming), multiword expressions (McShane et al., 2015), nominal compounds (McShane et al., 2014), reference resolution (McShane & Nirenburg, 2013), and the processing of fragments and ellipsis (McShane et al., 2005b). Some of these algorithms were implemented and evaluated on open text, whereas others remained on the stack for later integration.

To sum up, during the OntoSem period our attention broadened to agent modeling overall, with language understanding being only one contributor to agent cognition. As part of this process, we became more concerned with both psychological plausibility and the inspectability of processing results, since an agent's confidence in its language understanding must inform its decision-making about action. We determined that sentence-level processing using a pipeline architecture would not provide sufficiently human-like behavior, and decided to experiment with incremental parsing. We also sought to render all language-oriented decision-making as inspectable as possible so that the agent could achieve fine-grained estimates of confidence in its language understanding. For these reasons, we halted development of OntoSem and applied all of the available knowledge resources and algorithms to a completely new implementation, OntoSem2.

4.3 OntoSem2: 2015-

OntoSem2 focuses on language understanding for task-oriented dialogs {16}, which are expected to contain short, highly elliptical utterances whose full interpretation requires world knowledge, situational understanding, and the treatment of difficult linguistic phenomena.

OntoSem2 processes inputs incrementally and permits all heuristic evidence to be leveraged as soon as it becomes available, in a largely non-pipeline architecture {13}. This will offer human-like possibilities such as beginning to act in mid-sentence and interrupting for clarification {12}. For example, if a person tells a robot, "Take the nylon rope and tie it firmly to the fender of the car", the robot should already start reaching out toward the nylon rope after the 4th word is pronounced. Furthermore, if the robot doesn't understand a necessary aspect of input – e.g., "nylon" could get pronounced during a loud crash – it should immediately interrupt to ask for clarification.

Incrementality makes it natural to leverage any and all heuristic evidence as soon as it becomes available, avoiding the drawbacks of traditional, but cognitively implausible, pipeline architectures. For example, given the fragment, "*The pain started in my abdomen and then it"*, the system can hypothesize that *it* is referential (not pleonastic) and refers to *the pain* (McShane and Babkin, Forthcoming). This hypothesis, though overridable, immediately reduces the choice space for disambiguating the upcoming verb.

A known challenge of incrementality is the potential explosion of ambiguity while the system awaits disambiguating elements of input. For example, sentence-initial "*It made*" can have many interpretations until we know what comes next. However, the numerical threat is not nearly so imposing with dialog as with more formal text genres, since dialog tends to be organized into word packets of around 7 to 10 words (McWhorter, 2013). Such chunks can be: (a) complete sentences like *I'm hungry*; (b) fragments contributing to complete sentences like *I'm hungry / and a little bit nervous too*; (c) free-standing fragments like *How many scoops do you want? Three*, or (d) incomplete structures due to interruptions, changes in one's train of thought, distractions, or the availability of extra-linguistic support for one's meaning, like *"Hammer!"* meaning *"Pass me the hammer fast!"*

Deciding to pursue incrementality leads to a host of other decisions about implementation. We dive deeply into parameters involving incrementality because if semantic analysis is a rarity, all the more so is incremental semantic analysis.

4.3.1 Parameters Involving Incrementality

Although this inventory of parameters is similar in nature to the one presented in Table 1, we present it in running text rather than tabular format since we are discussing only one of many possible values for each parameter.

INCR-Parameter 1. Does OntoSem2 use a syntactic parser that is optimized for incremental processing? No. For preprocessing and syntactic analysis, OntoSem2 continues to use the preprocessor and parser of the Stanford CoreNLP toolkit. We are not modeling human-like syntactic processing for the following reasons: (1) Working on syntax would pull resources from our main interest, which is semantics and pragmatics. (2) Syntax has dominated the field of NLP for the past 40 years, meaning not only that quite good parsers are available, but that it is high time to pay attention to other things. (3) OS makes no theoretical claims about how syntactic structures are built; it simply uses syntactic evidence, to the extent it is available, as one type of heuristic evidence for semantic analysis. (4) Incremental syntactic parsers exist (e.g., Ball et al., 2014) and could replace our current parsing method if their results proved better, or if one wanted to make stronger claims about the alignment of agent cognition with human cognition. (5) Even though CoreNLP is not optimized for incremental parsing, it does a reasonably good job of it. We expect that parsing errors – like errors in semantic and discourse analysis – will be more frequent for fragments than for full sentences and are working toward enabling the agent to function as well as possible in this non-optimal situation. The incorporation of not purely-knowledge-based systems into our knowledge-based environment has the benefit of advancing system development but at the expense of full explanatory power. We accept that compromise in order advance what we consider the most useful and scientifically interesting aspects of OS.

INCR-Parameter 2. Does OntoSem2 use other externally generated results? Yes, it uses CoreNLP named entity recognition and reference resolution, the latter described in Parameter 4. We are open to using the results of other engines as well, if they can produce results without requiring automatically unachievable prerequisites (something not true of many reported systems).

INCR-Parameter 3. How strictly incremental is the system? Syntactic analysis is run after every word,[7] so the parser is called after sentence-initial "The", then "The big", then "The big white", etc. We are not concerned about efficiency or processing speed at this time (which is another choice within this choice). As regards semantic/pragmatic analysis, it is run only after every noun or verb. We find it superfluous, e.g., to dwell on what an agent's semantic interpretation of sentence-initial *The* in isolation should be, and we consider it of low priority to optimize the semantic interpretations of other fragments that are missing a semantically weighty head, such as *The very* or *The very big* (exceptions below).[8] So, for the sentence, *The big white horse ran into the barn after the gunshot,* semantic/pragmatic analysis will be launched at 4 points, indicated by slashes: *The big white horse / ran / into the barn / after the gunshot /*. This strategy is intentionally generic: e.g., we do not treat verb forms that might be functioning as auxiliaries differently from main verbs (*have* can serve both functions), and we do not look ahead to determine if nouns are the initial component in a nominal compound (*the tree* will be ultimately be analyzed differently from *the tree structure*). We do, however, account for the special case in which a non-verbal, non-nominal fragment serves as a full utterances: *"What kind of wine would you like?" "Red."*

[7] There are some exceptions, but they are too fine-grained for this paper.

[8] The question, if one chose to pursue it, is: Would *big,* outside of context, be understood as SIZE, WEIGHT or IMPORTANCE? Would the default interpretation change in certain contexts? Does this hold any scientific interest or have any impact on human or agent reasoning? Measuring human responses just because they are measurable, and modeling agents accordingly just because we can, has little scientific or practical rationale.

INCR-Parameter 4. How are syntactic, semantic and pragmatic analysis integrated? Our current processing algorithm, which is under implementation, is as follows: (1) Syntactic and reference analysis is provided by CoreNLP. (2) Three types of pre-semantic processing are run, which offer useful heuristics for upcoming semantic analysis: multi-word expression detection; basic nominal compounding analysis; and homegrown coreference for certain types of difficult referring expressions not handled by CoreNLP. (3) Basic semantic analysis is launched, aimed at lexical disambiguation and the establishment of the dependency structure. (4) Deep coreference resolution is run, which validates or overturns prior coreference votes using semantic evidence; it also establishes co-reference links for so-far unresolved referring expressions. (5) Semantic analysis and deep co-reference analysis are rerun, in turn, to optimize the combination of decisions about lexical disambiguation and reference resolution. (6) Triggered meaning procedures (recorded in the lexicon) are run, to resolve contextually-bound meanings such as *yesterday* and *approximately.* (7) Advanced contextual reasoning is applied, as needed, to handle sentence fragments, non-lexically recorded indirect speech acts, implicatures, etc. (8) TMRs are grounded in agent memory: For known objects and events, new information is added to their anchors in memory, whereas new objects and events establish new anchors.

INCR-Parameter 5. In cases of less than full confidence, are multiple candidate analyses retained? They can be, and currently are. This is an application-specific decision.

INCR-Parameter 6. Can later processing override the results of earlier processing? Yes. For example, surfacy coreference votes can be overridden by semantic analysis: In *The janitor talked to the surgeon and then he operated,* syntactically-oriented coreference will suggest that *he* is coreferential with *the janitor*, whereas semantic analysis will prefer coreference with *the surgeon.*

INCR-Parameter 7. Is reasoning about action interleaved with reasoning about language? Absolutely, as described in McShane and Nirenburg (2015a,b).

Zooming out, we have just considered some of the choices involved in developing an incremental semantic analysis system. Now we turn to how these choices, in conjunction with the larger choice space of Table 1, are manifesting in the nascent OntoSem2 system.

4.3.2 Current State of Development of OntoSem2

We are a little over a year into the development of OntoSem2 and are making fast progress thanks to the availability of previously compiled knowledge resources, precisely described microtheories, and a sober attitude toward choosing our battles. We expect that in the next 6 months domain-neutral language understanding capabilities will be stable enough for a baseline formal evaluation. During this time, we also expect to start building an ontological model to support a new medical application. This model will include physiological and clinical aspects, and will permit robust reasoning by the agent even in linguistically underspecified contexts. For example, if a patient says, “God, my lower gut!” the agent must understand this as the report of severe pain in the lower abdomen. Our past experience in medical modeling suggests that this level of understanding is entirely achievable.

In this implementation, we are paying close attention to the inspectability of processing {18}, both to enhance code and knowledge debugging, and to permit the agent to more accurately gauge its confidence in each instance of language understanding {14}. For example, we can generate three levels of traces of decisions related to lexical disambiguation and semantic dependency determination, and we are well into developing interfaces that show the intermediate and final stages of processing, appended with selectable inventories of metadata.

5. Comparisons with Others

As regards semantic analysis, the programs of research that most closely resonate with OS are Schank's (1972) Conceptual Dependency Theory and Wilks' Preference Semantics (Wilks, 1985; Wilks & Fass, 1992). Unlike OS, however, these theories were not applied to broad-scale text analysis, though they did inspire approaches that were: e.g., Schank-like script-based reasoning inspired the Knowledge Machine project (Clark & Porter, 2004). The generational attribution of these programs of research should come as no surprise for reasons described earlier.

As regards incremental analysis, there are two interesting points of comparison. Jerry Ball's (2014) Double-R incremental parser has successfully treated difficult syntactic structures in English (e.g., *What could he have been given to be eaten?*) but it does not pursue semantics. By contrast, Ruth Kempson's Dynamic Syntax is a purely linguistic (rather than computational linguistic) theory that was used by Purver at al. (2011) as the theoretical substrate for the approach to incremental parsing described in Purver and Kempson (2004) and Purver et al. (2011). This system generates a decorated tree structure that is said to represent the semantic (not syntactic) interpretation of the utterance; however, the system does not engage in lexical disambiguation, instead employing a version of upper-case semantics. In short, they seem to have mechanisms to cover various linguistic phenomena, but they seem to make the same assumptions as do most formal semanticists: that external, upstream processors will deal with problems such as lexical and referential ambiguity, idiomaticity, and indirect speech acts. It is noteworthy, of course, that the closest points of comparison that we could find are not, in the end, very close at all.

6. Final Thoughts

The inventory of parameters and values we presented here is not intended to be complete; instead, it is intended to capture some noteworthy, high-level distinctions between implementations of OS that were describable in a paper of this length. A longer treatment of this topic could have taken a number of different paths, of which we mention just two.

On the one hand, we could have continued to focus on OS implementations but have included more fine-grained parameters and larger value sets. However, in order for those parameters to make sense, we would have had to have provided much more detail about how each system worked. This would be entirely appropriate, and is an excellent idea, for a book-length treatment of implementations of OS.

Another direction for a much longer paper – and, indeed, a dedicated program of study – would have been to have proposed a universal set of parameters that could be used, as applicable, to describe any NLP system. It is interesting to consider some of the things that would have to be included in the most rigorous interpretation of that goal:

1. The individual listing of every linguistic phenomenon that could be encountered. For example, does the system treat broad referring expressions, scalar attributes, quantifiers, heavy-NP shift, sentence fragments, indirect speech acts, and each of hundreds more language phenomena? This inventory could be informed, e.g., by the lists that guide the work of field linguists, or by grammars like Quirk et al.'s *Comprehensive Grammar of the English Language* (1,792 pages!).
2. All possible levels of treatment of each phenomenon. For example, does analysis include morphological, syntactic, semantic and/or discourse analysis?

3. What counts as "treatment" of a given phenomenon? For example, does resolving a referring expression require finding any single member of its textual coreference chain, or all members of its textual coreference chain, or the latter plus grounding the referring expression to agent memory to account for cross-document coreference?
4. What theoretical framework is being used to motivate the approach to describing each phenomenon? For example, does one use 6, 12 or 28 parts of speech to describe the lexical stock of a given language? What are the pros and cons of the selected theoretical framework, and do they differ across applications?
5. What counts as a correct treatment? For example, does the agent have to have a high-confidence answer that it can explain? (If so, how high does the confidence have to be, and what counts as explanation?) Or is the agent's best guess without explanation sufficient, with the likelihood of a correct answer being defined solely by a system-level evaluation metric?

This is just a sampling of the very large inventory of parameters that could be included in a universal inventory. Clearly, not only building, but also using, such an inventory would quickly become overwhelming, which suggests that inventory creation is better informed either by specific scientific questions or by specific practical needs. In our case, as described earlier, we undertook this analytical exercise as a precursor to deciding to halt work on a mature system implementation in favor of developing a new one from scratch. In retrospect, we believe that this sort of programmatic spring cleaning is invaluable and we intend to make it a regular practice.

Before leaving the issue of a universal inventory of parameters, let us mention that the center of gravity of such an inventory – i.e., innumerable linguistic details – would probably be of less interest to the cognitive systems community than the more broad-strokes inventory we presented here. Our main point is to advocate the utility of making manifest one's research decisions, rather than to suggest that there is a single best way to organize the results of that process. If a full menu of parameters and values were to exist, it might speed up analysis work of this kind and permit neater cross-system comparisons. However, we believe there is also utility in looking at one's program of research bare, without any externally-imposed organizational principles, and analyzing which decisions one has made, why, and to what effect. After all, any inventory will be incomplete, and the reliance on an inventory can affect judgment – even leading to missing the most salient features if they are, idiosyncratically, not accounted for.

In closing, although the details of this paper will likely be of interest only to developers of language understanding systems, we hope that the overarching point will be of broader interest. Throughout artificial intelligence and, specifically, cognitive systems, it would be of great benefit if developers would spend more time talking about *why* they are doing what they are doing, framed among the competing options. The recent preoccupation in computational linguistics circles with numerical system evaluations has detracted from what is far more interesting: how can we work toward truly sophisticated intelligent agents, and what part in that big picture does each necessarily small contribution play? Such issues should not be implicit knowledge for the in-crowd. Instead, they should be front and center in scholarship aimed at a wider audience.

Acknowledgements

This research was supported in part by Grants N00014-09-1-1029 and N00014-16-1-2118 from the U.S. Office of Naval Research. Any opinions or findings expressed in this material are those

of the authors and do not necessarily reflect the views of the Office of Naval Research.

References

Ball, J., with Freiman, M., Rodgers, S., & Ball, A. (2014). Double-R Grammar: A Computational Cognitive Grammar of English. Last Updated 27 August 2014. Available at http://www.doublertheory.com/double-r-grammar/index.html

Beale, S., Nirenburg, S. & Mahesh, K. (1995). Semantic analysis in the Mikrokosmos machine translation project. *Proceedings of Symposium on Natural Language Processing*, Kaset Sart University, Bangkok, Thailand.

Beale, S., Nirenburg, S. & Mahesh, K. (1996). Hunter-Gatherer: Three search techniques integrated for natural language semantics. *Proceedings of The Thirteenth National Conference on Artificial Intelligence* (AAAI-96), Portland, Oregon.

Clark, P., & Porter, B. (2004). KM: The Knowledge Machine 2.0: Users Manual. Available at http://www.cs.utexas.edu/users/mfkb/km.html.

Cowie, J., Ludovik, E., Molina Salgado, H., Nirenburg, S. & Sheremetyeva, S. (2000). Automatic question answering. *Proceedings of the RIAO Conference*, Paris.

Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S. J., & McClosky, D. (2014). The Stanford CoreNLP Natural Language Processing Toolkit. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations,* pp. 55–60. Stroudsburg, PA: The Association for Computational Linguistics.

McShane, M. & Babkin, P. (Forthcoming). Resolving difficult referring expressions. *Advances in Cognitive Systems*.

McShane, M., Beale, S., & Babkin, P. 2014. Nominal compound interpretation by intelligent agents. *Linguistic Issues in Language Technology* (LiLT), 10(1): 1-34. CSLI Publications.

McShane, M., Fantry, G., Beale, S., Nirenburg, S., & Jarrell, B. (2007). Disease interaction in cognitive simulations for medical training. Proceedings of the MODSIM World Conference & Exposition, Virginia Beach, pp. 74–80.

McShane, M., & Nirenburg, S. (2012). A knowledge representation language for natural language processing, simulation and reasoning. *International Journal of Semantic Computing*, 6(1): 3–23.

McShane, M. & Nirenburg, S. (2013). Use of ontology, lexicon and fact repository for reference resolution in Ontological Semantics. In Oltramari, A., Vossen, P., Qin, L., Hovy, E. (Eds.), *New Trends of Research in Ontologies and Lexical Resources: Ideas, Projects, Systems*, pp. 157–185. Springer.

McShane, M., Nirenburg, S., and Beale, S. (2005a). An NLP lexicon as a largely language independent resource. *Machine Translation*, 19(2): 139–173.

McShane, M., Nirenburg, S., & Beale, S. (2005b). Semantics-based resolution of fragments and underspecified structures. *Traitement Automatique des Langues*, 46(1): 163–184.

McShane, M., Nirenburg, S., & Beale, S. (2015). The Ontological Semantic treatment of multiword expressions. *Lingvisticæ Investigationes*, 38(1): 73–110. John Benjamins Publishing Company.

McShane, M., Nirenburg, S. & Beale, S. (Forthcoming). Language understanding with Ontological Semantics. *Advances in Cognitive Systems*.

McShane, M., Nirenburg, S., and Jarrell, B. (2013). Modeling decision-making biases. *Biologically-Inspired Cognitive Architectures*, 3: 39–50

McWhorter, J. (2013). Txting is killing language. JK!!! TED2013. Filmed 2013. Available at http://www.ted.com/talks/john_mcwhorter_txtng_is_killing_language_jk/transcript?language=en

Nirenburg, S., & McShane, M. (Forthcoming). Natural language processing. In Chipman, S. (Ed.), *The Oxford Handbook of Cognitive Science*.

Nirenburg, S., McShane, M. & Beale, S. (2008). A simulated physiological/cognitive "double agent". In Beal, J., Bello, P., Cassimatis, N., Coen, M. & Winston, P. (Eds.), *Papers from the AAAI Fall Symposium, Naturally Inspired Cognitive Architectures*. AAAI technical report FS-08-06, Menlo Park, CA: AAAI Press.

Nirenburg, S., McShane, M., & Beale, S. (2010). Aspects of metacognitive self-awareness in Maryland Virtual Patient. In Pirrone, R., Azevedo, R. and Biswas, G. (Eds.), *Cognitive and Metacognitive Educational Systems: Papers from the AAAI Fall Symposium*. AAAI Technical Report FS-10-01. Association for the Advancement of Artificial Intelligence.

Nirenburg, S., & Raskin, V. (2004). *Ontological Semantics*. Cambridge, MA: MIT Press.

Purver, M., Eshghi, A., & Hough, J. (2011). Incremental semantic construction in a dialogue system. *Proceedings of the 9th International Conference on Computational Semantics* (IWCS), Oxford.

Purver, M., & Kempson, R. (2004). Incrementality, alignment and shared utterances. *Proceedings of the 8th SEMDIAL Workshop on the Semantics and Pragmatics of Dialogue*, pp. 85–92, Barcelona, Spain.

Quirk, R., Greenbaum, S., Leech, G., & Svartvik, J. (1985). *A Comprehensive Grammar of the English Language*. Longman.

Schank, R. (1972). Conceptual Dependency: A theory of natural language understanding. *Cognitive Psychology* 3: 532–631.

Wilks, Y. (1985). Text structures and knowledge structures. *Quaderni di Semantica, 6*, 335–344.

Wilks, Y. & Fass, D. (1992). The preference semantics family. *Computing and Mathematics with Applications* 23: 205–221.