# Content-Centric Computational Cognitive Modeling

**Sergei Nirenburg** ZAVEDOMO@GMAIL.COM
**Marjorie McShane** MARGEMC34@GMAIL.COM
**Jesse English** DRJESSEENGLISH@GMAIL.COM
Language-Endowed Intelligent Agents Lab, Rensselaer Polytechnic Institute, Troy, NY 12180, USA

## Abstract

That human-level cognitive systems will require a lot of knowledge is common knowledge. However, for a number of reasons, issues of content are not directly addressed at scale by the cognitive systems community. This paper makes the case for the centrality of content-related issues for the long-term goals of cognitive system research. Topics addressed include: the nature and scope of content-centric cognitive modeling (CCM); the juxtaposition of CCM with work on cognitive architectures and cognitive systems; the importance of addressing long-term knowledge needs immediately, since they cannot be met by gradual enhancements in application systems; the ways in which a content-oriented perspective affects design preferences for cognitive architectures; how CCM minimizes the need for heuristic search; and the practice of CCM in the OntoAgent environment.

## 1. Introduction

There is a lacuna at the core of cognitive systems research. One of the components of work toward developing systems with human-level capabilities of perception, reasoning and action is underrepresented. This component is *knowledge content* of a size and sophistication commensurate with the architectures and systems it serves. Modeling human-level functioning will only become possible if cognitive systems are endowed with non-toy amounts of knowledge covering a non-toy inventory of knowledge kinds and described in detail sufficient to support human-level decision-making in all cognitive tasks. But knowledge acquisition is a time-consuming undertaking that does not bring about an immediate payoff. So, people are finding ways to avoid or bypass addressing content directly:

a) by concentrating on other, equally important, theoretical and engineering issues
b) by selecting application domains where knowledge requirements can be limited
c) by claiming that content-related work is outside the purview of cognitive system research as such and the best we can do is to import whatever content is available from outside sources
d) by looking for alternative ways (such as opaque modeling based on machine learning) of providing fodder for the heuristics driving cognitive systems.

The danger of this state of affairs continuing is that without renewed attention to adequate content the current crop of cognitive systems is probably too close to reaching the maximum of its cognitive functionalities and application scope. Moreover, this maximum falls short of the long-term goals of the community.

This paper motivates an approach to reintroducing work on content into the purview of cognitive systems research. We discuss the objectives of this type of work, analyze how taking the content perspective, in turn, motivates certain preferences in the design of cognitive architectures, and give examples of the use of this paradigm in our team's past and current research. A companion paper (English & Nirenburg, submitted) describes a computational infrastructure for this type of work.

## 2. The Place of Content-Centric Modeling in the Cognitive Systems Enterprise

The inventory of tasks facing developers of cognitive systems is vast – consider the nine large classes of processes described in the community's manifesto (Langley et al., 2009) as cognitive architecture (CA) capabilities, or the thirteen rubrics that Sloman & Scheutz (2002) introduce as dimensions on which CAs can be compared. These processes and dimensions cover a comprehensive range from recognition and categorization, to decision making, to learning and communication. While the manifesto acknowledges the importance of content for cognitive systems, the discussion of content in it addresses the representation, organization, utilization and acquisition of knowledge but does not directly address the actual content for which these tasks are relevant. Sloman and Scheutz mention knowledge bases but do not discuss them. This reflects the general priorities of the developers of cognitive systems: while being recognized as important, content-related issues typically take a back seat.

Assessing this state of affairs, Lieto, Lebiere, & Oltramari (2018) argue that the major current CAs are deficient in that they "only process a simplistic amount (and a limited typology) of knowledge." As a result, the authors claim that "the … mechanisms that [CAs] implement concerning knowledge processing tasks (e.g., retrieval, learning, reasoning, etc.) can be only loosely evaluated and compared … to those used by humans." *A fortiori*, today's cognitively-inspired application systems, by their nature, cannot be expected to help with such evaluation or comparison either – their primary objectives are to achieve a specific well-circumscribed functionality. Lieto et al. recommend that "cognitive architectures should address … the problems concerning the limited size and homogeneity of the encoded knowledge". In support of this recommendation, they describe knowledge-related gaps in some of the most prominent of today's CAs.

While Lieto et al.'s criticisms are valid, we believe they are aimed at the wrong target. That is because CAs, being theoretical constructs, are responsible only for accommodating different types of knowledge, not actually acquiring it. Similarly, CA researchers should not necessarily be held responsible for developing large knowledge bases. CAs can be initially validated using carefully selected subsets of knowledge, along the lines of the broadly-accepted practice in theoretical linguistics of supporting theoretical statements by a small set of carefully selected examples.

However, whereas theoretically-oriented CA research can be absolved of accounting for realistic-scale knowledge, not so the application systems that implement CAs. As a rule, such systems use knowledge bases of limited size – just enough to cover the needs of a particular application. This is true even of contributions that are expressly devoted to knowledge (e.g., Jacobsson et al., 2007). The first order of business in developing such systems is to mitigate the complexity of system engineering and promote system performance. In service of these goals, the knowledge bases in such systems are constrained not just in size but also, as a rule, in the depth of description of the phenomena in the world, language, and the agent's "mind." Spending resources beyond immediate needs of an application is imprudent. Efficiency concerns also dictate the preference for stricter modularization, with system modules using dedicated knowledge bases.

One opinion to the contrary is that the current state of affairs is justified theoretically, and not only by the immediate engineering needs. Consider the recent popularity of the "now-or-never bottleneck" hypothesis, which Christiansen & Chater (2016) claim to be "a fundamental constraint on language".[1] The now-or-never principle essentially states that all decisions in language processing must be local. Developers of cognitive systems have found this a convenient principle (and slogan) to justify simplifications in how they operationalize natural language understanding (NLU). The problem is that the claim utterly falls apart when exposed to the actual phenomena of natural languages – as is convincingly argued in published responses to Christiansen & Chater. To give just a taste, Christiansen & Chater's theory cannot accommodate non-local dependencies, which are a core design feature of natural languages (Levinson, 2016); it does not account for how downstream input affects the interpretation of earlier material, "which shouldn't occur if chunking greedily passes off the early information to the next level" (MacDonald, 2016); it does not account for many pragmatic phenomena that support the communicative function of language, such as clarification, repair, long-distance repetition, and balancing the needs of speakers with those of listeners (Levinson, 2016; Healey et al. 2016; Lewis and Frank 2016); and Christiansen & Chater's "very bottom-up characterization of chunking is inconsistent with evidence for top-down influences in perception" (MacDonald, 2016). Clearly, neither these rebuttals nor Christiansen & Chater's proposal can be sufficiently assessed in this short space. However, the point should be clear: Theories of cognitive functionalities must, from the outset, be informed by a realistic reckoning of the actual scope of eventualities that must be covered. *Awareness of these eventualities is a core aspect of the knowledge that must be recorded in a cognitive system*. If a cognitive system developer adopts now-or-never as a tenet for its NLU component, he or she runs the risk needing to summarily discard this component and redesign it from scratch if the systems using it are ever to advance beyond toy domains. This is tantamount to starting out by building a foundation for a ranch house and expecting to grow it into a skyscraper.

Our insistence that realistic-sized knowledge must be considered from the outset does not mean that we are unaware of, or unsympathetic to, the extenuating circumstances that lead to small-domain development. Developers of cognitive systems have important engineering concerns – for example near-real-time performance, the need for format conversions between the inputs and outputs of the many system modules, and achieving satisfactory performance of the individual modules themselves. Indeed, the complexity of integrating the many diverse processing modules in a comprehensive cognitive system application is formidable even when knowledge coverage is limited (cf., e.g., the robotic learning application described in Scheutz et al., 2017). Concerns related to the breadth and depth of knowledge support must be viewed from several perspectives, including extrascientific ones, such as the funding climate. Still, invoking *a priori* questionable simplifications – such as now-or-never language processing – will predictably lead to failure in the face of realistic inputs.

While understanding the demands of realistic-scale knowledge involves one set of concerns, actually acquiring it involves another. Since knowledge acquisition is expensive, it is natural for cognitive system applications to seek to import whatever knowledge resources are available (cf., for example, Forbus (2018) on the importation of the OpenCYC ontology). However, experience shows that integrating imported resources is fraught with complications. First of all, it's not only that they have lacunae, it is the nature of the lacunae that can be the Achilles heel – as was discussed, e.g., in Mahesh et al (1996) with respect to the utility of CYC for language processing. Also, if

[1] The proposed approach is not new. It was practiced already in the early 1980s, for example, in the Mumble language generation system (McDonald, 1983).

undertaken at scale, the importation of resources is quite costly in terms of effort. Our team learned this the hard way when we decided to substitute the extensive homegrown suite of pre-semantic language processing tools in our language understander, OntoSem, with the resources of the Stanford CoreNLP toolkit (Manning et al., 2014). As described in Chapter 3 of McShane and Nirenburg (in press), aligning CoreNLP output with the semantic resources of OntoSem required significant effort.[2]

Before importing resources, it is necessary to assess how useful they will be for a system. A large number of resources have been created over the past 40 years, too many to mention in this paper. Centrally, they include formal descriptions of the world (starting with CYC and proliferating to many other formal world models) and resources developed as training data for a variety of application of machine learning techniques (e.g., language-oriented resources such as machine-readable dictionaries and annotated text corpora). Not all of these resources are of direct utility to cognitive systems. Recommendations concerning the importation of knowledge resources must still be developed.

Of course, importing resources is not the only path forward. Developing knowledge resources expressly for cognitive systems is a valid and attractive option for CCM. It is attractive conceptually since it most seamlessly contributes to theoretical and descriptive work on CAs and their components. It is also attractive in practical terms – not least because it will obviate the need to patch over the insufficiencies of imported resources. A third option for acquiring resources is to develop methodologies for combining externally and internally developed resources in ways that assure their smooth interoperability.

An alternative to pursuing broad-scale knowledge coverage from the outset is to envisage the gradual acquisition of knowledge over consecutive versions of an application system. However, this approach turns out to be impractical because practice shows that solutions that cover only limited examples do not readily expand in scope; instead, they are thrown away every time more examples need to be covered. For example, disregarding (or drastically reducing the prevalence of) lexical ambiguity in language inputs – which is almost universally practiced in current cognitive system applications – obviates the huge requirement for a realistic-scale language understander.[3] As a result, such systems appear to perform at a far more sophisticated level than could be achieved without such simplifications. It is more efficient – in fact, imperative – to address cognitive phenomena more holistically from the outset, using explanatory microtheories.

A key to solving the knowledge problem in cognitive systems is enabling intelligent agents to engage in human-like lifelong learning, which combines language understanding with bootstrapping from existing knowledge bases. However, as early experimentation showed, both the knowledge bases and the language understanding capabilities must first reach a critical level of quality and coverage to support useful learning in this style (Nirenburg et al., 2007).

The above content-oriented tasks are not a focus of CA research. CAs are responsible for the modularization and organization of component processes. They propose a typology and requirements for knowledge components, including heuristics. They also suggest formalisms and processing approaches to be used. Application systems implement these recommendations. The more

[2] This effort was further exacerbated by the frequency of errors (which, for a statistically-trained system, cannot be remedied), and the fact that syntactic parsers are designed to make decisions that, in reality, need to be postponed until semantic analysis can weigh in.

[3] In fact, the field of natural language processing has experimented with many types of simplifications over the years, such as forcing people to use controlled languages, and pre-editing text as input to machine translation. None of these methods has proven successful, not for lack of trying.

theoretically-motivated application systems attempt to carry out as little "tailoring" (to use Forbus's term (Forbus, 2018: p. 21)) beyond the tenets of a CA as possible, while still striving to meet application requirements.

What remains an orphan – undertaken by neither CAs nor application systems – is work on guaranteeing the timely availability and broad coverage of knowledge content, including models of the world, language, and the agent itself (notably, the agent's decision heuristics).

Large-scale and diverse content is a core prerequisite for making cognitive systems truly human-level, as well as for achieving meaningful validation of the theoretical statements made in CA research. Once the acquisition of content is decoupled from the immediate needs of a particular application system, our systems will be able to implement not just predominantly reactive or preprogrammed agents but agents that will be able to expand their capabilities by using what they know to expand their know-how. If our community hopes to develop cognitive systems that emulate the human ability to interpret perception signals, make decisions, and act, then work on content must become more central to the entire enterprise. Statistical and deep learning approaches can provide some grist for the heuristic mill of cognitive systems. But by themselves they do not hold the promise for fulfilling this need (it is outside the scope of this paper to argue why this is the case – see argumentation in Marcus (2020)). If it is neither for CAs nor for application systems to address the issue, it follows that the community must promote work on developing content-centric models (CCMs)[4] to bridge this gap.

## 3. Objectives of Content-Centric Modeling

The objectives of content-centric modeling (CCM), which centrally address concerns related to feasibility and utility, include:

A. describing natural phenomena of the world and language (including how they are perceived by agents) in terms of ontological properties and their value sets
B. defining types of objects, events, and the relations among them, in terms of the above properties
C. determining how descriptions of *instances* of objects and events are related to descriptions of their *types*
D. defining processing paradigms that reconcile efficiency requirements with the necessity of using non-toy amounts of diverse knowledge
E. ensuring that modeling supports the specific content needs of different modes of reasoning (cf. the modes of reasoning discussed in Section 5 of Davis & Marcus (2015))
F. ensuring that the content required to support all the components of a cognitive system uses an interoperable metalanguage anchored in a common world model
G. using the above conceptual infrastructure for ongoing application-oriented knowledge acquisition
H. developing methods for automating content acquisition, based either on processing perceptual inputs using available knowledge resources or on reasoning over available knowledge
I. developing an efficient methodology of converting inputs and outputs of application system modules to and from the interoperable format supporting CCM
J. developing methods of assessing the quality of acquired content
K. developing the infrastructure to store, maintain and augment the above knowledge

[4] We distinguish between CCM (uncountable) as a methodology and CCMs (countable) as concrete instances of such modeling.

L. supporting the integration of CCMs of individual phenomena into an overall CCM of human cognitive functioning (a CCM can, in fact, be a society of CCMs)
M. combining elements of knowledge used by and generated by different architecture modules in heuristic bundles to support particular reasoning and decision-making processes.

Some of the above objectives (for example, Objectives G and H) are more difficult to achieve than others. Many of them can be pursued either as part of work on a particular end application or with the goal of developing generic resources and capabilities to serve a variety of end systems. No matter the emphasis of CCM-oriented work, its utility will be in a large part assessed by how easily the CCM can be adapted to new applications. In what follows we illustrate our views on the nature and utility of CCMs using examples from our team's research.

## 4. Size, Heterogeneity and Detail of Knowledge

To reiterate, Lieto et al. underscore that the small size and lack of heterogeneity of knowledge in current CAs will ultimately impede progress on cognitive systems research. However, not all work on cognitive systems falls under the CA paradigm that Lieto et al. critique. In fact, our OntoAgent research framework already features a large *amount* of many *kinds* of knowledge. The knowledge is organized into CCMs (called *microtheories*) and is recorded in several knowledge resources – most notably, the ontology, the semantic lexicon, the long-term episodic memory of object and event instances and the agent model. To illustrate this generalization, Table 1 lists some of the stored knowledge needed to support the task of language understanding – which is just one component of the intelligent agent's perception apparatus in OntoAgent.

**Table 1**. Static (prerecorded) knowledge used by the consecutive stages of language understanding in OntoAgent. Abbreviations indicate whether the knowledge is stored in the ontology (O), the lexicon (L), dedicated rule sets supporting language understanding (R), knowledge components unrelated to language processing (K), or in resources aimed at supporting machine learning (ML).

| Stage of Language Analysis | Kinds of Static Knowledge It Uses |
|---|---|
| Pre-Semantic Analysis | - morphological and syntactic grammars (R)<br>- gazetteers or ML-based named entity recognizers (R, ML)<br>- lemmas from the lexicon (L)<br>- annotated corpora (ML) |
| Pre-Semantic Integration | - pre-semantic analysis error recovery rules (R)<br>- reambiguation rules for syntactic "decisions" that require semantic analysis[5] (R)<br>- rules for linking lexicon senses to words of input (R)<br>- rules for assessing the confidence of input-to-lexicon linking (R) |
| Basic Semantic Analysis | - the semantics of the lexical senses of words and phrases (L)<br>- syntax-to-semantics linking rules derived from lexicon entries (L)<br>- procedural semantic routines[6] (L)<br>- ontological constraints to gauge the confidence of semantic dependencies (O) |

[5] Examples include prepositional-phrase attachments, distinguishing prepositions from particles, and the bracketing of nominal compounds comprised of three or more nouns.
[6] For example, *very* in *very smart* must dynamically increase the scalar value of INTELLIGENCE used to describe *smart*.

| | |
|---|---|
| | - microtheories of phenomena (modality, aspect, mood, time, etc.)[7] (L, O, R) |
| Basic Coreference Resolution | - select property values of candidate sponsors (O) (see discussion in Section 4.3)<br>- microtheories for the treatment of specific types of referring expressions: personal pronouns, definite descriptions, etc. (R) |
| Extended Semantic Analysis | - object-to-object relations for extra-clausal disambiguation (O)<br>- microtheories of non-literal language processing (e.g., metaphor) (L, O, R)<br>- repository of compounding patterns (O, R)<br>- repository of classes of metonymies (O, R)<br>- microtheory of extra-clausal disambiguation (O, R)<br>- idiomatic creativity rule set (R) |
| Situational Reasoning | - goal and plan inventory (K)<br>- agenda of active goals and plans (K)<br>- models of self and other agents (K)<br>- microtheories of "mindreading", indirect speech acts, ellipsis, fragments (R, K)<br>- results of other non-linguistic perception interpretation services (K)<br>- active concept instances in the situation model (K)<br>- the microtheory of reference (i.e., anchoring referents to agent memory) (K) |

In addition to this great diversity of prerecorded knowledge, each of the analysis stages also uses results generated by prior stages of processing. Table 2 lists some of the kinds of knowledge that each stage *produces*.

**Table 2.** Knowledge resources dynamically generated during the processing of language input by various stages of language understanding in OntoAgent. Entries in *italics* mark results that may remain underspecified until a subsequent stage. For a detailed discussion of the stages of language interpretation, see Chapters 3-7 of McShane and Nirenburg (in press).

| **Stage of Language Analysis** | **Knowledge Resources It Produces** |
|---|---|
| Pre-Semantic Analysis | - sentence boundaries<br>- word information: lemmas, part of speech tags, morphological features<br>- *syntactic parses (constituency and dependency)*<br>- named entities |
| Pre-Semantic Integration | - linkings between elements of input and OntoSem lexicon senses<br>- *lexicon entries for unknown words (with underspecified semantics)* |
| Basic Semantic Analysis | - *word-sense disambiguation decisions*<br>- *semantic dependency decisions*<br>- *the identification of the availability of both direct and indirect speech-act interpretations*<br>- *a more precise specification of the semantics of unknown words* |
| Basic Coreference Resolution | - *k-best candidate sponsors for textual referring expressions*[8]<br>- the sponsor for many elided referring expressions (objects and events)<br>- the understanding that a given referring expression does not require a sponsor<br>- bridging reference interpretations for definite descriptions |
| Extended | - the semantic/pragmatic incorporation of fragments into the discourse context |

[7] Knowledge to support microtheory implementation is distributed between the ontology, the lexicon, and dedicated rule sets.

[8] Personal pronouns, definite descriptions, proper nouns, demonstrative pronouns.

| Semantic Analysis | - *a culled set of candidate text meaning representations* |
|---|---|
| Situational Reasoning | - the intended meaning of the input[9]<br>- true reference resolution[10]<br>- improved interpretation of unknown words<br>- any residual ambiguities (for optional clarification) |

Even a casual perusal of the content of these tables shows how hopeless the now-or-never approach is when the goal is to produce content that is sufficient to support human-level reasoning and decision-making.

This paper does not attempt to discuss the particulars of the actual knowledge content of OntoAgent. Much of that – particularly as regards language understanding – is described in a forthcoming book (McShane and Nirenburg, in press). In addition to presenting the current state of many microtheories, the book integrates our work into the many related literatures (to the tune of nearly 400 references); it describes our considerable ongoing efforts to evaluate the content of knowledge-based systems; and it offers a roadmap for future R&D on the large number of foundational issues needed to achieve human-level agent functioning.

### 4.1 Size Is a Tricky Measure

Lieto et al. state that "…the size problem is intuitively easy to understand (i.e., it concerns the dimension of the knowledge base available to the agents)." Broadly speaking, this is a true statement but it can lead to misunderstanding. That's because pure numbers do not reflect the utility of a resource. The latest version of the OntoAgent ontological world model, used by every processing model, covers about 9,000 mostly general-domain concepts but, counted differently, records over 165,000 RDF triples. The latest version of the OntoAgent lexicon for English, used to support language understanding and generation, includes about 30,000 word senses. But this number also hides a lot of additional complexity. For example, the semantic descriptions of most word (and phrase) senses make reference to ontological concepts. But in many cases a simple pointer to an ontological concept is not sufficient; instead, certain property values of the underlying ontological concepts must be locally modified inside the lexicon entries. This enriches the expressive power of text meaning representations without cluttering the ontological model with excessive (and often language-dependent) differences among concepts. Many lexicon entries also contain procedural attachments that contain knowledge about how to determine specific meanings in context – e.g., how to combine the meaning of *very* with scalar attributes like *tall* and *cold.*

In short, simply counting the number of entries in a lexicon, or the number of concepts in an ontology, is of little use in assessing the coverage provided by a knowledge resource. Breadth of coverage needs to be considered in combination with the depth of description of each knowledge unit. So, while it is not appropriate to make any conclusions about the natural language capabilities of a cognitive system that operates with a lexicon of a few dozen entries, it is equally inappropriate to assume size-related benefits if a system incorporates, say, the six million facts from the shallow ontology of DBpedia (Bizer et al., 2009). CCM jointly addresses the breadth and depth of content in cognitive systems.

[9] This includes understanding implicatures, speech-act interpretation, etc.

[10] This is not textual coreference; it involves grounding all referring expressions in the agent's long-term memory.

### 4.2 Heterogeneity

In their critique of CAs, Lieto et al. emphasize that a system's world model must include both prototypes and exemplars. OntoAgent already incorporates these types of knowledge. Prototypes are encoded in the OntoAgent ontology. For example, the value of the AGENT property of the event MILITARY-OPERATION is specified as any descendant of the concept MILITARY-ROLE on the property's *default* facet, any descendant of HUMAN on its *sem* facet and any descendant of GEOPOLITICAL-ENTITY on its *relaxable-to* facet. This, among other things, allows the language understander to prefer to interpret *The general started the operation* as referring to MILITARY-EVENT and not SURGERY. If the input were *The chief started the operation,* the choice between the two kinds of events would require additional knowledge, because the lexical sense of *chief* is a descendant not of MILITARY-ROLE but of HUMAN, which is the filler of the *sem* facet of the AGENT property of *both* MILITARY-EVENT and SURGERY.

Exemplars are realized as instances of ontological concepts stored in the agent's episodic memory (also referred to in the OntoAgent literature as the *fact repository* or *belief repository*); as such, they are ontologically typed. This means that constraint satisfaction will be able to use static ontological knowledge in addition to the information available in the exemplar. Thus, if Napoleon Bonaparte is listed in an agent's fact repository as an instance of both HEAD-OF-STATE and GENERAL (a descendant of MILITARY-ROLE), *Napoleon started an operation* will be interpreted as reporting about an instance of MILITARY-EVENT on account of the exemplar being grounded in the ontology. Of course, if a particular person is remembered as being both a military officer and a physician, the facet-level comparison illustrated above will not resolve the ambiguity, and further analysis will be required – if the agent deems the resolution of this ambiguity essential.

### 4.3 Depth of Description

OntoAgent goes beyond accommodating prototypes and exemplars. Let us give some more illustrations of the detail of knowledge in the environment. OntoAgent's ontology accommodates detailed descriptions of complex events – also called scripts. For example, Figure 1 presents a small excerpt from a comprehensive ontological complex event (script) detailing the progression of the gastroesophageal reflux (GERD) disease.

This script was developed for OntoAgents in the Maryland Virtual Patient (MVP) system for training clinicians (McShane & Nirenburg, in press). The excerpt is shown in its original Lisp format and illustrates time management, cause-effect relationships, checking and updating property values, triggering new events (scripts), and stopping the operation of scripts in progress. From the CCM perspective, it is important to stress that the script was viewed as part of the system's knowledge content. And, in fact, two different decision-making methods (rule-based and Bayesian) were implemented to run it using the same basic knowledge infrastructure.

The script from which this excerpt was drawn supports a dynamic simulation process that models human physiology. Our choice of how to implement this simulation carried methodological implications. Physiology is not cognitively controlled (though psychosomatic effects are real enough, and we did model them in MVP). So, in principle, we could have developed or imported (had it existed) a statistically-based, non-explanatory physiological simulation and then built an interpreter to convert its output (such as perceived symptoms) into the uniform OntoAgent metalanguage. This would have made this task similar to interpreting outputs from other perception modalities, such as vision or language. However, we decided, instead, to develop the simulation in the OntoAgent metalanguage from the start, both because having an explanatory model better fits the pedagogical

needs of the MVP application system, and because it eliminates the need for an extra interpreter in the overall architecture.[11]

```
(gerd
<…>
(tracks
<…>
  (if (> total-time-in-acid-reflux 1.2)
    then
    (bind-variables
      (fraction-through (- 1.0 (/ (- time-since-start time-increment) gerd-time)))
      (gerd-t-duration-days (get-attribute gerd-t-duration gerd-patient 1000))
      (extra-time (fraction-through gerd-t-duration-days 60 60 24)))
    (effect
      (unassert-background-script heal-preclinical-gerd $var0)
      (assert-background-script preclinical-gerd $var0
        ((extra-time (+ extra-time time-increment))))))
  (if (and   (>= time-since-start gerd-time)
        (<= total-time-in-acid-reflux 1.2))
    then
    (effect
      (unassert-background-script heal-preclinical-gerd $var0))))
<…>)
```

Figure 1. A small excerpt from an OntoAgent script supporting simulation of the physiology of an OntoAgent implemented as a virtual patient in a training application for physicians.

Separately from the ontological model, an OntoAgent agent is also endowed with a model of agency and its instances – notably, the agent's model of itself and its models of other agents it knows. The model of agency is a complex CCM that defines the agents' goal and plan inventories,[12] models of their personality traits, biases, emotions and physical states. An agent typically has several instances of this model in its long-term memory, one for itself and one for each of the other agents it knows.[13] Agents equipped with models of other agents are thus endowed with a theory of mind, which allows them to "mindread" other agents.

To illustrate the operation of an agent equipped with models of self and others, consider the following example from the MVP application, which highlights how agents operating with different content generated different behavior while playing the role of a (virtual) patient in a conversation with a (human) clinician during a simulated office visit. The physician asks the question "Have you traveled recently?" A content-poor virtual patient (VP), Tom Dumb, understands the meaning of this input roughly as, "Does your long-term episodic memory contain recent instances of TRAVEL-EVENT with you as the AGENT?". The VP consults its episodic memory, finds just one instance (TRAVEL-EVENT-37) with an appropriate time stamp, and generates a response directly on its basis:

[11] Note also that the perception of physiology is different from, say, visual perception: it is "private" (hidden from the observer, like the agent's reasoning processes) and must therefore be treated in a phenomenological, first-person perspective in applications such as social robotics.

[12] Since OntoAgent agents attempt to model humans as closely as possible, they follow the omnipresent principle of least effort and avoid, if at all possible, planning from first principles, preferring to adapt and use plans already recorded in their memory.

[13] The fact that the agent model of self may differ from what an omniscient observer knows about it provides fertile ground for experimentation in application systems that investigate decision-making on the basis of biased premises.

"Yes, I went to Philly by car last Tuesday." Clearly, this is not what the physician had in mind. By contrast, a content-rich VP, Tom Smart, has several relevant scripts in its ontology (DOCTOR-VISIT, TRAVEL-EVENT, INFECTIOUS-DISEASE) along with a model of the clinician agent that enables mindreading. These resources allow it to carry out a more complex and human-like sequence of operations: a) establish the intended meaning of the question ("Could travel have caused or contributed to your condition?"); b) recognize that the physician's goal is to establish a diagnosis; c) instantiate a (sub)goal of understanding whether the question is part of a plan toward that goal (rather than, say, small talk, which pursues a different goal); d) conclude that the question is, in fact, part of the clinician's diagnosis script, since some kinds of TRAVEL-EVENTs are potential PRECONDITIONs of the event INFECTIOUS-DISEASE; e) check its episodic long-term memory to see whether it contains any instance of such a TRAVEL-EVENT; f) establish that this is not the case; and g) generate a response based on the intended meaning of the question: "No, I haven't been anywhere that might have made me sick." The reasoning parts of the above processing sequence involve accessing diverse elements of the agent's memory, which underscores the benefits of the CCM methodology. Although the above example was developed as a proof of concept, our current work (within the applications of furniture building and driving) is attempting to generalize about such reasoning in service of dialog-based collaboration and learning through a combination of dialog and experience.

## 5. Insights from Language Understanding

Our experience in working on knowledge-based language understanding has motivated us to adopt the following tenet as the core principle of CCM: *Decision heuristics within any perception or reasoning module may require knowledge from any of the system's static knowledge resources and/or any of the dynamic knowledge generated by any of the system's processing modules*.

The opinion that knowledge about the world is required for understanding language is, of course, not new. CCM goes further by suggesting that language understanding and the general reasoning it supports share the same knowledge substrate. Validating this hypothesis has been an important part of our work over the years. This has involved inventing the ever-evolving theory of ontological semantics; developing non-toy, heuristics-oriented knowledge bases and processing models; and configuring several proof-of-concept computational systems that demonstrated the validity of the hypothesis.

One direction of our work involved organizing the language understanding module in stages (see the left-hand columns of Tables 1 and 2) that rely on particular sets of knowledge resources. One reason for this was to allow the system to produce *actionable* interpretations (that is, ones sufficient to support general reasoning and decision-making) as soon and as cheaply as possible, without always applying the entire battery of available language-oriented reasoning. To take a simple example, given the input *Alice ate an apple,* there is no reason for the agent to invoke the processors that seek out indirect speech acts, metaphorical interpretations, or implicatures. The reason is that the knowledge about eating recorded in the OntoAgent lexicon and ontology, when leveraged by the semantic analyzer in OntoAgent, leads to a single, high-scoring interpretation of this input:

```
INGEST-1
   AGENT          HUMAN-1
   THEME          APPLE-1
   TIME           < find-anchor-time      ; indicates 'before the time of speech'
HUMAN-1
```

HAS-PERSONAL-NAME 'Alice'
AGENT-OF INGEST-1
APPLE-1
THEME-OF INGEST-1

Since all of the recorded syntactic and semantic constraints are fulfilled by this interpretation, there is no flag for the agent to dig deeper. Examples of common flags for digging deeper are:

- incongruities, such as *Alice reprimanded the apple*; there is no sense of *reprimand* whose direct object can mean APPLE, so the sentence must feature either a metaphor, a metonymy, or an as-yet unrecorded meaning of *Alice, reprimand* or *apple*
- residual ambiguities, such as *Turn the steak;* does this mean flip it or rotate it?
- utterances that may or may not be indirect speech acts, such as *I'd love something to eat right now*; should the listener do something about that?

In short, the agent will typically follow the principle of least effort just like a person would. Of course, in some situations, people can go beyond their original, actionable (i.e., satisficing) interpretation by adding inferences and corollaries. In fact, for a long time, the ability to derive a "complete" set of meanings, inferences and corollaries from an input was considered the ultimate objective for language-endowed intelligent agents. However, we believe that the incrementality-actionability hypothesis – which, itself, is a corollary of viewing language processing as a part of comprehensive cognitive functioning – seems to provide both a more realistic measure of success as well as a more realistic path of advancement. As part of our future work we intend to apply actionability judgments to modules of a CA beyond language understanding, such as general reasoning and decision-making.

## 6. Efficiency

Efficiency is a major concern in implementing cognitive system applications. Indeed, today's cognitive robotics systems rely on small-scale content not only because large-scale knowledge is unavailable but also because processing large amounts of knowledge and addressing large numbers of phenomena takes too much time. To be considered realistic, a robot must respond to a user's question within at most one second, preferably less. But at the same time, its response must make sense, and this, outside of very limited domains, requires knowledge-based processing. How does one reconcile the ostensibly contradictory requirements for fast processing and utilizing large-scale content? We believe that the incrementality-cum-actionability apparatus will make CCM a practical alternative. By offering ways to cut the amount of knowledge-based processing it will alleviate the need to impose *a priori* restrictions on the breadth and depth of content used by the system.

An alternative approach to maintaining efficiency involves making modules of a system and the knowledge they use as autonomous as possible. It offers two benefits: first, it helps to circumscribe the requirements for the inputs and outputs of system modules; and second, it streamlines control – typically, through a pipeline control architecture. This approach disregards the (ubiquitous) eventuality that local decisions can be incomplete, underspecified, or vague before a sufficient amount of knowledge has been brought to bear – which often requires crossing the boundaries of system modules. Justifying this approach, at least in part, by appealing to the now-or-never principle does not work either: Not only is the now-or-never interpretation of incrementality insufficient, it is

unnecessary if one adopts the CCM perspective based on actionability, along with a more sophisticated approach to incrementality. In English & Nirenburg (submitted) we illustrate how the ability to integrate inputs from multiple knowledge sources facilitates modeling the attention and reasoning components of OntoAgent, while better reflecting human behavior overall.

## 7. Heuristic Search

Heuristic search involves planning from first principles. Since it is the method at the core of classical planning, it is a defining feature not only of cognitive systems but of AI in general. In order to position CCM within work on cognitive systems overall, we must assess how, and to what degree, CCM employs heuristic search.

CCM aims to model the everyday functioning of humans. Although people are certainly capable of heuristic search, they seem to use it in quite limited contexts (outside of game playing).[14] Instead, they prefer to recall remembered types or instances of ready-made, habitual solutions and, if necessary, adapt them to the situation at hand (e.g., through acceptable levels of partial matching). The fact that people do not regularly operate from first principles explains the popularity of stories about human resilience and success in the face of disaster. The rare and novel situations in *Robinson Crusoe*, Jules Verne's *The Mysterious Island* or any number of disaster movies describe how the protagonists are forced to think, plan, and operate from first principles because their stored knowledge does not suffice. But even then, some remembered first principles play a role – knowing how lenses can be used to start a fire in the sun allowed the modern-day Robinson Alvaro Cerezo to use a plastic bag filled with sea water for this purpose.[15]

For a more formal assessment of the role of search in CCM, Langley's (2018) taxonomy of types of heuristic search has proven useful. Langley succinctly reformulates Newell and Simon's (1976) heuristic search hypothesis, according to which problem solvers rely on four main factors – indicated in boldface in the discussion below.

The space of **candidate solutions** in CCM consists of instances of world states as interpreted by the agent using its perception interpretation apparatus operating over its stored knowledge. For example, goal instances are selected from the goal inventory, and candidate meaning representations of utterances are composed from the stored meanings of the words of input guided by the stored rules of meaning composition. CCM "prepackages" knowledge and organizes it in a way that limits the number of tasks that require search and minimizes the size of the search spaces for the rest.

The **generation of new candidates during system operation** occurs when so-called "unexpected inputs" are encountered. For example, the agent might be unable to interpret a component of perceptual input, or it might fail to find an appropriate goal or plan to pursue in a particular situation. Unexpected input is addressed in CCM using recovery procedures (those related to language processing are detailed in McShane and Nirenburg, in press). Some involve making do with the available knowledge, while others involve learning on the fly – primarily through language, just as people do. In the long run, its emphasis on lifelong learning through language makes CCM a

[14] People can use heuristic search to look for a misplaced cellphone, and robots can use it to navigate to a particular room in an office building (Barrett et al. 2018); but even then it is the quality of heuristics that is crucial for overall efficiency.

[15] https://www.mirror.co.uk/news/weird-news/real-life-robinson-crusoe-reveals-11976781

favorite in the race toward human-level cognitive systems. The overall CCM emphasis is on continuous enhancement and improvement of the breadth and depth of content coverage both during task execution and at other times.

**Heuristic decision functions** and **test criteria** are core components of the agent's long-term memory (LTM). Heuristic decision functions help select an instance of a known type of solution (e.g., a recorded plan, or the recorded semantic interpretation of an input word). Parameter values for heuristic functions come both from the agent's situation model and from different components of its LTM. Once the heuristic function has made its decision, test criteria are applied. "…to determine if a candidate is an acceptable solution" (Langley, 2018). The main task for test criteria in an application system is to assure that various system inputs – and generally, the content of the agent's situation model – intelligently and realistically match the relevant elements of the system's background knowledge stored in LTM. This requires additional layers of knowledge, such as various levels of prototyping, the fine-grained description of ontological property values (for example, using the facet mechanism in OntoSem), appropriate penalties for inexact matches on particular property values, and so on. So, in our view, test criteria are part of content that describes the ways of applying and assessing heuristic decision functions, and the role of search in their operation is peripheral.

If the system performs as expected, the results of an application of a heuristic decision function will satisfy the test criteria, and no additional processing will be required. Sometimes, however, the values of arguments of heuristic decision functions are unavailable. In such cases, in order to test a solution, the system must first carry out additional processing, such as instantiating a (sub)goal of obtaining the missing value, selecting a plan for doing so, and executing it. CCM minimizes the need for search even in this case. A CCM stores knowledge about the relative significance of the contribution of a particular argument to the result of a decision function. This facilitates – without the need to be concerned about the efficiency of search – ruling on whether a decision will be acceptable without the contribution of the missing parameter value. Even when subgoaling is required, CCM will guarantee the availability of plans to attain the subgoals, so that such plans will not have to be created on the fly. If such plans fail, then the agent will immediately enter the recovery mode and apply stored procedures for adjusting the confidence values of its results. In any case, this processing does not involve the scope of search that would trigger efficiency concerns.

To summarize, CCM strives to substitute retrieval from structured knowledge stores for heuristic search. That is, it underscores the centrality of heuristics and test criteria and downplays the dynamic generation of search spaces and search itself. Making retrieval and storage operations efficient is one of the underlying concerns of implementing CCMs. The need for deep and broad knowledge coverage can thus be viewed as a corollary of the need to support the formulation and efficient application of heuristic decision functions and test criteria. This, in turn, motivates the preference for an interoperable knowledge representation substrate across all the in-house static and dynamic knowledge resources.

## 8. Final Thoughts

In making a case for the centrality of content-related issues for the long-term goals of cognitive system research, we have covered a lot of territory. We have described the nature and scope of content-centric cognitive modeling (CCM), juxtaposing it with work on cognitive architectures, on the one hand, and work on cognitive systems, on the other. We have explained the importance of addressing long-term knowledge needs immediately, since they cannot be met by gradual enhancements to the knowledge substrate of application systems. We have discussed various methodologies

of acquiring knowledge for use in cognitive systems, acknowledging both their inevitable cost and their individual pros and cons. We have described ways in which a content-centric perspective affects design preferences for cognitive architectures (recall the explanation of why the now-or-never approach to language understanding will not work outside of toy domains). We have explained why heuristic search has decreased prominence in CCM, being largely replaced by retrieval from structured knowledge sources. And we have provided as much evidence as the space allows to show that work on CCM is well underway in the OntoAgent environment.

We believe that it is for pressing, practical reasons that the cognitive systems community widely adopts a R&D methodology centered around demonstration systems. Although this is a venerable approach to testing the maturity of a technology, its very nature biases against long-term objectives. CCM offers a different and complementary path from theories (embodied in cognitive architectures) to application systems. It facilitates interleaving the task-oriented operation of an agent with the continuous accumulation of a variety of kinds of descriptive and decision-supporting knowledge ("lifelong learning") – all while still supporting the development of proof-of-concept systems. The long-term goal of CCM is to support shifting from the currently unavoidable hand-crafting of knowledge to automatic learning through language, reasoning, and exploration.[16] A more immediate goal is to reach the critical mass and depth of knowledge that will facilitate lifelong learning in non-toy domains. We believe that we are on the cusp of this opportunity today.

Additionally, CCM supports reusing knowledge across the set of application systems developed by a research team. In other words, in this paradigm, knowledge that supports a demonstration system is not thrown away. It is added to the cumulative knowledge in the underlying CCM model. As a result, the developers of later demonstration systems will have an ever growing chance for avoiding creating knowledge from scratch. Methodologically, striving for the continuing utility of CCMs is roughly similar to the strategy adopted for the GATE environment for natural language engineering (e.g., Cunningham, Tablan, Roberts, and Bontcheva 2013). This strategy underscores the importance for CCM of attention to infrastructure development – not only ergonomic tools for system developers and users but also the support for efficient knowledge management (storage, update and retrieval). Our latest knowledge management environment, developed specifically to support CCM, is described in English and Nirenburg (submitted).

To sum up, we believe that the CCM perspective is essential for the long-term success of cognitive systems research. CCM complements other approaches by offering reusable content, both for a single team and across research teams (though in the latter case knowledge formats will have to be adjusted). It also offers the best hope for developing agents that can engage in lifelong learning and, therefore, independently address the cost of knowledge acquisition for cognitive systems.

## Acknowledgements

This research was supported in part by Grants N00014-16-1-2118 and N00014-17-1-2218 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. Many thanks also to the anonymous reviewers whose comments and suggestions have been welcome.

## References

[16] Manual knowledge acquisition in support of CCM differs from knowledge acquisition within approaches that do not adopt an agent perspective. Lack of space does not allow for a detailed analysis of these differences. We intend to address them in future work.

Barrett, D.P., Bronikowski, S.A, Yu, H, & Siskind, J.M. (2018). Driving under the influence (of language). *IEEE Transactions on Neural Networks and Learning Systems, 29*:2668-2683.

Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak R., & Hellmann, S. (2009). DBPedia – A crystallization point for the web of data. *Journal of Web Semantics, 7*:154-165.

Christiansen, M.H. & Chater, N. (2016). The now-or-never bottleneck: A fundamental constraint on language. Behavioral and Brain Sciences, *39*: 1-72.

Cunningham, H., Tablan, V., Roberts, A. & Bontcheva, K. (2013). Getting more out of biomedical documents with GATE's full lifecycle open source text analytics. PLoS Comput Biol 9(2): e1002854.

Davis, E. & Marcus, G. (2015). Commonsense reasoning and commonsense knowledge in artificial intelligence. *Communications of the ACM*, 58:9, 92-103.

English, J. and Nirenburg, S. (submitted). OntoAgent: Implementing content-centric cognitive models. Submitted to Advances in Cognitive Systems 2020.

Forbus, K. D. (2018). *Qualitative Representations*. MIT Press.

Healey, P. G. T., Howes, C. Hough, J., & Purver, M. (2016). Better late than Now-or-Never: The case of interactive repair phenomena. In Christiansen and Chater (2016).

Jacobsson, H., Kruijff, G-J., Hawes, N. & Wyatt, J. 2007. Crossmodal content binding in information-processing architectures. *Proceedings of HRI-08*. Amsterdam.

Langley, P., Laird, J. E., & Rogers, S. (2009). Cognitive architectures: Research issues and challenges. *Cognitive Systems Research, 10:*141-160.

Langley, P. (2018). Theories and models in cognitive systems research. *Advances in Cognitive Systems*, *6*:3-16.

Levinson, S. C. (2016). "Process and perish" or multiple buffers with push-down stacks? In Christiansen and Chater (2016).

Lewis, M. L., & Frank, M. C. (2016). Linguistic structure emerges through the interaction of memory constraints and communicative pressures. In Christiansen and Chater (2016).

Lieto, A., Lebiere, C., & Oltramari, A. (2018). The knowledge level in cognitive architectures: Current limitations and possible developments. *Cognitive Systems Research, 48*:39-55.

MacDonald, M. C. (2016). Memory limitations and chunking are variable and cannot explain language structure. In Christiansen and Chater (2016).

Mahesh, K., Nirenburg, S. and Beale, S. (1996). Knowledge representation requirements for natural language semantics: A critical evaluation of Cyc. *Proceedings of KR-96*.

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.

Marcus, G. (2020). The next decade in AI: Four steps towards robust artificial intelligence. arXiv: 2002.06177.

McDonald, David D. (1983). Description directed control: Its implications for natural language generation. In Brady, N. (editor), *Computational Linguistics*. Pergamon Press. 111-129.

McShane, M., & Nirenburg, S. (in press). *Linguistics for the Age of AI*. MIT Press.

Newell, A. & Simon, H. A. (1976). Computer science as empirical enquiry: symbols and search. *Communications of the ACM, 19*:113-126.

Nirenburg, S., T. Oates and J. English. 2007. Learning by Reading by Learning to Read. *Proceedings of the International Conference on Semantic Computing*. San Jose, August.

Scheutz, M., Krause, E., Oosterveld, B., Frasca, T., & Platt, R. (2017). Spoken instruction-based one-shot object and action learning in a cognitive robotic architecture. *Proceedings of the Sixteenth International Conference on Autonomous Agents and Multiagent Systems*

Sloman, A., & Scheutz, M. (2002). A Framework for comparing agent architectures. *Proceedings of URCI'02*.