Proceedings of the Second Annual Conference on Advances in Cognitive Systems ACS-2013 (135-152)

On the Representation of Inferences and their Lexicalization

David McDonald DMCDONALD@SIFT.NET
Smart Information Flow Technologies, 14 Brantwood Road, Arlington, MA 02476 USA

James Pustejovsky JAMESP@CS.BRANDEIS.EDU
Department of Computer Science, Brandeis University, 415 South Street, Waltham, MA 02453 USA

Abstract

We have recently begun a project to develop a more effective and efficient way to marshal inferences from background knowledge to facilitate deep natural language understanding. The meaning of a word is taken to be the entities, predications, presuppositions, and potential inferences that it adds to an ongoing situation. As words compose, the minimal model in the situation evolves to limit and direct inference. At this point we have developed our computational architecture and implemented it on real text. Our focus has been on proving the feasibility of our design.

1. Introduction

1.1 Gaps

One of the central facts about language is that speakers regularly omit information that their listeners fill in without conscious effort. — they leave *gaps*. Examples of such gaps are everywhere. Consider the following text.[1]

> "*. . . a 14-year-old girl died in the Kurdish city of Sulaimaniya . . . The rest of the family is in good health . . .* "

We effortlessly know that this is the family of the girl, even across the three intervening sentences in the full text. The writer could have said "*the girl's family*" but didn't have to, knowing that readers would supply this information though inference.

Gaps like these are a pervasive and essential component of language use: speakers appreciate what their listeners will infer from their knowledge of the world (e.g., children are presumed to have families) and from the context that they share. This is one of the points of Grice's Maxim of Quality: do not be more informative than required (Grice, 1975). It is central part of what makes a text cohesive.

The question is how this is done. How is our extensive body of background knowledge and inference organized? How do we deploy it so effortlessly? That is the subject of this paper, where we lay out our initial results from a recently initiated project into this question.

1. This passage is excerpted from a January 18th, 2006 Aljazeera news article about the first bird flu victim in Iraq.

1.2 Speed implies structure

Psycholinguists have known for decades that language comprehension is immediate and incremental and works on all levels at once (syntactic, semantic, pragmatic, discourse) (Marslen-Wilson, 1973). People interpret utterances word by word without noticeable delay. Recent work has shown that an event verb will activate its prototypical objects in just the time it takes to hear the verb and that this will influence the interpretation of later syntactic structures (Matsuki et al., 2011).

When cognitive psychologists explain this ability, they talk about people having *schemas* that organize their knowledge of ordinary things and events (Bartlett, 1932). This resonates with the ideas and mechanisms of frames and scripts that were developed in Artificial Intelligence more than thirty years ago (Minsky, 1975; Schank & Abelson, 1977). But in areas of research such as neuroscience (Speer et al., 2009), or cognitive linguistics (Bergen, Chang, & Narayan, 2004), what a schema consists of or what it means, computationally, to 'activate' a schema and 'provide' expectations has different answers — it is usually not the point of their research. It is, however, the point of our own research. This paper describes our computational account of what a schema is, how they are activated, their mechanisms for controlling interpretation, and how they provide expectations, implicatures, and defaults.

1.3 It takes knowledge

The knowledge-rich approaches of the 1970s and 1980s were abandoned by main-stream NLP research as part of the move to 'empirical' approaches that were made possible by the construction of large machine-readable text corpora and advances in machine learning (Church & Mercer, 1993). At about the same time, a shift to ever-larger projects increased the salience of the "knowledge acquisition problem" — that without a vast amount of knowledge, systems will be too brittle and will fail on anything outside of what has been expressly modeled. As a result people working in NLP use techniques that stop with just a description of what a text says and has none of the active, "fill in the gap" inferential capability that is critical for full deep language understanding.

We agree that knowledge modeling is difficult. It is intellectually challenging to come up with conceptualizations that have the requisite sensitivity to context, capacity to compose, and associated expectations for actions and inference. But this background knowledge is absolutely needed if automated systems are to learn from reading or fully understand our instructions. We are not alone in this belief, as witnessed by the steady body of work by people such as Len Schubert (Durme, Michalak, & Schubert, 2009) and Jerry Hobbs (Montazeri & Hobbs, 2011; Hobbs & Gordon, forthcoming). Moreover there are now significant knowledge stores to draw on. In addition to Schubert's KNEXT there is the MIT Media Lab's ConceptNet (Speer, Havasi, & Lieberman, 2008), FrameNet (Fillmore & Baker, 2001), and of course the long-term work of the CYC project (Guha & Lenat, 1993).

We do not presume to do this by ourselves. Once our designs have been refined through testing on a realistic corpus against the series of prototypes we will implement, we intend to formalize our knowledge requirements and look for assistance from like-minded people in the language-centric part of the knowledge-representation community for follow-on collaborations.

1.4 What we are actually doing

The focus of our work is on how inferences are marshaled from background knowledge when we use language. While it has old roots, our conception of how it is done is new. In order to focus our efforts, we have pushed to one side a set of issues that we know are important parts of any operational solution, but which now would just be a distraction.

- We are working from a corpus of written texts, not speech.
- We are not doing dialogue.
- We are not currently trying to acquire background knowledge automatically.

What we are doing instead is working out how highly efficient, lexically triggered inference and expectation can even be done at all. We are deliberately not yet invested in a choice of ontology or working with a large knowledge store. We think it is more important to test and refine our computational machinery before drawing on the work listed above and working at a proper scale.

In the next section we lay out the elements of our architecture, and illustrate them in §3 with the example that we drew on when formulating our design but have not implemented. That is followed in §4 by a smaller but thoroughly implemented example, that we walk through in detail.

2. Representation: Situations, Predicates, and Packets

Every cognitive architecture has a notion of *working context*, some means of defining and delimiting what it will attend to and what it can be aware of at any given moment. Every architecture also has a *control structure*, a policy or mechanism that dictates what actions it will take and in what order.

In our architecture – *C3*[2] – our working context is a structured *situation*, where what we mean by 'situation' is close to what it means in situation semantics (Barwise & Perry, 1983; Devlin, 2006). We use a data-directed, event-driven control structure adapting the techniques used in our language analysis engine *Sparser*.[3] We are focusing on the notion of a "situation type": a reoccurring pattern of events and participants. The situation semantics literature has instead focused on situations as a device that can provide a denotation for a complex of events and participants — a static representation similar to Schubert's notion of an episode in his episodic logic (Schubert & Hwang, 2000). The populated situation that accompanies an ongoing discourse supplies the information that is latent in what is perceived, i.e.., in the words of a text. Situations hold the general world knowledge that perception unconsciously brings to mind. They supply the bulk of the iceberg of information that lies below the tip that is perceivable.

At its base, the situation holds representations of the entities, events, and predications that have been mentioned in the ongoing discourse. It provides a minimal model, consisting of a set of typed structured objects. For example, if the text is "*a 14-year-old girl*" then when that phrase has been read, the situation contains representations of the girl, the age, and of the fact that the girl is described as being that age.

2. The name C3 stands for "the Compositional Construction of Context".
3. The Sparser 'sparser parser' is a high precision information extraction system. See (McDonald, 1992; McDonald, 1996). We are using it in in a new configuration designed specifically for C3.

2.1 Lexicalized Pragmatics

In a lexicalized grammar, the terminals of the rules are specific words instead of lexical categories such as proper noun or transitive verb. We propose to lexicalize meaning and inference – to establish it directly from the incremental composition of the meaning of the words in a text without an intervening logical form.

The meaning of words, phrases, and meaning-bearing constructions is defined in terms of the set of propositions, relations, or potential inferences they convey. Situations are created dynamically by composing these *packets* of content and inference as the words of a text are scanned. Most packets correspond to small individual categories or inferences: e.g., the affordances of a cup as a container, the consequences of a process being canceled. Packets are small because they are designed to compose with other packets to collectively define the suite of inferences that are active in a situation. Packets are activated singly or in groups according to what work they are designed to do and how and where they are triggered. The notion of *packet composition* is how we expect to satisfy one of the fundamental properties of language that have been recognized since the time of von Humboldt: the ability to make infinite use of finite means.

2.2 Predicates linked to language

As a concrete example of a packet, consider the word *black*. It is the English realization of the individual in the ontology that is used to represent the color black, (denoted as `black`) as opposed to other colors such as red or titanium white. Like all colors, it is associated with a two-place predicate that establishes a relationship between an entity that can have a color (tree leaves, cars, etc.) and the specific color `black`. The predicate looks like this, where the type of object to which the predicate can apply is restricted: it must include the type `has-surface`.

$$\lambda x_{hasSurface}[color\_of(x, black)]$$

The object and the predicate together are the contents of the packet. When the parser scans *black*, these packets are introduced into the situation.

Every predicate in the ontology must specify the word or fixed phrase that expresses it and their linguistic properties.[4] The knowledge engineer adding colors to his conceptual model must indicate the word or phrase that names the color and that it has the syntactic patterns of a predicate adjective. For C3, we do this using the notation for simultaneously defining semantic categories and their realizations described in (McDonald, 1994).

2.3 Latent predicates

When a phrase is fully instantiated, as in "*a black SUV*", the predicates receive values and establish predications. For example, the value of the color property of this SUV is bound to `black`. The meaning of substantive nouns or verbs will typically include a great many predicates, only a few

4. We use a Lexicalized Tree Adjoining Grammar for analysis and generation. A word's linguistic properties are established by indicating what TAG tree family or families it goes with. See (McDonald & Pustejovsky, 1985; McDonald, 1996).

of which will be present in a text and therefor explicitly represented as predications in the minimal model. The other predicates are *latent*. They may be relevant as the text continues; they may supply default assumptions that drive implicatures; or they may simply remain part of the background knowledge associated with the word; see §4.3.

In C3 we treat predicates formally as a kind of *lambda variable*. These variables are structured objects that define a relationship between individuals of specific categories, and are constrained in the range of values they can take, i.e. what the variable can be bound to; for details see (McDonald, 2000). This information is self-contained within the object that defines the variable – the category of individuals it applies to, the restrictions on its possible values, and the default values that can be assumed in the absence of actual ones.

For example, if the the participants of an event are physical objects then it is always the case that the event happened at some location, even if we don't know what that location is. This is the sort of thing that latent variables are used for. When the analysis of our initial example had only gotten this far: "*a 14-year-old girl died*," we knew that the death must have happened at some location, but we didn't know what that location was. The location could still be referred to, but only indirectly: "*where the girl died*" or "*the place where the girl died*." Once the text continued, "*. . . in the Kurdish city of Sulaimaniya*," the latent variable that represented the location of the event was accessed and bound to that city. Note that this narrows the category of the location to `city`, and we would say "*the city where the girl died*."

Pre-anchored latent variables In our implementation, a composite category (§2.6) defines all the possible properties, relationships, and habitats that its instance individuals can have or be part of, all implemented by lambda variables.

When we introduce a packet into the situation, this potential becomes accessible, even when just a small part of it is present in the minimal situation model. It is accessible in that any later explicit reference to a predicate that is latent in an individual's composite category has already been linked to the individual it applies to. We employ a wrapper around all variables, what amounts to a programming trick, that permits C3 to create an instance of each variable (potential predication) linking he individual it is predicated of instantaneously in one step, at the moment the individual is introduced into the situation.

2.4 Frames and Habitats

Packets are C3's building blocks. Most packets contain roughly the same amount of information as we intuitively associate with a single word (*black*, *cancel*). But of course there are relational structures that are considerably bigger, structures that should be instantiated as a single unit but which have multiple parts and activities such as the representation of an airport, a government building or a birthday party. Historically, these have been done as frames (Minsky, 1975).[5] We have

5. Drawing on Bartlett's notion of a schema, Minsky developed the concept of a frame in 1972 in reaction to Newell and Simon's book "Human Problem Solving". Originally, frame theory emphasized the transformations that would occur as perspectives changed or scenarios progressed There was a focus on frame recognition and repair to account for variations. Action was tied to the creation of frames and to changes in their slot values via "attached procedures." Frames subsequently evolved into today's RDF triple-stores and weakly expressive decision logics, in the process losing most of their value as a representation of background knowledge.

returned to something close to original conception of frames, but built from modern computational tools.

We have added an organizing overlay to Minsky's frames by adopting the notion of a *habitat* (Pustejovsky, 2013a). This is an extension and deepening of the well-established concept of *qualia theory* (pg. 144). We introduce a habitat into the situation all at once, but which aspect of it is in focus (which gets priority in dictating interpretations and making inferences) depends on what is in focus in the text being read, as we illustrate on page 145. The term "habitat" deliberately plays on the ecological metaphor to guide intuition as to what should be included in a frame and what should not.

2.5 Indexical Functional Variables

The contents of a situation reside in a web of relationships and possibilities, most of them coming from the active habitats, others coming from the discourse relationships that structure the interpretation of the text, including relations that keep track of partial information as the text is being read. To represent this we use a set of indexical functional variables following the design used by Phil Agre in his Pengi system (Agre, 1988) These variables designate constant, funtionally identical relationships within the processes of the system while their values vary transparently to fit the moment-to-moment situation.

One of Agre's examples is the variable `the-cup-I-am-drinking-from`, which gets bound to whichever of the three cups that he kept in his office he was actually drinking from at the moment. The things he could do with this cup were always the same, drink tea or use as a paper weight, while the identity of the cup would vary. The actions the system takes are stated once in terms of indexical variables – the presuppositions and significance of a functionally designated object is always the same. Actions are not dependent on particular values only on the function those values serve. Their actual, runtime values are managed automatically and transparently according to the situation at hand.

Pegs For Agre, the deictic variables were managed by the Pengi perceptual system. For us they are managed by the parser and identify the structure it has observed and the relationships it expects. In most instances an indexical such as `theme` or `new` will be bound to specific, typed individual, but since we are updating the situation incrementally as each word is scanned, there are always moments where a phrase is incomplete, its head and type not yet identified, but we still need to establish its impact on the situation. To do this we are using Susan Luperfoy's notion of a *peg* (Luperfoy, 1992).

For example, at the point in the parse where we have read just "*a 14-year-old*" the indexical variable `current-np-referent` is bound to a peg that was created when the parser scanned the "*a*" and recognized that it was starting a noun phrase that would have a referent. The peg is a standin for the eventual referent, and provides a place to accumulate predications. In this instance, we know that whatever this referent may turn out to be, it is something for which it makes sense to have an age measured in years. The peg's properties are transferred to a regular individual once the head of the NP (*girl*) has been scanned. See §4.2 for another example.

It is an interesting psycholinguistic question whether the earlier context has established the overall topic and narrowed the semantic field from which the referent of an incomplete phrase like

"*14 year old*" will be drawn. The news article that this excerpt appeared in had "bird flu" in its title. Anyone familiar with the subject will know what types of individuals will be discussed, and given the age phrase will presume that it will be a person. In other contexts, for example at a bar, the presumption might be that the 14 year old was a single malt scotch. Whether people use such pre-established semantic fields or wait a moment to hear the head word could be tested in a well-designed experiment.

2.6 Representational principles and their consequences

We have arrived at a set of principles for the representation of world knowledge in C3. These are an overlay on an otherwise conventional system of categories and properties in a specialization lattice. The aim is to provide a flexibile link from language to the ontology while retaining the economy of only having to state axioms and relation types once.

- Only add a category to the ontology (T-Box) if it makes a contribution, e.g. it adds predicates, state-change affordances, presuppositions, or defaults.
- No representation without realization – Every category should correspond to some word, phrase, feature, or syntactic construction.
- Predicates are only defined once. They may be restricted to different values at different levels in the category lattice but they retain their identity.

In a conventional representation there is a substantial distance in the specialization lattice between the particulars that appear in a text, such as a sport utility vehicle, which will be close to the bottom, and what we know about the vehicle, e.g. that it is a `container`, which is stated at a high level and applies to a great many things besides SUVs. It is difficult to use language in such a system. Our need to put have packets for domain-specific words that refer to general predicates and affordances (our *lexicalized pragmatics*) cannot be easily accommodated.

Unique variables We chose instead to separate the realization facts (what words and construction are used) from the axiomatic facts (what predicates and operations apply and what follows from them). An SUV acts like a container because its category literally incorporates the `container` category and uses the container's variables to express the affordances available to its passengers and to state facts such as when one passenger gets out there is one fewer inside.

We do this by making all variables (predicates) unique. They are defined once, as one object in the representation, on a category as far up in the lattice as possible for maximal application. On more specific categories the variable will usually be restricted. For example the `contents` variable of `container` is defined there as just a collection of an unknown number of entities of unknown types. When we move down to, say, `passenger-transporter` (see pg. 146), the type of the collection is restricted to `person`. On a particular type of `passenger-transporter`, say `airline`, the restriction on the variable will be further restricted to incorporate the different roles that the people on an airline have.

The vocabulary, the realization facts, is stated against these restrictions. Any packet that includes `container` adds to the situation model that fact that its contents are in one of two states, expressible as being *in* (*inside*) or *out* (*outside*) of the container, and have the affordance of being able to move between these states. But we say that we *take* or *pick out* jelly beans from a jar (they cannot move

on their own). We watch a squirrel *climb out* of a garbage can (they can move on their own, and the movement involves ascending a hight). When the variable is restricted to the category `person` we refer to to them as *passengers* or by their role (*driver*, *pilot*, *stewards*). and they *go into* or *get out of* the container.

Pre-cached, "composite" categories Allowing different local restrictions on the same predicate object lets us achieve an economy of expression for axioms, which is essential for working with large ontologies, while retaining flexibility in how to define packets of the vocabulary since realization facts can refer to restriction categories at very different levels in the ontology. But this comes at a cost, since any word with a rich meaning will have a packet that introduces dozens if not hundreds of latient variables (particularly for habitats) which will entail including a proportional number of categories.

We make this manageable by using what we call *composite categories*. We define them as a conjunction of regular categories. We then pre-cache the categories' variables (with their restrictions) to create a single computation object. The result has the behavior we would get by using ordinary inheritance, but with none of the costs of traversing the lattice to collect the variables and apply the restrictions. In the implementation, a composite is just a class with mixins for each of the categories it includes.[6]

While a a composite category is often just collecting the categories that are above it in the hierarchy, there is no requirement to do so. Categories from very different parts of the ontology can be incorporated into a single composite. This makes for an ontology that is easier to maintain, since there is no requirement to force everything into a single lattice with coherent lines of inheritance.

Composite categories can be incorporated into other composites. When this happens, the incorporated composites are treated like macros that are unpacked inline and repackaged as a new CLOS class.

2.7 The C3 Architecture

Figure 1 shows the basic framework of C3 using the example discussed in next section. Solid blue lines from the text trace the activation path up from the first part of the text to add packets (in green) or larger habitat frames (in blue) and become part of the situation as a whole (outer box). Dotted lines show later additions to the situation (upward arrows) or inferred interpretations made by the situation (downward arrows). Orange arrows within the situation sketch relationships developed among the packets by binding variables.

We can summarize C3's workflow as follows. It begins with the perceived input. In our research this is the sequence of words in a text. Words are interpreted as they are reached by the parser and contribute packets of content of different sizes and function to a growing situation. This leads to the instantiation and assembly of highly structured sets of prototype relations and events, anticipated scenarios, and specific and prototypical individuals, places, etc. The situation then governs the expectations and interpretations of words and phrases as the analysis continues.

6. We work in Lisp, and make heavy use of the multiple inheritance capabilities of the Common Lisp Object System (CLOS); see (Gabriel, White, & Bobrow, 1991).

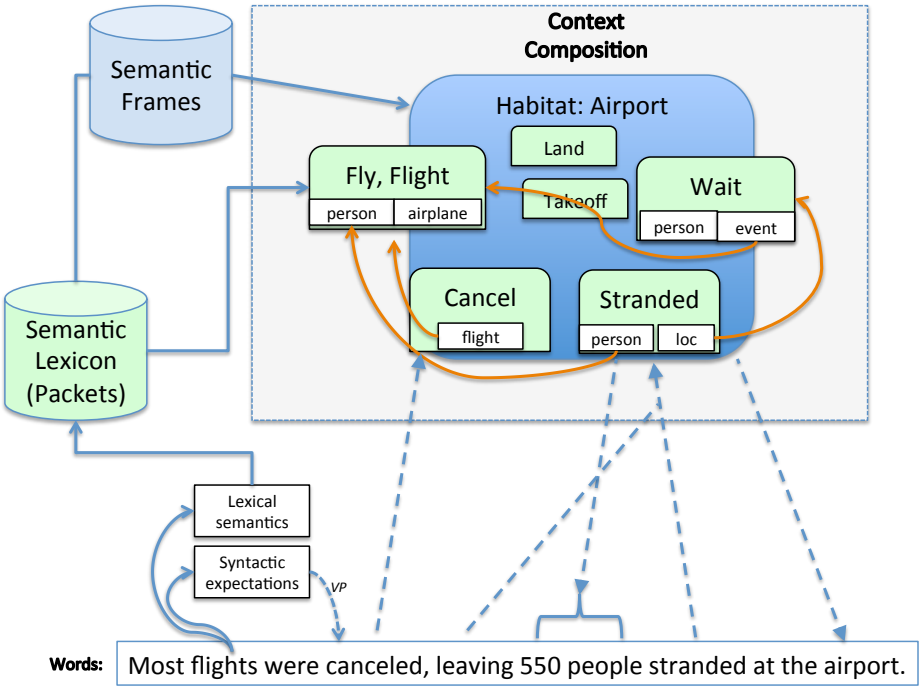

Figure 1. The C3 Architecture.

3. Air travel and inference

We will illustrate how our architecture works by describing how the situation is established and drives inferences during the comprehension of the following text.[7]

> "*Most flights from the Luis Munoz Marin Airport in San Juan to the Leeward Islands were canceled Monday, leaving about 550 people stranded at the airport.*"

If it was read for just its literal content, as today's language understanding systems would,[8] the result would leave many questions open. In particular,

- Who are these people?
- Why are they stranded?

7. This is a self-contained excerpt from a news article about the impact of Hurricane Earl on Puerto Rico (The New York Times, August 31, 2011).
8. The ongoing work of James Allen (Allen et al., 2007), and Peter Clark (Clark, Harrison, & Thompson, 2003) is a notable exceptions to the generalization that today's parsers read just for literal content.

3.1 Lexical structure

Outside of a specific context, most high frequency words are ambiguous. Even once a word sense has been determined there are still differences in logical perspective to sort out or metonymies to decode. We describe our approaches to these problems in this section.

Simple ambiguity Consider the word *flights*, which has different meanings in different domains. It could refer to a *flight of stairs* or be part of a fixed phrase like *flight from stocks*; it could refer to a quantity of beer or champagne. It could be a nominalization of *flee*. A fully populated language understanding system would have all of those reading and more. In the context of this example it of course refers to an airline flight, but we have to establish that fact before we can instantiate the `air-travel` habitat and activate its affordances.

We know from psycholinguistic studies that all of the senses of a polysemous word are active when the word is read, and that all but the intended sense drop away shortly afterwards (Marslen-Wilson, 1973). We get this behavior in C3 by using the approach that Sparser is already using. Each kind of 'flight' that C3 knows about (for which is has a packet in its lexicon) has its own projection to the grammar, and will introduce its own semantically-labeled reading (its own edge for the word in Sparser's chart) when the word is scanned, e.g. `airline-flight` and `fight-amount`. This mirrors the observed immediate activation of all the word's senses.

In such cases Sparser uses a simple disambiguation policy. Only the edge that extends through composition with the phrases around it has its meaning incorporated into the situation. The the others are ignored. In this example, the others are ignored at the moment the word *from* is scanned because that preposition is part of the rule pattern that applies to 'flights' as movement.

Lexical entries in the generative lexicon In Pustejovsky's Generative Lexicon theory (Pustejovsky, 1995; Pustejovsky, 2013b), the lexical entry for a content word (as opposed to a grammatical function word such as *most* or *from*) encodes three kinds of information (in addition to basic typing):

- Its **Argument structure**, which spells out what arguments the word takes, how they are realized syntactically and govern semantic role selection.
- Its **Event structure**. For verbs and many nominals, we can categorize the class of event it refers to (state, process, transition) and how it structures its implicatures (Pustejovsky, 1991).
- Its **Qualia structure**, the basis of logical polysemy, implicated in coercion and type shifting.

The argument structure is integrated into the rule sets of Sparser's grammar and helps with the task of simple disambiguation. The event structure is part of the habitats that are added to the situation and provides a scaffolding for anchoring events and action sequences in time. The qualia structure organizes the applicable predicates and affordances.

Qualia and Logical polysemy The Qualia consist of four basic roles, each of which can be seen as answering a specific question about the object it is associated with. Each contributes a complementary set of latent predicates to a word's meaning.

- **Formal Roles** encode taxonomic information about the lexical item (the `is-a` relation); *What kind of thing is it, what is its nature?*

- **Constitutive Roles**: encode information on the parts and constitution of an object (`part-of` or `made-of` relation); *What is it made of, what are its constituents?*
- **Telic Roles** encode information on purpose and function (the `used-for` or `functions-as` relation); *What is it for, how does it function?*
- **Agentive Roles** encode information about the origin of the object (the `created-by` relation); *How did it come into being, what brought it about?*

Most words have alternative readings that are characterized by different qualia: the newspaper you read (telic), the one you spill coffee on (constitutive), the one whose editorial opinions you disagree with (agentive). This distinction is referred to as *logical polysemy* (Pustejovsky & Boguraev, 1993). Once a content word has been narrowed to the domain where it has a specific meaning (simple disambiguation) the next step is to determine its qualia role, to disambiguate it logically.

The qualia role that applies in a particular instance cannot be determined independently of the rest of the context. If the text was *My flight just landed* it would be the constitutive role since we are talking about the airplane that the flight used and only physical things can land. If our flight was rescheduled it would be the agentive role. All of these alternatives are part of the `air-travel` habitat — a frame that factors into different parts (incorporated habitats) according to which qualia is involved. In this instance of *flight*,[9] it will be the telic role and it will link to the portion of the habitat that organizes knowledge about flights as conveying people from place to place.

Metonymy An important kind of inference is decoding what is actually meant when a general reference is used in place of a specific one. When "*the White House issued a statement*" it is not the building that did it but some spokesman and the identity of the spokesman is unimportant. This in happening in our example. The flights are "*from the Luis Munoz Marin Airport in San Juan to the Leeward islands . . .* "

The *from . . . to . . .* construction occurs in many situations, not just motion. In another context this could just as well describe movement in stock prices. The fact that we are in an `air-travel` situation imposes a interpretation on the from–to arguments that they refer to airports. The airport in San Juan is given explicitly. The destination, however, is given as just a named location. An inference is required to convert from that to an airport. (Note that the writer could equally well have just said "*from San Juan*" in which case both references would require a metonymic interpretation.) Here are the steps that C3 would follow to get an airport from a place. Stated in conventional form, the rule would be something like this. Given this axiom (meaning postulate):

$$\Box[\forall f{:}\text{flight}\ \forall l{:}\text{loc}[destin(l,f) \rightarrow \exists a{:}\text{airport}[at(a,l)\ \wedge\ destination(a,f)]]]$$

we can devise a coercion operation such as this:

$$\lambda f{:}\text{flight}\ \lambda l{:}\text{loc}[destin(l,f)] \stackrel{coerce}{\Longrightarrow} \lambda f{:}\text{flight}\ \lambda l{:}\text{loc}\exists a{:}\text{airport}[destin(l,f) \wedge at(a,l)]$$

In GL terms, we are coercing the named location *the Leeward Islands* into the airport that is associated with that location. In C3 we can deploy that inference efficiently without needing search or

9. Recall that the context is *"Most flights from the Luis Munoz Marin Airport in San Juan to the Leeward Islands were canceled Monday . . . "*

matching because we can use the old idea of *attached procedures*. Within the `air-travel` habitat, the origin and destination variables of a flight can only be bound to airports. This is already the case for the origin, but the destination is a place rather than an airport. Binding the destination to a place triggers a procedure to identify the needed airport in the situation. If it cannot be found then the rule simply creates a representation of 'the airports in the Leeward Islands' and adds it to the situation as the value of the destination.

3.2 Habitats, actions, and composition

Airports have control towers, runways, taxiways, gates, terminals, etc. These are all available in the airport habitat. As we described in §2.4, these are entities and relationships that the habitat knows about, but they are latent rather than part of the minimal model actually in the situation.

The principal activity at airports is air-travel, and if we ignore its personal aspects (making reservations, getting to/from the airport, buying food, shopping, etc.), the most salient aspect of air-travel is the flights. Flights are also habitats. They have a plane (the equipment), a crew, passengers, baggage, food, etc. They are run by particular airlines, have a flight number, and travel from one airport to another.

In the telic reading of flight, the habitat includes a script that lays out the typical sequence of events and activities that constitute air travel. Airplanes are containers and they can move. Like any moving container, when they move (taxi, take off, fly, land) they convey their contents with them from their starting point to their destination. There are enough of these passenger-transporters in the world that they form a useful composite class: cars, buses, trains, bicycle-pulled carts, trucks, etc. This ensures that their common core is shared, particularly for our purposes the words that accrue to this level, such as *passenger*.

The interpretation of *flight* is as a process. There is a state of affairs that holds before this process starts and a different one after it ends. The principal difference between these two is in the location of the airplane and its contents: the passengers, their baggage, the crew. Before the flight leaves they are at the origin airport, afterwards they are at the destination airport. Any habitat like flight that involves scheduled process comes with the default assumption that once the process has started it will continue until it ends.

To represent the content of the first part of this text, we instantiate a flight habitat with values for the variables that we know. This adds to the situation a collection of an indefinite number of individual flights, where each of these otherwise unidentified flights originates in San Juan and terminates in an airport in the Leeward Islands. Each of these flights has a carrier and a flight number, a crew and a passenger manifest, but these are latent properties, just as we don't know the actual number of flights in the collection.

Canceling movement The meaning of the word *cancel* is an operator, It modifies the situation rather than simply adding to it. *Cancel* is an operator over processes. Its syntactic configuration (as main verb) establishes that it is predicated of the value of functional variable `syntactic-subject`, i.e. the flights. Since the only qualia of flight that involves a process is its telic function of transporting its passengers from one place to another, that aspect of the `flight` habitat becomes central to the situation.

Applying the operator `cancel` to the flights cancels this process. To cancel a flight means that it does not start (the flights don't *take off*). This modifies the situation to reflect that fact that the conditions that held before the process would have started still obtain: the passengers who would have been on the flights are still at the San Juan airport, as are the crews and the planes.

Situation-driven binding In the last portion of this example we have a result clause

"*leaving about 550 people stranded at the airport.*"

Given its form, the syntactic relation of this adjunct to its main clause tells us that this state of affairs (the stranding of the people) happened because of the event in the main clause (the cancelation of most of the flights). Being stranded is a habitat in itself, associated with air travel but not a part of it per se in the way that, say, losing one's luggage is. The meaning of *stranded* is that there was an intention to move that has been blocked: The path of the passengers' expected futures has been interrupted. Note that the airport employees are not stranded, because they have a different role in the `air-travel` habitat, i.e., they work at the airport.

Inferences should be guided by what is salient in what is perceived – the text that C3 is interpreting and the situation model created for it. The cancelation brings into focus within the situation those elements that were most affected by it: the passengers, the air crews, and any other individuals whose intended future path of events was shifted. This salience makes it simple to interpret the two definite references in the result clause. Given the context provided by this situation, we can bind the referent of *the airport* to San Juan's Luis Munoz Marin airport because the `flight` habitat has already created properties for two airports (origin and destination). The origin airport is the more salient of the two because it is the one impacted by the cancelation.

Similarly, the *550 people* are resolved to be the only people who are made salient by the cancelation: the passengers and crew who would have been on the flights that did not take off — did not follow their intended, default future path.

4. A worked example

At SIFT we have access to a set of logs of actual text-chat collected from an Intelligence, Surveillance, and Reconnaissance team during the Empire Challenge 2010 (EC10) military exercise.[9] These are from a GBOSS team (Ground-Based Operational Surveillance System) that was composed of three camera operators, an analyst, and a coordinator, all communicating over Internet Relay Chat reporting their on the movements and activities of the other players in this live Army exercise in a simulated set of Afghani villages. This excerpt illustrates the sort of gap that we are focusing on. Camera operator Heavy2 is reporting on an event involving a car 'of interest' in the Wakil village that he is observing.

9. We have permission from the Office of Naval Research to use these Ground-Based Operational Surveillance System (GBOSS) team transcripts, as well as similar EC10 tactical operations transcripts of sensor operators reporting status and commanders directing them.

Line	Time	Message
72	[19:51]	<Heavy2> black ford suv has entered wakil
73	[19:52]	<Heavy2> two people are dismounting

Table 1. GBOSS team chat excerpt from EC10.

It is obvious to us where the people came from. In this section we layout how we make it equally obvious to a computer program.

4.1 The initial situation

Line 72 of the chat transcript, entered at 19:51 pm, is the first time that observer Heavy2 has typed anything for several minutes. This speaker shift has cleared the situation of any active habitats or facts, and moved their content to a passive store from which they can be reactivated when mentioned again. In this case, the "*black Ford SUV*" was already identified and designated as a 'vehicle of interest' earlier at 18:27, and at 18:50 there was the report "*three guys have gotten in to black ford suv at wakil*." Not only is there a known individual to add to the situation (rather than building a new individual), but we already know something about it.[10]

```
SUV-1:  container.contents = collection(count > 3, type = person)
```

The discourse history established that The SUV is value of the `given` indexical variable. The value of the `new` variable is the fact that it has entered the village. This re-introduces this already known village in to the situation model, along with the fact of the event, but nothing else. We know the present location of the SUV (it is part of the minimal model), but we do not know anything about its previous location except that it has one: "*where the SUV was before it entered Wakil*."

We do not know anything else about the SUV, not even whether it has stopped moving. In the actual world of the observer, all of this is an esablished part of reality. It approached along a particular road at a particular angle to the viewer. The sun was shining and created shadow of a particular size. The buildings in Wakil are made of concrete and painted some color. While all of this is true, only what we actually know from the text is present in the situation. The rest is latent.

4.2 Expectations

For C3, Sparser parses texts incrementally word by word so as to get the greatest amount of leverage from the situation. From line 73, reported a minute after the report about the SUV, it first reads the word *two*. As a nominal premodifier, that deploys a peg and its packet establishes that there is a collection of size two, but that is all we know at that moment. It could have been that two windows on the SUV were opened, or two of its doors.

```
Peg(x):  collection(count = 2, type = x)
```

Upon reading *people*, the head of the NP, the peg is replaced by an individual representing a collection of two people, but again we know nothing more. We do have an expectation however. The

10. The expressions used in this section are purely notional for purpose of illustration. In C3's implementation their equivalents are configurations of type-objects linked by pointers and organized by indexical-variables bound by the situation object. Describing their actual elements and organization cannot be done is the space available.

people must have been somewhere before this even if we don't yet know where. Since some things, like the locations of the objects of discourse, are essential to understanding. (physical objects don't just appear in a puff of smoke), this information gap leads to an expectation that we will either be told the location or should assume one given the available evidence.

```
people-2:  type = collection-2, physical-object.location = ?)
```

4.3 Composition

Then Sparser reads the verb group "*are dismounting*." It adds the packet for *dismount* to the situation and notes that this is an ongoing action.

dismount = `transition.inprogress„ movement.from = high, movement.to = low(ground) movement.actor = v:subject`

From the syntactic construction, it knows that the collection of people supplies the obligatory argument to *dismount*: who is doing the action.

`Dismount` is a movement. Every instance of a movement comes with predicates for where its participants (the two people who are moving) were before the action and where they are after it. None of these values have been given explicitly, though a firm default for `dismount` is that the final location is the ground. (One dismounts from a horse or a piece of gymnastics equipment.)

To establish the value of their prior location (where they dismounted *from*), we use what amounts to anaphoric reasoning: namely, what are the locations that we know about given the present situation? This gives us the village and the SUV, but the SUV should be preferred because the thing one dismounts from must be close by (compare "*two people are walking up to it*") and the SUV is salient because it is the value the discourse `theme` indexical because is is a 'vehicle of interest.'

```
during.before(dismount-1):  people-2.physical-object.location = SUV-1
dismount-1.movement.from = SUV-1
```

This binding furthermore has significant side effects. To be dismounting from the SUV presupposes that it is stopped, so we coerce the motion of the SUV in 72 to a 'stopped state.' (Compare secret service agents dismounting from the presidential limo during a motorcade.) If two people have *left* the SUV, qua container, then the number of people known to be in the vehicle (at least four) is reduced by two.

What has happened is that the introduction of the `dismount` to the situation initiated a limited inference process to identify the location the people dismounted from. Integrating the `dismount` with the established `enter` or the SUV provides a 'people-containing' location to the inferential search ('inside the SUV'). If there had not already been such a location in the current situation, the search would not go any further, and just posit that the location exists and wait for more information to come in, just as with our initial example of the Iraqi girl.

5. Conclusion

In this paper, we have presented a computational architecture for a new way to deploy and exploit the knowledge and inferences in a word's meaning. Much of it is simply reviving techniques that had gone out of style, but there are also innovations designed to increase efficiency and semantic transparency.

- Treat situations computationally as the sum of the understanding of what has been said, along with what is implied and what might follow §2.1.
- Organize the meaning of words as "packets" of model-level content along with overt and implicit predications §2.2.
- Use a representation that enables constant-time access to the knowledge that is latient in the situation §2.3.
- Provide function-based landmarks to the content of a situation to permit one-step application of anaphoric-style inferential gaps §2.5.
- Separate the linguistic realization from the statement of the axiomatic facts by defining predicates just once and stating wording constraints over restrictions on them §2.6.

Clearly, there is much more to be fleshed out, and it is difficult to evaluate our proposal without more elaborate and more extensive modeling. But the outline presented here suggests a specific way in which people deploy their linguistic and general knowledge jointly, to understand discourse, and we invite those who think that there is merit in our goal — to understand how people can deploy their knowledge through language as quickly and effortlessly as they walk or breathe — to an extended conversation about how this is possible.

Acknowledgements

This work was supported in part by the Department of the Navy, Office of Naval Research under grant N00014-13-1-0228. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the Office of Naval Research.

References

Agre, P. E. (1988). *The dynamic structure of everyday life* (Technical Report). MIT Artificial Intelligence Laboratory.

Allen, J., Manshadi, M., Dzikovska, M., & Swift, M. (2007). Deep linguistic processing for spoken dialogue systems. *Proceedings of the Workshop on Deep Linguistic Processing* (pp. 49–56).

Bartlett, F. C. (1932). *Remembering*. Cambridge University Press.

Barwise, J., & Perry, J. (1983). *Situation semantics*. Stanford, CA: CSLI Publications.

Bergen, B., Chang, N., & Narayan, S. (2004). Simulated action in an embodied construction grammar. *Proceedings of the Twenty-Sixth Annual Conference of the Cognitive Science Society* (pp. 108–113).

Church, K. W., & Mercer, R. L. (1993). Introduction to the special issue on computational linguistics using large corpora. *Computational linguistics*, *19*, 1–24.

Clark, P., Harrison, P., & Thompson, J. (2003). A knowledge-driven approach to text meaning processing. *Proceedings of the HLT-NAACL 2003 workshop on Text meaning-Volume 9* (pp. 1–6).

Devlin, K. (2006). Situation theory and situation semantics. *Handbook of the History of Logic*, *7*, 601–664.

Durme, B. V., Michalak, P., & Schubert, L. K. (2009). Deriving generalized knowledge from corpora using wordnet abstraction. *EACL-09*.

Fillmore, C. J., & Baker, C. F. (2001). Frame semantics for text understanding. *Proceedings of WordNet and Other Lexical Resources Workshop*. Pittsburgh: NAACL.

Gabriel, R. P., White, J. L., & Bobrow, D. G. (1991). Clos: Integrating object-oriented and functional programming. *Communications of the ACM*, *34*, 29–38.

Grice, H. P. (1975). Logic and conversation'in p. cole and j. morgan (eds.) syntax and semantics volume 3: Speech acts.

Guha, R., & Lenat, D. (1993). Cyc: A midterm report. *Readings in knowledge acquisition and learning* (pp. 839–866).

Hobbs, J. R., & Gordon, A. (forthcoming).

Luperfoy, S. (1992). The representation of multimodal user interface dialogues using discourse pegs. *Proceedings of the 30th annual meeting on Association for Computational Linguistics* (pp. 22–31).

Marslen-Wilson, W. (1973). Linguistic structure and speech shadowing at very short latencies. *Nature*, 522–523.

Matsuki, K., Chow, T., Hare, M., Elman, J., Scheepers, C., & McRae, K. (2011). Event-based plausibility immediately influences on-line language comprehension. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *37*, 913–934.

McDonald, D. (1992). An efficient chart-based algorithm for partial-parsing of unrestricted texts. *Proceedings of the third conference on Applied natural language processing* (pp. 193–200).

McDonald, D. (1996). The interplay of syntactic and semantic node labels in parsing. *Recent advances in parsing technology*, *1*, 295.

McDonald, D., & Pustejovsky, J. (1985). Tags as a grammatical formalism for generation. *Proceedings of the 23rd annual meeting on Association for Computational Linguistics* (pp. 94–103).

McDonald, D. D. (1994). Reversible nlp by linking the grammar to the knowledge base. In *Reversible grammar in natural language processing*, 257–291. Springer.

McDonald, D. D. (2000). Issues in the representation of real texts: The design of krisp. In L. M. Iwanska & S. C. Shapiro (Eds.), *Natural language processing and knowledge representation*, 77–110. MIT Press.

Minsky, M. (1975). A framework for representing knowledge. In P. H. Winston (Ed.), *The psychology of computer vision*, 211–277. McGraw-Hill.

Montazeri, N., & Hobbs, J. R. (2011). Elaborating a knowledge base for deep lexical semantics. *Proc. of 9th International Workshop on Computational Semantics* (pp. 195–204).

Pustejovsky, J. (1991). The syntax of event structure. *Cognition*, *41*, 47–81.

Pustejovsky, J. (1995). *The generative lexicon*. MIT Press.

Pustejovsky, J. (2013a). Dynamic event structure and habitat theory. *Proceedings of GL2013* (pp. 1–20).

Pustejovsky, J. (2013b). Type theory and lexical decomposition. In *Advances in generative lexicon theory*, 9–38. Springer Netherlands.

Pustejovsky, J., & Boguraev, B. (1993). Lexical knowledge representation and natural language processing. *Artificial Intelligence*, *63*, 193–223.

Schank, R., & Abelson, R. (1977). *Scripts plans goals and understonding*. Lawrence Erlbaum Associates.

Schubert, L. K., & Hwang, C. H. (2000). Episodic logic meets little red riding hood. In L. M. Iwanska & S. C. Shapiro (Eds.), *Natural language processing and knowledge representation*, 111–174. MIT Press.

Speer, N., Reynolds, J., Swallow, K., & Zacks, J. (2009). Reading stories activates neural representations of visual and motor experiences. *Psychological Science*, *20*, 989–999.

Speer, R., Havasi, C., & Lieberman, H. (2008). Analogyspace: Reducing the dimensionality of common sense knowledge. *AAAI* (pp. 548–553).