Microplanning with Communicative Intentions: The SPUD System

Matthew Stone Christine Doran Bonnie Webber Tonia Bleam Martha Palmer
Rutgers MITRE Edinburgh Northwestern Pennsylvania

Abstract

The process of microplanning in Natural Language Generation (NLG) encompasses a range of problems in which a generator must bridge underlying domain-specific representations and general linguistic representations. These problems include constructing linguistic referring expressions to identify domain objects, selecting lexical items to express domain concepts, and using complex linguistic constructions to concisely convey related domain facts.

In this paper, we argue that such problems are best solved through a uniform, comprehensive, declarative process. In our approach, the generator directly explores a search space for utterances described by a linguistic grammar. At each stage of search, the generator uses a model of interpretation, which characterizes the potential links between the utterance and the domain and context, to assess its progress in conveying domain-specific representations. We further address the challenges for implementation and knowledge representation in this approach. We show how to implement this approach effectively by using the lexicalized tree-adjoining grammar formalism (LTAG) to connect structure to meaning and using modal logic programming to connect meaning to context. We articulate a detailed methodology for designing grammatical and conceptual resources which the generator can use to achieve desired microplanning behavior in a specified domain.

In describing our approach to microplanning, we emphasize that we are in fact realizing a deliberative process of goal-directed activity. As we formulate it, interpretation offers a declarative representation of a generator's communicative intent. It associates the concrete linguistic structure planned by the generator with inferences that show how the meaning of that structure communicates needed information about some application domain in the current discourse context. Thus, interpretations are PLANS *that the microplanner constructs and outputs. At the same time, communicative intent representations provide a* RICH AND UNIFORM RESOURCE *for the* PROCESS *of NLG. Using representations of communicative intent, a generator can augment the syntax, semantics and pragmatics of an incomplete sentence simultaneously, and can work incrementally towards solutions for the various problems of microplanning.*

Contents

Contact Address
Matthew Stone
Department of Computer Science
and Center for Cognitive Science
Rutgers, the State University of New Jersey
110 Frelinghuysen Road
Piscataway NJ 08854-8019
mdstone@cs.rutgers.edu
August 27, 2003

CONTENT PLANNING
MICROPLANNING
REALIZATION

Figure 1: Microplanning in the NLG pipeline.

1 Introduction

Success in Natural Language Generation (NLG) requires connecting domain knowledge and linguistic representations. After all, an agent must have substantive and correct knowledge for others to benefit from the information it provides. And an agent must communicate this information in a concise and natural form, if people are to understand it. The instruction in (1) from an aircraft maintenance manual (USAF, 1988) suggests the challenge involved in reconciling these two kinds of representation.

(1) Reposition coupling nut.

The domain knowledge behind (1) must specify a definite location where the coupling nut goes, and a definite function in an overall repair that the nut fulfills there. However, the linguistic form does not indicate this location or function explicitly; instead, its precise vocabulary and structure signals one to draw on one's existing understanding of the repair to fill in these details for oneself.

In the architecture typical of most NLG systems, and in many psycholinguistic models of speaking, a distinctive process of MICROPLANNING is responsible for making the connection between domain knowledge and linguistic representations.[1] Microplanning intervenes between a process of CONTENT PLANNING, in which the agent assembles information to provide in conversation by drawing on knowledge and conventions from a particular domain, and the domain-independent process of REALIZATION through which a concrete presentation is actually delivered to a conversational partner. These processes are frequently implemented in a pipeline architecture, as shown in Figure 1. Concretely, the content planner is typically responsible for responding to the information goals of the conversation by identifying a body of domain facts to present, and by organizing those facts into a rhetorical structure that represents a coherent and potentially convincing argument. Microplanning takes these domain facts and recodes them in suitable linguistic terms. Finally, realization is responsible for a variety of low-level linguistic tasks (including certain syntactic and morphological processes), as well as such formatting tasks as laying out a presentation on a page or a screen or performing speech synthesis. See Reiter and Dale for a thorough overview of these different stages in NLG systems (Reiter and Dale, 2000).

Microplanning often looks like a grab-bag of idiosyncratic tasks, each of which calls for its own resources, representations and algorithms. For example, consider the three microplanning tasks that Reiter and Dale survey: referring expression generation, lexical choice, and aggregation.

[1] The name microplanning originates in Levelt's psycholinguistic model of language production (Levelt, 1989), and is adopted in Reiter and Dale's overview of NLG systems (Reiter and Dale, 2000). The process has also been termed SENTENCE PLANNING, beginning with (Rambow and Korelsky, 1992).

- In referring expression generation, the task is to derive an identifying description to take the place of the internal representation of some discourse referent; see e.g., (Dale and Haddock, 1991; Dale, 1992; Dale and Reiter, 1995). An identifying description must distinguish the intended referent from the salient alternatives, using descriptive concepts known in context to characterize the intended referent but not the alternatives; accordingly, the generator requires a model of the contextual background as input. In constructing the description, generators often maintain representations of the provisional semantic specification of an utterance (*the rabbit*, say) and the alternatives it would refer to (all the rabbits in the context). Generators proceed by executing rules to refine these representations by incorporating additional descriptive concepts (for instance *white*, to yield *the white rabbit*, referring to the white rabbits in the context).

- In lexical choice, the task is to select a word from among the many that describe an object or event; see e.g., (Nogier and Zock, 1991; Elhadad et al., 1997; Stede, 1998). Lexical choice requires specifications of the meanings available in the language and the lexical and grammatical structures that express those meanings; the generator also starts from the domain content that an utterance should express. To perform lexical choice, generators often invoke a pattern-matching process that rewrites domain information (that there is a caused event of motion along a surface, say) in terms of available language-specific meanings (to recognize that there is *sliding*, for example). Along the way, the generator maintains intermediate representations that record the linguistic words and constructions the generator has used and the further content that remains to be expressed.

- In aggregation, the task is to use modifiers, conjoined phrases, and other linguistic constructions to pack information concisely into fewer (but more complex) sentences; see e.g., (Dalianis, 1996; Shaw, 1998). The resources for aggregation include operators that detect relationships within the information to be expressed, such as repeated reference to common participants (that Doe is a patient and that Doe is female, say). The generator proceeds opportunistically to apply these operators and correspondingly reorganize related linguistic material into complex, nested structures (to obtain *Doe is a female patient*, for example). We identify aggregation as an abstract task of microplanning, neutral between syntax and semantics, but aggregation systems generally work by maintaining structures at one particular level of representation, and stitching them together.

Although they are typically treated by distinct and incommensurate generation processes, tasks like referring expression generation, lexical choice and aggregation in fact interact in systematic and intricate ways (Wanner and Hovy, 1996). These interactions represent a major challenge to integrating heterogeneous microplanning processes—all the more so in that NLG systems adopt widely divergent, often application-specific methods for sequencing these operations and combining their results (Cahill and Reape, 1999).

In contrast to this heterogeneity, we advocate a UNIFORM approach to microplanning. Our generator, called SPUD (for sentence planning using description), maintains a common representation of its provisional utterance during microplanning and carries out a single decision-making strategy using this representation. In what follows, we draw on and extend our preliminary presentations of SPUD in (Stone and Doran, 1996; Stone and Doran, 1997; Stone and Webber, 1998; Stone et al., 2000) to describe this approach in more detail. In so doing, we offer three contributions to the literature on microplanning.

First, we provide a simple intuitive basis for solving problems of microplanning in a uniform way. In our approach, the microplanner directly explores a search space for utterances described by a linguistic grammar. At each stage of search, the microplanner uses a model of interpretation, which characterizes the potential links between the utterance and the domain and context, to assess its progress in communicating domain-specific representations. We begin in Section 2 with a simple example of decision-making in microplanning that shows how grammar and interpretation provide a basis to solve problems such as referring expression generation, lexical choice, and aggregation. We substantiate the argument with detailed case studies in Section 6.

Second, we describe the techniques required to formalize and implement this model effectively. Search through derivations is facilitated by a grammar formalism that packages meaningful decisions together and allows those decisions to be assessed incrementally; SPUD uses the lexicalized tree-adjoining grammar formalism. To provide a conceptual framework to integrate the interpretive factors that influence choice in microplanning, we analyze interpretation via abstract representations of speakers' communicative intentions—adopting a Gricean approach to communication. The use of techniques such as logic programming and constraint satisfaction leads to efficient methods for calculating this interpretation for a given linguistic form. We describe our techniques fully in Section 4. We have made a number of implementations of SPUD available for research, including a full version written in SML and a lightweight version written in Prolog (Stone, 2002).[2] These implementations have formed the basis for several detailed exploratory characterizations of microplanning in specific domains (Bourne, 1999; Yan, 2000).

Third, we describe what we have learned about developing specifications for NLG in this framework. SPUD calls for rich syntactic, semantic and pragmatic characterizations of linguistic forms, and detailed descriptions of domain context. Formalizing this information is undoubtedly labor-intensive. However, once the information is available, SPUD uses it efficiently and declaratively. In this sense, SPUD changes microplanning from a programming problem into a problem for knowledge representation. We emphasize throughout this paper that the techniques and methods of practical knowledge representation carry over to the design of specifications for SPUD. In particular, as we discuss in detail in Section 7, developers can adopt a concrete methodology for designing grammars that will allow SPUD to achieve desired behavior in a specified domain.

Considerable further research remains before realizing this framework in a broad application. But we are optimistic that the principles of design, implementation and knowledge acquisition we articulate here will provide a solid and lasting basis for such work.

[2] See http://www.cs.rutgers.edu/˜mdstone/nlg.html.

2 Microplanning, Interpretation and Communicative Intent

This section offers an extended overview of our approach to microplanning. Our approach springs from a simple insight that frames decision-making about language in terms of more general decision-making. Specificially, microplanners that formulate representations of interpretation realize deliberative processes of goal-directed activity. Thus, we call our generator's representation of interpretation COMMUNICATIVE INTENT. In doing so, we emphasize that language use involves a LADDER OF RELATED INTENTIONS (Clark, 1996), from uttering particular words, through referring to shared individuals from the context and contributing new information, to answering open questions in the conversation. (Clark's ladder metaphor particularly suits the graphical presentation of communicative intent that we introduce in this section.) Since many of these intentions are adopted during the course of microplanning, communicative intent represents the RESULTS of generation. At the same time, we emphasize that microplanning is a deliberative process like any other, in which the provisional intentions that an agent is committed to can guide and constrain further reasoning (Bratman, 1987; Pollack, 1992). Thus, communicative intent also serves as a key resource for the PROCESS of generation.

We begin in Section 2.1, by showing how interpretation—understood as communicative intent—can link grammatical derivation to choices in microplanning. In Section 2.2, we use a high-level case-study of communicative intent to discuss more precisely how such representations may be constructed from linguistic and domain knowledge, and how they could be used to guide microplanning decisions. Finally, in Sections 2.3 and 2.4, we identify the key assumptions that we have made in SPUD, in order to construct an effective NLG system that implements a model of communicative intent, and we sketch some of the further investigations in which our implementation has figured.

2.1 Problem Statement

We view microplanning as a single, coherent problem in natural language generation. The microplanner has access to three sources of input.

- The first input is a specification of what LINGUISTIC RESOURCES are available, and how these resources can be combined together into complex utterances. The linguistic resources encode the microplanner's grammatical knowledge; they uniformly capture the concepts used in constructing referring expressions, the meanings that underlie lexical choices, and the operations of syntactic combination that enable aggregation.

- The second input is a description of the CONTEXT in which the utterance is to be produced. This input is a KNOWLEDGE BASE that provides the private background information that the system can draw on to formulate the utterance and spells out the alternative information that the hearer might use to understand it.

- The third input spells out the COMMUNICATIVE EFFECTS which the utterance is to achieve. These effects typically involve conveying particular propositions, and might for example be realized by particular lexical choices or particular operations of aggregation.

The microplanner's task is to formulate an utterance that will achieve the specified communicative effects in the specified context; the utterance must be a legal combination of the available linguistic resources.

For example, consider the microplanning task that might result in the instruction (2); we draw (2) from an aircraft maintenance domain that we have studied in detail and report on fully in Section 7.

(2) Lift reset lever slightly.

In this case, the linguistic resources will describe the possible utterances in the application. These resources will make available a range of action verbs, such as *lift*, *push* or *move*; they will make available a range of nouns for hardware, such as *lever*, *handle*, *coupling*, *sleeve* and so forth; and they may also offer descriptive modifiers like *slightly* and *reset* as well as grammatical function words like *at*, *it*, etc. Linguistic resources also must describe how these words are combined into grammatical utterances; for example, the microplanner must know that *lift* takes a noun phrase complement and allows a variety of optional verb phrase and sentence modifiers. The microplanner therefore has the information required to derive a range of possible sentences by assembling these resources together. Figure 2 sketches some of the derivations relevant to the construction of instruction (2).[3]

For instruction (2), the context will describe the equipment and its function in both linguistic and domain terms. That is, it will answer questions such as what parts the equipment has, what words tend to be used to describe each of these parts, and which of those parts are currently salient, perhaps because they are central to the task currently being performed or just because they have been evoked in recent discourse. Finally, the communicative effects will characterize the domain information that the utterance should convey. In this case, we might suppose what that the hearer needs to know is that specified object needs to be moved, that the application of force will be required, that the object will be displaced vertically, and that the distance moved will be small.

The solution to this microplanning problem encompasses the classic tasks of referring expression generation, lexical choice and aggregation. In choosing to identify the moved object with the telegraphic noun phrase *reset lever*, the microplanner must construct a referring expression. In choosing to describe the action with the verb *lift*, the microplanner makes a lexical choice. In choosing to qualify the verb with the modifier *slightly*, the microplanner aggregates the information that the distance moved will be small into a larger description that gives other information about the event.

However, we do not believe that the decisions of the microplanner are usefully decomposed in terms of these tasks. Any content that the microplanner plans for its utterance must be carried by specific lexical resources; the microplanner cannot act without making lexical choices. At the same time, steps of lexical choice can simultaneously further other aspects of microplanning

[3] Note that there is no requirement in these derivations to include required complements before optional modifiers. We can add any applicable resource at any time.

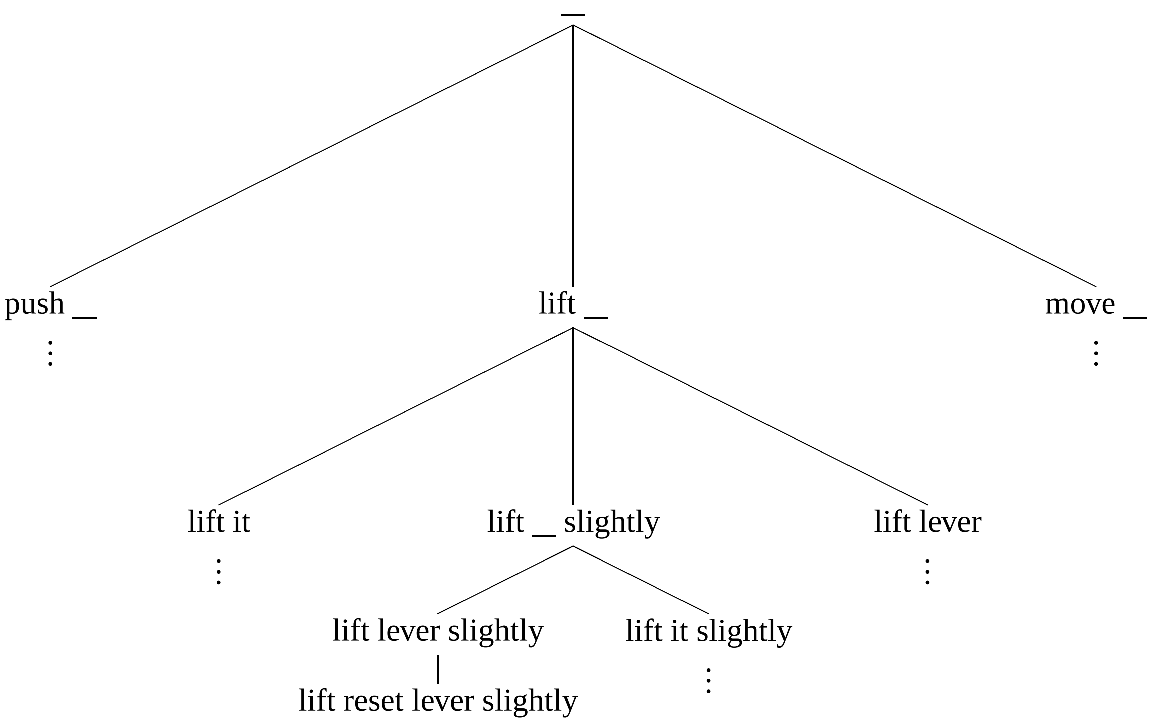


Figure 2: Microplanning as a search problem. The microplanner's grammatical knowledge describes a search space in which provisional utterances can be incrementally extended by linguistic elements with specified forms and meanings. The microplanner negotiates paths through this search space by assessing the interpretation of these provisional utterances.

(Stone and Webber, 1998). The choice of *lift*, for example, not only lexicalizes an action concept as needed to construct an instruction; the choice of *lift* also "aggregates" together in a single linguistic realization the information that the action involves motion, force and vertical displacement; and the choice of *lift* may also help to distinguish the object to be moved, since anything lifted must be capable of vertical motion.

Instead, we believe that decisions in microplanning are best characterized directly in terms of the linguistic resources that the microplanner selects for use in planned utterances. In other words, the microplanner should use the space of grammatical derivations to map out its possible choices, and so should act by adding resources step-by-step into the derivation of a provisional utterance. In deriving (2), for example, the microplanner should work directly in the search space displayed in Figure 2. The microplanner acts by selecting in turn the resources *lift*, *slightly*, *lever* and *reset*. This leads the microplanner down a particular path of derivation (the only one presented in full in Figure 2) until it arrives at a specific utterance that embodies a solution to its microplanning task. This paper describes the design and implementation of a microplanner we have implemented which works this way: SPUD (sentence planning using description).

A range of microplanning strategies are compatible with this general characterization, from incremental greedy decision-making to flexible heuristic best-first search. But all such strategies depend on evaluating combinations of linguistic resources as potential solutions to microplanning problems. We propose to measure microplanning progress by computing the PROVISIONAL INTERPRETATION of incomplete utterances. These interpretations formalize the hearer's ability to disambiguate an utterance, to link an utterance to the context, and to infer the intended conversational effects from it. Thus, a satisfactory interpretation shows that the hearer will resolve the references in an utterance as intended, and will get all the required information from the utterance. By contrast, an incomplete interpretation leaves references ambiguous or omits needed information. Accordingly, an incomplete interpretation requires the microplanner to put additional linguistic material into the utterance by exploiting the available linguistic resources.

Consider how interpretation could guide the microplanner along the path shown in Figure 2. The microplanner starts from the empty derivation at the root. At this stage of search, the generator knows that it has to find a sentence that describes the motion, force, distance and direction of an action the hearer is to perform. It begins by applying the linguistic resources that head up a sentence, and constructs provisional utterances consisting of one candidate verb. This assumes a rich repository of verb-specific and verb-class-specific semantic descriptions, such as that being developed in VerbNet (Kipper et al., 2000a; Kipper et al., 2000b; Dang et al., 2000).

To decide among the options *lift*, *push* and *move*, the microplanner evaluates their interpretations. These are all incomplete utterances, as they all fail to supply the required object noun phrase, and so, not surprisingly, they all leave the object of motion ambiguous. But these utterances differ in the communicative effects they have the potential to express; *move* describes the motion only; *push* describes motion and force; *lift* describes motion, force and verticality. The interpretation of *lift* goes the furthest. This gives the microplanner a reason to pursue this path in the derivation first.

At the next step, the microplanner compares provisional utterances *lift slightly*, *lift it* and *lift lever*. Again, none of these utterances specifies the object of motion uniquely, so none is complete. But the fact that *lift slightly* could be interpreted as conveying all four of the input communicative effects gives the microplanner a reason to pursue this derivation further. In favoring this derivation at this stage, our interpretive strategy selects what is effectively a step of aggregation.

The next stage of derivation contrasts *lift it slightly* and *lift lever slightly*. In such decisions, we expect the microplanner to prefer the pronoun, other things being equal. Pronouns have a contextual requirement that their referent is ACTIVATED or central to the current discourse, while definite noun phrases have a weaker requirement that the referent is merely uniquely identifiable in the context (Gundel et al., 1993). Thus, when the context supports the pronoun, the pronoun highlights a more specific fit between the utterance and the context than the definite noun phrase does. We can model this preference, like the others we have considered, as a function of interpretation—interpreting an utterance requires recognizing its links with context.

In this case, though, where we assume both expressions are ambiguous, this preference is not the decisive one. The microplanner should prefer *lift lever slightly*, which offers the potential to disambiguate the reference to the object. Thus, at this stage the microplanner effectively works to generate a referring expression. The microplanner may have an immediate reason to use the full noun phrase. When there are multiple salient objects, they will typically not all share the same type; in this case, the full noun phrase will be less ambiguous than the pronoun. But the microplanner's decision will ultimately be forced though additional lookahead or search, since no further descriptive modification will be forthcoming for *it*. Finally, the microplanner constructs *lift reset lever slightly*, and completes its task.

In this intuitive sketch of interpretation as a guide to microplanning, we implicitly appeal to a broad view of interpretation. Our model assumes that the hearer, in understanding an utterance, must recognize at least these three aspects of how the utterance is used:

- First, the hearer must recognize the conversational effects that the utterance is meant to achieve. In (2), for example, we saw that these effects specify that the hearer is to move a specified object, vertically, a small distance, by applying force.

- Second, the hearer must recognize which referents the utterance describes and evokes: in (2), it is the reset lever.

- Third, the hearer must recognize the contextual constraints that make it appropriate to frame the utterance in terms of specific linguistic resources. The instruction is an imperative; that choice shows (among other things) that the instructor's relationship with the actor empowers the instructor to impose obligations for action on the actor. (The maintenance instructions of (USAF, 1988) are actually military orders.) Meanwhile, the use of definite noun phrases that omit the article *the* reflects the distinctive telegraphic style adopted in these instructions. Of course, the relationship of instructor and actor and the distinctive linguistic style of the domain are established in the context, and the instructor anticipates that the actor will make connections with these shared representations in interpreting (2).

To use interpretation to guide microplanning, then, we will need to associate an utterance, such as that in (2), with a model of interpretation that describes these three components: how the utterance adds information that links up with the goals of communication; how it imposes constraints that link up with shared characterizations of objects; and how it establishes specific connections to the status of participants and referents in the discourse.

In accord with our strategy of constructing and assessing interpretations incrementally during search, our representations for microplanning record PROGRESS on these three components during search, not just the final results. Provisional interpretations may be incomplete. For example, provisional interpretations record progress towards unambiguous formulation of referring expressions by representing both the intended referents of an utterance and the candidate referents that the hearer might consider in context. Provisional interpretations record progress toward achieving required communicative effects not just by recording links between linguistic meanings and the constellations of domain information they are intended to express, but by indicating the communicative effects that have not yet been realized in the utterance. We make the simplifying assumption that these interpretations grow monotonically; they can be made more specific when additional material is added, but not revised. This is why we link steps of derivation to progress in microplanning—we can discard an interpretation as soon at it is flawed, rather than using the defeasible inference of more general models of interpretation (Lascarides and Asher, 1991; Hobbs et al., 1993) to extend the sentence in a way that corrects the flaw.

Our representations of provisional interpretation show how alternative choices of words and syntactic constructions suit an ongoing generation task to different degrees because they encapsulate different constellations of domain information or set up different links with the context. They also characterize how given structures and meanings may be elaborated with modifiers so that multiple pieces of information can be organized for expression in a single sentence. Our central claim about the uniformity of microplanning is that with such a rich model of interpretation, a microplanner can augment the syntax, semantics and pragmatics of an incomplete sentence simultaneously, and can assess its progress on the various interacting subproblems of microplanning incrementally.

Our approach to representing interpretation involves analyzing interpretations as communicative plans, and thus involves treating microplanning analogously to other processes of decision-making. The inspiration for this approach comes from Grice's description of communication in terms of intention recognition ((Grice, 1957), as updated by Thomason (Thomason, 1990)) and Clark's approach to language use as joint activity (Clark, 1996). In these theories, communicative plans map out how a speaker might use certain words to convey certain information: they describe the utterances of words and linguistic constructions, spell out the meanings of those utterances, and show how these utterances, with these meanings, could contribute structure, representing propositions and intentions, to the CONVERSATIONAL RECORD, an evolving abstract model of all the information interlocutors take as uncontroversial in a dialogue (Thomason, 1990). We use the term COMMUNICATIVE INTENT to refer to such structures—representations, built by reasoning from a grammar, which summarize the interpretation of an utterance in context.

The significance of this pragmatic perspective is that it emphasizes the role of inference in representing interpretation (Stone, 2003a). In communicative intent, the pairing between structure and meaning is specified by a grammar which describes linguistic analyses in formal terms. Likewise, links between domain knowledge and linguistic meanings are formalized in terms of logical relationships among concepts. To construct communicative intent, we draw conclusions about interpretation by reasoning from these specifications. We record the reasoning we do as inference representations. Thus, communicative intent is a DECLARATIVE representation; it enjoys the numerous advantages of declarative programming in Natural Language Processing (Pereira and Shieber, 1987). For example, the same representations of communicative intent can be used in an understanding module as in a generation module; both modules can construct these representations using the same linguistic resources and the same domain and contextual knowledge (Stone, 2003a). This symmetric characterization of generation and understanding facilitates the implementation of dialogue agents that pursue language interaction flexibly and collaboratively (Stone, 2003b).

2.2 Communicative Intent and the Representation of Interpretation

Any approach to microplanning based on communicative intent clearly depends on rich linguistic resources and rich models of interpretation. The richness of these representations does make them expensive to develop, but it does not require managing complex interactions or undertaking

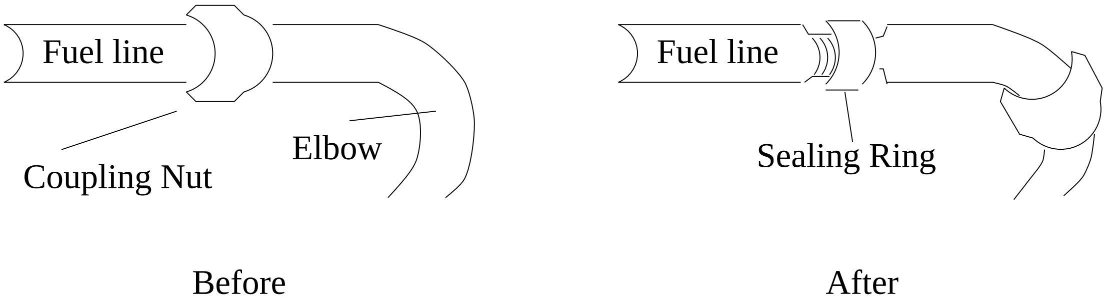


Figure 3: Carrying out instruction (3) in an aircraft fuel system.

open-ended formalizations. Rather, we argue that knowledge engineers for NLG may appeal to systematic methodology to construct such representations in a constrained and manageable way. The antecedents for this methodology lie in existing methodologies for practical knowledge representation, as exemplified by (Brachman et al., 1990), and in existing methodologies for formal description of language, as exemplified by (Butt et al., 1999; Doran et al., 2000). We make our case schematically here by considering the interpretation we need for (3).

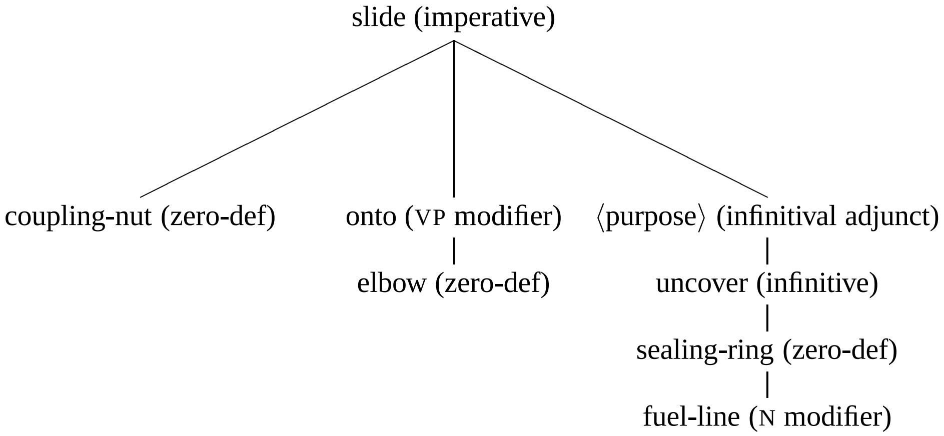
(3) Slide coupling nut onto elbow to uncover fuel-line sealing ring.

This utterance is representative of the upper range of syntactic and semantic complexity in a corpus of instructions we studied in detail: the F16 fuel-system instruction manual (USAF, 1988). We can describe this utterance in sufficiently systematic and precise computational terms in order to generate it in SPUD, while representing only a limited amount of relevant linguistic and domain knowledge.

Figure 3 shows the effect of the action on the aircraft fuel system. In this system, pipes are joined together using sealing rings that fit snugly over the ends of adjacent pipes. Sometimes these joints are secured by positioning a coupling nut around the seal to keep it tight and then installing a retainer to keep the coupling nut and seal in place. In checking (and, if necessary, replacing) such sealing rings, personnel must gain access to them by first removing the retainer and then sliding the coupling nut away. Figure 3 illustrates part of this process for a case where an instructor could use (3) to direct an actor to perform the step of sliding the coupling nut clear. In modeling this instruction, we assume that the speaker is presenting the utterance to the hearer in a situation in which the action is to be performed immediately. In formalizing this information, we thus describe the state of the task and the state of the discourse at a single moment in time, as the instruction is presented. To generate a sequence of instructions, we will update this knowledge base between successive steps of generation to reflect changes in the world from executed actions and changes in the context from realized utterances.

The first task for representation is to account for the linguistic structure of this utterance in a way that sets up a search space for generation analogous that in Figure 2. We use the lexicalized TAG formalism for this representation (Joshi et al., 1975; Schabes, 1990). Lexicalized TAG allows us to factor a standard linguistic surface syntax into atomic elements of primitive syntactic tree-fragments, each carrying specific content. We derive the sentence in microplanning by assembling these elements together. Schematically, we can represent a grammatical combination of elementary trees using a dependency representation as shown in (4); the elements correspond to the nodes in the tree.[4]

(4)



Each choice that arises in constructing this structure in generation realizes a specified meaning by adding concrete material to the incomplete sentence, as advocated by (Joshi, 1987). Thus, the labels of nodes in Figure 2 are shorthand for dependency structures such as (4).

We then pair (4) with a record of interpretation by formalizing two further sources of information: the GRAMMAR associates meaningful conditions with an utterance across contexts, in a public representation accessible to speaker and hearer; and the SPEAKER'S PRESUMPTIONS describe specific intended ways of linking these conditions to individuals in the context, and determine the specific intended communicative effects of the utterance.

We assume that the grammar associates each of the elements in (4) with an ASSERTION that contributes to the update intended for the utterance; a PRESUPPOSITION intended to ground the utterance in shared knowledge about the domain; and a PRAGMATIC condition intended to reflect the status of participants and referents in the discourse.[5] Following (Hobbs, 1985), we assume that

[4] The specific tree-fragments that compose (3) are given in detail in Appendices A and B. Informally, the leftmost leaf in (4), labeled *coupling-nut (zero-def)*, represents the fact that the noun *coupling nut* is used here, in construction with the zero definite determiner characteristic of this genre, to contribute a noun phrase to the sentence. Generally, these elements include lexical items, as *coupling nut* does; but in cases such as the ⟨*purpose*⟩ element, we may instead find some distinctive syntax associated with meaning that could otherwise be realized by a construction with explicit lexical material (*in order*, for a purpose relation). Edges in the tree represent operations of syntactic combination; the child node may either supply a required COMPLEMENT to the parent node (as the node for *coupling-nut* does for its parent *slide*) or may provide an optional MODIFIER that supplements the parent's interpretation (as the node for ⟨*purpose*⟩ does for its parent *slide*). Note that the use of surface structure tree-fragments in LTAG eliminates any need for "abstract" linguistic structures or resources, as in (Meteer, 1991).

[5] Our use of assertion and presupposition reflects the increasingly important role of this distinction in linguistic semantics, in such works as (van der Sandt, 1992; Kamp and Rossdeutscher, 1994); the particular assertions and presuppositions we use draw not only on linguistic theory but also on research in connecting linguistic meanings with independently-motivated domain representations, such as those required for animating human avatars (Badler et al.,

utterance meaning is a conjunction of these atomic contributions. This allows us to to associate a set of logically well-formed propositions with each partial utterance, as we search word-by-word through derivations.

This threefold structure provides a framework for regimenting the semantic information that goes into choices in microplanning. As an illustration, consider the item *slide* as used in (3) and represented in (4). We start with a partial analysis of the assertion. We assume that our domain reasoning requires us to reason in detail only about the kinematics of actions and their order in sequence.[6] Thus, although we may also use coarse representations of force and change-of-state, we can refrain from fine-grained temporal, modal or causal reasoning. Accordingly, for its assertion, we represent *slide* as introducing an event $a1$ in which H (the hearer) will *move* N (the coupling nut) along a path P (from its current location along the surface of the pipe to the elbow); this event is to occur *next* in the maintenance procedure.

We distinguish the verb *slide* from other motion verbs by a presupposition about the spatial layout of the environment. The constraint is that there is a path P which *starts at* the current location of the moved object N, and lies along the *surface* of an object. In modeling these kinematic constraints as presuppositions, we assume that the hearer uses them to identify paths through space from among sensible candidates. In using the verb, the speaker asserts that the path so identified is the path of motion. Concretely, consider (5):

(5) Bill slid the sleeve there.

Just as the word *Bill* provides presupposed information that identifies an individual b from the context, and enables the speaker to assert that b is the agent of the motion, the word *slide* here provides presupposed information that helps identify a particular surface path p from the context, and enables the speaker to assert that p is the trajectory of motion. In such examples, the constraint helps specify what it means for the event to be a sliding, but also helps identify the object moved, its origin and its destination. The formulation of such presuppositions meshes tightly with the articulation of the formal ontology of the domain, which determines the individuals the hearer must identify in understanding an utterance.

As an imperative, *slide* carries a presupposed constraint on who the *participants* in the conversation are, which helps identify the agent H as the hearer, and at the same time introduces a variable for the speaker S. Moreover, when used in the imperative, *slide* carries the pragmatic constraint that S be capable of imposing *obligations* for physical action on H. We formulate such pragmatic constraints in response to specific stylistic choices observed in application language. For example, a maintenance domain might require a context-sensitive choice between an imperative instruction

1999; Badler et al., 2000). Our further specification of pragmatic conditions is inspired by accounts of constructions in discourse in terms of contextual requirements, such as (Hirschberg, 1985; Ward, 1989; Prince, 1986; Birner, 1992; Gundel et al., 1993). This provides both a principled model of syntactic choice and a declarative language for controlling the output of the system to match the choices observed in a given corpus or sublanguage.

[6] Such representational assumptions are justified for specific applications on the basis of conceptual analysis, as typically practiced in knowledge representation (Brachman et al., 1990).

and a declarative formulation with *you should...*. We design pragmatic constraints to capture the preconditions and motivations for these stylistic choices in context.

Together, these conditions can be schematized as in (6), using variables S as a discourse anaphor for the speaker, H for the hearer, N for the coupling nut, and P for the path[7]:

(6) a Assertion: $\mathit{move}(a1, H, N, P) \wedge \mathit{next}(a1)$
b Presupposition: $\mathit{partic}(S, H) \wedge \mathit{start\text{-}at}(P, N) \wedge \mathit{surf}(P)$
c Pragmatics: $\mathit{obl}(S, H)$

Note that these conditions take the form of constraints on the values of variables; we call the variables that appear in such constraints the DISCOURSE ANAPHORS of an element; we call the values those variables take, the element's DISCOURSE REFERENTS.[8] Thus, these conditions formalize how a speaker can use the word *slide* to DESCRIBE specified discourse referents—this helps explain why we see description as central to the problem of sentence planning.

When elements are combined by syntactic operations, the grammar describes both syntactic and semantic relationships among them. Semantic relationships are represented by requiring coreference between the discourse anaphors of combined elements. We illustrate this by considering the element *coupling-nut*, which appears in combination with *slide*. The grammar determines that the element presupposes a coupling nut (*cn*) represented by some discourse anaphor *R*. The pragmatics of the element is the condition that the genre supports the zero definite construction (*zero-genre*) and that the referent for *R* has definite status in the conversational record (Gundel et al., 1993)—that is, that *R* is uniquely identifiable as an individual in the context (and further, that *R* is uniquely identified in the communicative effect that needs to be conveyed, an assumption we invariably make). The element carries no assertion. Thus, this use of *coupling-nut* carries the conditions schematized in (7).

(7) a Assertion: —
b Presupposition: $\mathit{cn}(R)$
c Pragmatics: $\mathit{def}(R) \wedge \mathit{zero\text{-}genre}$

Now, when this element serves as the direct object of the element *slide* as specified in (4), there is a corresponding step of grammatical derivation that incorporates the syntactic tree for *coupling-nut* into the syntactic tree for *slide*. Similar syntactic operations can add elements anywhere into the overall structure of the incomplete sentence. Such steps also add the meaning of the new element into the overall meaning of a sentence. Crucially, semantic combination includes operations of unification. For example, here unification ensures that what is slid must be the coupling nut;

[7] From here on, we adopt the abbreviations *partic* for *participants*, *surf* for *surface*, and *obl* for *obligations*.

[8] Our terminology follows that of (Webber, 1988), where a discourse anaphor specifies an entity by relation (perhaps by an inferential relation) to a referent represented in an evolving model of the discourse. Throughout, we follow the Prolog convention with anaphors–variables in upper case and referents–constants in lower case, but we use lower case for variables that are explicitly bound by quantifiers and other logical operators.

formally, in this case, the *N* of (6) must be the same as the *R* of (7). Applying this constraint, we would represent the conditions imposed jointly by *slide* and *coupling-nut* in combination as in (8). The new material contributed by *coupling-nut* is highlighted in bold.

(8) a Assertion: $move(a1,H,N,P) \wedge next(a1)$
b Presupposition: $partic(S,H) \wedge start\text{-}at(P,N) \wedge surf(P) \wedge \boldsymbol{cn}(\boldsymbol{N})$
c Pragmatics: $obl(S,H) \wedge \boldsymbol{def}(\boldsymbol{N}) \wedge \boldsymbol{zero\text{-}genre}$

Full details on grammar and derivation are given in Sections 3 and 4.

Let us now return to instruction (3).

(3) Slide coupling nut onto elbow to uncover fuel-line sealing ring.

Based on the capabilities of our underlying domain representation and the expressive requirements of application language, we have formulated the constraints in (9) to represent the meaning of (3).

(9) a Assertion: $move(a1,H,N,P) \wedge next(a1) \wedge purpose(a1,a2) \wedge uncover(a2,H,R)$
b Presupposition: $partic(S,H) \wedge start\text{-}at(P,N) \wedge surf(P) \wedge cn(N) \wedge end\text{-}on(P,E) \wedge el(E) \wedge$
$sr(R) \wedge fl(F) \wedge nn(R,F,X)$
c Pragmatics: $obl(S,H) \wedge def(N) \wedge def(E) \wedge def(R) \wedge def(F) \wedge zero\text{-}genre$

Each of these constraints is contributed by an appropriate lexical entry, including requirements for suitable unification operations, so that (9) can be derived automatically and incrementally. Spelling out the example in more detail, we see that in addition to the asserted constraints *move* and *next* contributed by the element *slide*, we have a *purpose* constraint contributed by the infinitival adjunct and an *uncover* constraint contributed by the element *uncover*; in addition to the presupposed constraints *partic*, *start-at*, *surf* and *cn* contributed by *slide* and *coupling-nut,* we have an *end-on* constraint contributed by *onto,* an *el* constraint contributed by *elbow*, an *sr* constraint contributed by *sealing-ring* and *fl* and *nn* constraints contributed by the noun-noun modifier use of *fuel-line*; *nn* uses a variable *X* to abstract some close relationship between the fuel-line concept *F* and the sealing ring *R* which grounds the noun-noun compound.

In any use of an utterance like (3), the speaker intends the presupposition and the pragmatics of the utterance to link up in a specific way with particular individuals and propositions from the conversational record; the speaker likewise intends the assertion to settle particular open questions in the discourse in virtue of the information it presents about particular individuals. These links constitute the PRESUMPTIONS the speaker bases an utterance on; these presumptions must be recorded in an interpretation over and above the shared conventions that we have already outlined. We assume that these links take the form of INFERENCES that the speaker is committed to in generation and that the hearer must recover in understanding.[9]

[9] These inferences are part of the REPRESENTATION of interpretation; they characterize the argument that lies behind the speaker's utterance. Hence this notion of inference has to be distinguished from the REASONING that speaker

We return to the element *slide* of (4) to illustrate these inferences. Let's adopt the notation that a boxed expression represents an update to be made to the conversational record, while an underlined expression represents a feature already present in the conversational record; boxed and underlined expressions are DOMAIN representations and thus can refer to the elements of application-specific ontologies and models. The other expressions we have seen are LINGUISTIC representations, since they are associated with lexical items and syntactic constructions in a general way. An edge indicates an inferential connection between a linguistic representation and a domain representation. Then we can provide representations of the presumption associated with the assertion of *slide* in (3) by (10).

(10)
$$\begin{tikzcd}
\boxed{move(a1,h0,n11,p(l(on,j2),l(on,e2)))} \arrow[d, dash] & \boxed{next(a1)} \arrow[d, dash] \\
move(a1,H,N,P) & next(a1)
\end{tikzcd}$$

In this inference, we take the speaker of (3) to be a computer system (including an NLG component), which represents itself as a conversational participant $s0$ and represents its user as a conversational participant $h0$. We suppose that the coupling nut to be moved here is identified as $n11$ in the system's model of the aircraft, the fuel-line joint is identified as $j2$ and the elbow is identified as $e2$. In order to describe paths, we use a function l which picks out a place using a landmark and a qualitative placement drawn from a predefined spatial ontology. For example, $l(on,e2)$ is the place *on the elbow*. We also use a function p whose arguments are two places and whose result is the direct path between them. Thus, $p(l(on,j2),l(on,e2))$ is the path that the coupling nut follows here. (For a similar spatial ontology, see (Palmer, 1990) or (Jackendoff, 1990).) Then the system here intends the contribution that the next action, $a1$, is one where $h0$ moves $n11$ by path $p(l(on,j2),l(on,e2))$. Deriving this contribution involves instantiating the meaning of *slide* for the particular discourse referents from the conversational record the speaker intends to describe.

Given what we have supposed, in uttering (3), the system is also committed to inferences which establish instances of the presupposition and the pragmatics of *slide* for appropriate referents. Our conventions represent the further inferences for the presupposition as in (11a) and those for the pragmatics as in (11b).

(11) a
$$\begin{tikzcd}
partic(S,H) \arrow[d, dash] & start\text{-}at(P,N) \arrow[d, dash] & surf(P) \arrow[d, dash] \\
\underline{partic(s0,h0)} & \underline{start\text{-}at(p(l(on,j2),l(on,e2)),n11)} & \underline{surf(p(l(on,j2),l(on,e2)))}
\end{tikzcd}$$

b
$$\begin{tikzcd}
obl(S,H) \arrow[d, dash] \\
\underline{obl(s0,h0)}
\end{tikzcd}$$

or hearer may do to construct or recognize an interpretation. For example, the speaker's presumptions might include a deductive inference that a description fits its intended referent. By contrast, the hearer's reasoning in understanding would aim to identify this presumption (represented as an inference). Since this reasoning aims to discover the best explanation of the speaker's use of the utterance, this reasoning should be abductive rather than deductive in character.

In (11), we use the same predicates for domain and linguistic relationships, so the inferences required in all cases can be performed by simple unification. This uses the same values for variables as figure in the inferences from the assertion—discourse anaphor S is unified with discourse referent $s0$ for the speaker; H with *h0* for the hearer; P with $p(l(on, j2), l(on, e2)), n11)$; and N with the coupling nut $n11$.

Our framework will also enable more complicated (and more substantive) connections. For example, suppose we use a predicate $loc(L, O)$ to indicate that the place L is the location of object O. Then we would represent the fact that the nut is located on the joint as (12).

(12) $loc(l(on, j2), n11)$

We know that if an object o is in some place l, then any path from l starts at o; (13) formalizes this generalization.

(13) $\forall loe(loc(l, o) \supset start\text{-}at(p(l, e), o))$

Since they provide common background about this equipment and about spatial action in general, both of these facts belong in the conversational record.[10]

From (12) and (13) we can infer that the path on the joint starts at the nut; that leads to a record of inference as in (14).

(14) $$\begin{array}{c} start\text{-}at(P,N) \\ | \\ \underline{loc(l(on, j2), n11)} \end{array}$$

That is, the understanding behind (14) is that $loc(l(on, j2), n11)$ is a fact from the conversational record intended to be linked with the linguistic presupposition $start\text{-}at(P,N)$ by appeal to premise (13) from the conversational record.

Similarly, we propose to analyze the modifier *fuel-line* in keeping with the inferential account of noun-noun compounds proposed in (Hobbs et al., 1988; Hobbs et al., 1993). This item carries a very general linguistic presupposition. F must be a fuel line or fuel lines in general, and there must be some close relationship X between F and the object R that the modifier applies to. In the context of this aircraft, this presupposition is met because of the fact that the particular ring intended here is designed *for* this particular fuel line: $X = for$. This link exploits a domain-specific inference rule to the effect that one thing's being designed for another counts as the right kind of close relationship for noun-noun modification. Concretely, we might use this structure to abstract the inference:

(15) $$\begin{array}{c} nn(R,F,X) \\ | \\ \underline{for(r11, f4)} \end{array}$$

[10] Recall that we are using the term conversational record to refer to all the information that interlocutors take as uncontroversial, not just what has been explicitly mentioned.

Assert:
links for utterance assertion

Presuppose:
links for utterance presupposition

Pragmatics:
links for utterance pragmatic conditions

Structure:
dependency representation of the utterance

Figure 4: General form of communicative intent representation.

As with (14), (15) represents that $for(r11, f4)$ is a shared fact linked with the linguistic presupposition $nn(R,F,X)$ by appeal to a shared rule, here (16).

(16) $\forall ab(for(a,b) \supset nn(a,b,for))$

In general, then, the communicative intent behind an utterance must include three inferential records.

- The first collection of inferences links the **assertions** contributed by utterance elements to updates to the conversational record that the instruction is intended to achieve; in the case of (10), we add instances of the assertion identified by the speaker.
- The second collection of inferences links the **presuppositions** contributed by the utterance elements to intended instances in the conversational record.
- The final collection of inferences links the **pragmatics** of the utterance elements to intended instances in the conversational record.

We will represent such inferences along with the linguistic structure of an utterance in the format of Figure 4. Reading Figure 4 from bottom to top, we find a version of Clark's ladder of intentions involved in language use, with higher levels dependent on lower ones. The speaker intends to use particular linguistic words and constructions. The speaker intends to refer to particular shared individuals from the context. The speaker intends to contribute specific information to the context, and thereby to address particular open questions. Identifying the speaker's uses of words is a prerequisite for further reasoning. The inferences to pragmatics and presupposition are prerequisites for recognizing what is asserted, while the inferences from the assertion in turn contingently determine the contribution of the utterance. Such diagrams constitute a complete record of communicative intent, since they include the linguistic structure of the utterance and lay out the conventional meanings assigned to this structure as well as the presumed inferences linking these meanings to context. For example, Figure 5 displays the communicative intent associated with the utterance of *slide*.

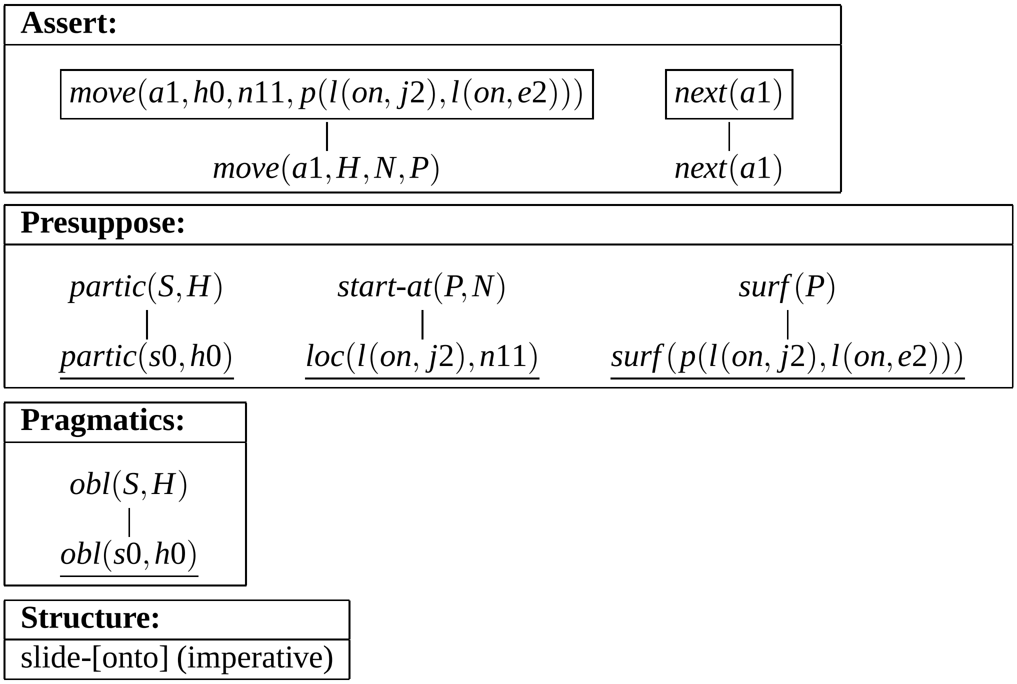


Figure 5: Interpretation of *slide* in (3). The speaker's presumptions map out intended connections to discourse referents as follows: the speaker S, $s0$; the hearer H, $h0$; the nut N, $n11$; the path P, $p(l(on, j2), l(on, e2))$; the elbow E, $e2$. The fuel-line joint is $j2$.

Figure 6 schematizes the full communicative intent for (3) using the notational conventions articulated thus far. As a whole, the utterance carries the syntactic structure of (4); in Figure 6 this structure is paired with inferential representations that simply group together the inferences involved in interpreting the individual words in their specific syntactic constructions.

2.3 Communicative-Intent–Based Microplanning in SPUD

Sections 2.1–2.2 have characterized microplanning as a problem of constructing representations of communicative intent to realize communicative goals. Communicative intent is a detailed representation of an utterance made up of inferences constructed from a declarative description of language, the grammar, and from a declarative description of context, the conversational record. By setting up appropriate microplanning choices and providing the means to make them, this representation reconciles the decision-making required for microplanning tasks like lexical choice, referring expression generation and aggregation.

A key further contribution of our research is the integration of a suite of assumptions and techniques for effective implementation and development of communicative-intent–based microplanners. These principles are realized in our implemented system SPUD.

- We use a logic-programming strategy to link linguistic meanings with specifications of the conversational record and updates to it. We base our specification language on modal logic

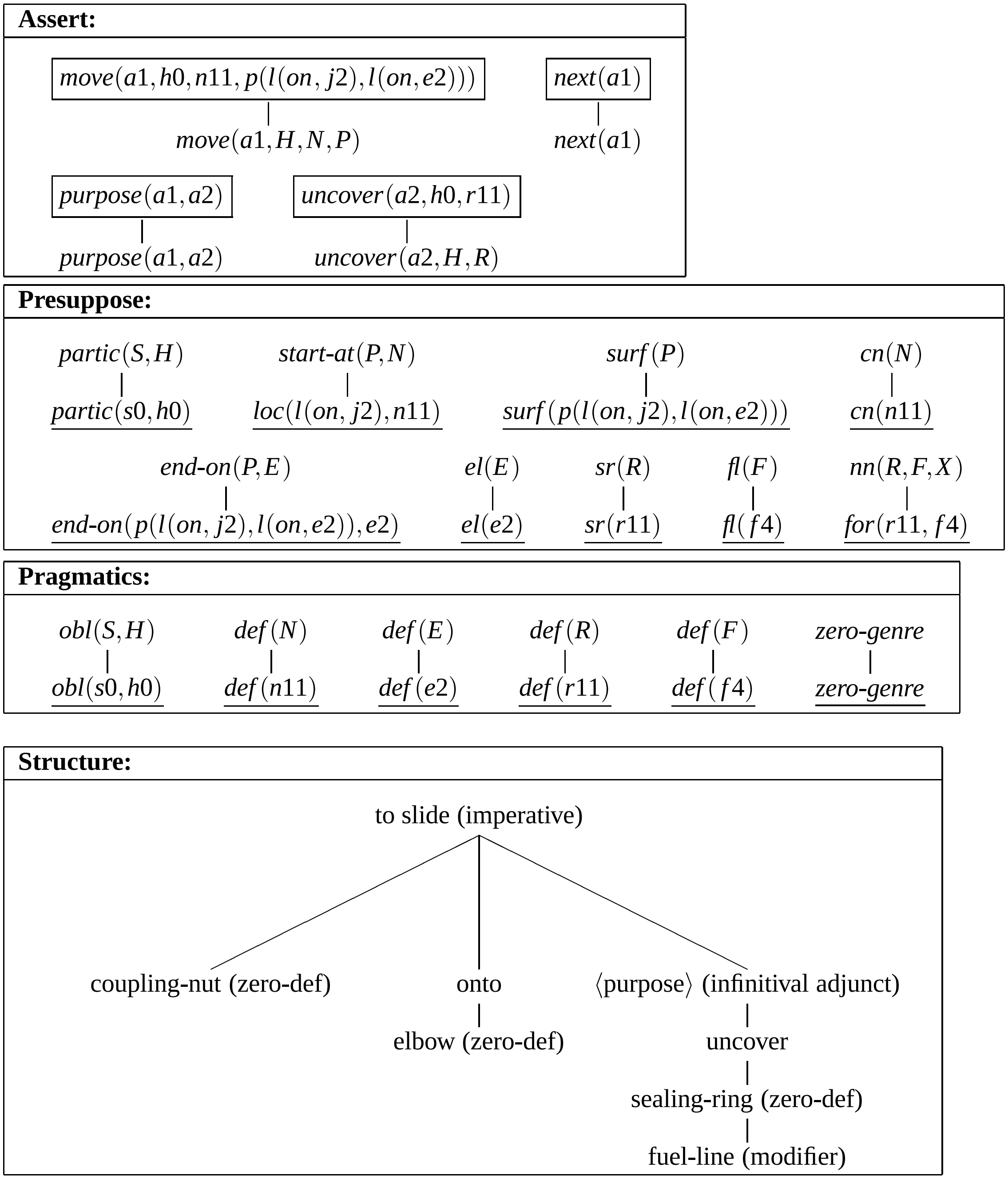


Figure 6: Communicative intent for (3). The grammar specifies meanings as follows: For *slide*, assertions *move* and *next*; for the infinitival adjunct, *purpose*; for *uncover*, *uncover*. For *slide*, presuppositions *partic*, *start-at* and *surf*; for *coupling-nut*, *cn*; for *onto*, *end-on*; for *elbow*, *el*; for *sealing-ring*, *cn*; for *fuel-line*, *fl* and *nn*. For *slide*, pragmatics *obl*; for other nouns, pragmatics *def* and *zero-genre*. The speaker's presumptions map out intended connections to discourse referents as follows: the speaker S, $s0$; the hearer H, $h0$; the nut N, $n11$; the path P, $p(l(on, j2), l(on, e2))$; the elbow E, $e2$; the ring R, $r11$; the fuel-line F, $f4$; the relation X, *for*. The fuel-line joint is $j2$.

in order to describe the different states of information in the context explicitly (Stone, 1999; Stone, 2000b); however, the logic programming inference ensures that a designer can assess and improve the computational cost of the queries involved in constructing communicative intent.

- By treating presuppositions as anaphors (cf. (van der Sandt, 1992)), we carry over efficient constraint-satisfaction techniques for managing ambiguity in referring expressions from prior generation research (Mellish, 1985; Haddock, 1989; Dale and Haddock, 1991).

- We adopt a head-first, greedy search strategy. Our other principles are compatible with searching among all partial representations of communicative intent, in any order. But a head-first strategy allows for a particularly clean implementation of grammatical operations; and the modest effort required to design specifications for greedy search is easily repaid by improved system performance.

We now return to our initial characterization of microplanning as a complex collaborative and deliberative process, guided by progress towards constructing representations of communicative intent such as that of Figure 6. We can now confirm precisely that these provisional representations of interpretation, as implemented in SPUD, provide a comprehensive resource for the decisions of microplanning.

Each provisional representation of interpretation includes a collection of instantiated grammatical elements with a particular intended interpretation, as sketched in Section 2.2. The provisional status of these intermediate stages of generation is reflected in two further records.

- First, intermediate stages are provisional in that the hearer may not be able to disambiguate the utterance and identify this intended interpretation. To summarize the alternative possible interpretations, provisional interpretations include a constraint network that represents all the possible ways of instantiating the presupposition of the utterance in the context. The interpretation counts as disambiguated when the intended reference is the only solution to this constraint network.

- Second, intermediate stages are provisional in that a partial utterance may not convey all the required communicative effects. Thus, provisional interpretations also list the communicative effects that remain to be conveyed. The interpretation is complete once this list is empty.

SPUD starts with a task set by a higher-level module, such as a content planner or dialogue manager. In (3), the task is to formulate an utterance that will contribute, in a recognizable way, the updates that a *move* is *next* and its *purpose* is to *uncover*. SPUD sees to it that its utterance satisfies these requirements by adding instantiated elements with a particular intended interpretation, such as the structure for *slide* of Figure 5, one at a time, to a provisional communicative-intent representation. When the interpretation is disambiguated and complete, SPUD returns the result—its recognizable communicative intent.

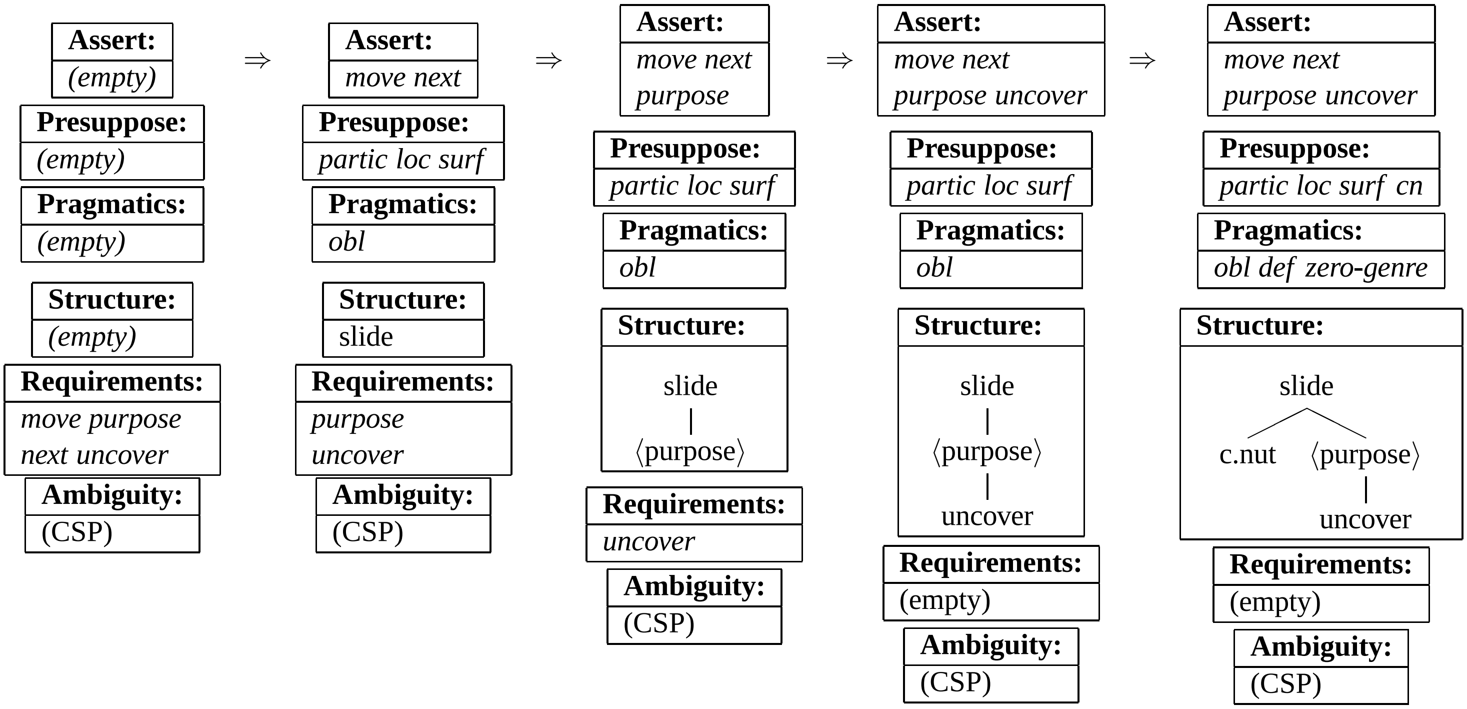


Figure 7: A schematic view of the initial stages of microplanning for (3). Each state includes a provisional communicative intent and an assessment of further work required, such as updates to achieve. Each transition represents the addition of a new interpreted element.

Figure 7 offers a schematic illustration of a few such steps: it tracks the addition first of *slide*, then of a purpose adjunct, then of *uncover*, and finally of *coupling-nut*, all to an initially empty structure. (Note that in Figure 7 we abbreviate inference structures and specified updates to the predicates they establish; we use the tag *CSP* as a mnemonic that the microplanner uses a constraint satisfaction problem to represent whether the structure can be recognized as intended, or whether it has a family of alternative interpretations in context.)

To start, the first transition in Figure 7, which results in a structure that repeats Figure 5, can be viewed as a description of the use of the particular word *slide* in a particular syntactic construction to achieve particular effects. We will see SPUD creates such descriptions by an inferential matching process that checks a pattern of lexical meaning against the discourse context and against the specified updates. In particular, to be applicable at a specific stage of generation, a lexical item must have an interpretation to contribute: the item's assertion must hold; the item's presupposition and pragmatics must find links in the conversational record. Moreover, to be preferred over alternative options, use of the item should push the generation task forward: in general, the updates the item achieves should include as many as possible of those specified in the microplanning problem, and as few others as possible; in general, the links the item establishes to shared context should appeal to specific shared content that facilitates the hearer's understanding process.

Thus, in deriving structures like that of Figure 5 from its grammatical inventory, the generator can implement a model of lexical and grammatical choice. The generator determines available options by inference and selects among alternatives by comparing interpretations.

Meanwhile, in extending provisional communicative intent as suggested in Figure 7, SPUD's further lexical and syntactic choices can simultaneously reflect its strategies for aggregation and for referring expression generation. Take the addition of an element like the infinitival purpose clause, in step two of Figure 7. As with *slide*, this entry represents a pattern of interpretation where linguistic meaning mediates between the current context and potential updates to the context. In particular, the entry for a infinitival purpose clause depends on an event $a1$ with an agent $h0$ already described by the main verb of the provisional instruction (in this case *slide*). The entry relates $a1$ to another event $a2$ which $a1$ should achieve and which also has $h0$ as the agent; here $a2$ is to be described as an *uncovering* by a subsequent step of lexical choice. Thus the syntax and semantics of the entry amount to a pattern for aggregation: the modifier provides a way of extending an utterance that the generator can use to include additional related information about referents already described in the ongoing utterance.

As another illustration, take the addition of a complement like *coupling nut*, as in step four of Figure 7, or a modifier like *fuel-line*. The contribution of these entries is to add constraints on the context that the hearer must match to interpret the utterance. With *coupling nut*, for example, the hearer learns that the referent for N must actually be a coupling nut; similarly, with *fuel-line*, the hearer learns that the referent for R must be for some fuel line F. Here we find the usual means for ensuring reference in NLG: augmenting the content of an utterance by additional presupposed relationships.

2.4 Applications of SPUD

SPUD has provided the computational infrastructure for several investigations of the grammatical basis of choice in NLG. In Section 7, we describe our own work in modeling the choice of verbs in maintenance instructions by suitably representing their syntax and semantics.

In her dissertation (Bourne, 1999), Bourne investigates the linguistic choices in instructions that tell a hearer when to stop a required activity. This information can be realized in many different ways—in the choice of verb, in the selection of a prepositional phrase describing path or duration, with a purpose clause, or with a subordinate clause headed by a conjunction such as *until*. Bourne uses SPUD to develop a declarative description of the information an actor needs to complete an instruction, the information expressed by different linguistic constructions, and the contextual background brought to the instruction by users with different expertise. Her specification allows SPUD to carry out the context-sensitive generation of a short procedure. A representative example of Bourne's instructions is (17).

(17) Reset pump by moving lever to RESET position and then holding lever until light is green.

The expression of termination conditions in this procedure accords with Bourne's corpus analysis of instructions across a range of domains. Bourne's experiments involve a knowledge base of several hundred facts and a grammar of some sixty words and syntactic constructions; her generation instances involve multiple communicative goals (six for example (17)) which are achieved through integrating substantive domain and linguistic inference. She reports generation times in the minutes (on a 1998 workstation).

In his master's thesis (Yan, 2000), Yan investigates the allocation of content between speech and gesture. Yan uses SPUD to develop a declarative description of the syntactic and pragmatic contexts in which speakers produce gestures in coordination with speech, and a declarative description of the choices speakers can make to convey information in gesture. His specification allows SPUD to construct descriptions of embodied utterances, including both speech and gesture, to describe houses. (18) shows a representative response the system can give; the utterance continues the system's description of the entry hall of a house.

(18) There is a staircase in the middle of it.
realized with right hand moving upward in a spiral pointing gesture

Again, the expression of information across speech and gesture in his results accords with his corpus analysis of the spontaneous descriptions produced by human speakers. Yan uses a knowledge base of a few hundred facts and a grammar of about one hundred lexical and syntactic entries. Yan's formalizations avoid deep inference in establishing presuppositions or achieving communicative goals, so typical generation times in this case are fractions of a minute (on a 2000 desktop machine). In fact, SPUD is used in the REA system, alongside canned text and templates, to generate utterances in an end-to-end interactive conversational agent. See (Cassell, 2000; Cassell et al., 2000).

2.5 Roadmap

In the remainder of this paper, we provide complete details of our principles for design, implementation, and knowledge representation of communicative-intent–based microplanning. We first describe the grammar formalism we have developed (Section 3) and the model of interpretation that associates grammatical structures declaratively with possible communicative intent (Section 4). We then introduce the SPUD sentence planner as a program that searches (greedily) through grammatical structures to derive a communicative intent representation that describes a desired update to the conversational record and that can be recognized by the hearer (Section 5). This material provides a complete specification of an implementation of a microplanner, and thus serves as a benchmark and guide for future implementations.

We then go on to illustrate how SPUD's declarative processing provides a natural framework for addressing sentence planning subtasks like referring expression generation, lexical and syntactic choice and aggregation (Section 6). This material further justifies our conception of microplanning as a uniform problem and thus serves to contrast our approach with the specialized resources, representations and algorithms of alternative microplanning processes. We then show how SPUD's declarative processing supports a concrete methodology for building grammatical resources for specific generation problems (Section 7). This serves as an introduction to the technical practice of building declarative grammars for microplanning, and highlights the commonalities and differences between this practice and established methodologies for grammar-building and knowledge representation.

3 Grammar Organization

In SPUD, a grammar consists of a set of SYNTACTIC CONSTRUCTIONS, a set of LEXICAL ENTRIES, and a database of MORPHOLOGICAL RULES. Through these specifications, SPUD implements a standard lexicalized TAG syntactic formalism, much as in the Xtag system (Doran et al., 1994; The XTAG-Group, 1995), and couples this syntax with an "ontologically promiscuous" descriptive semantics (Hobbs, 1985). Note however that SPUD does not itself implement sophisticated functionality for developing and organizing these specifications. For example, SPUD grammars cannot represent commonalities across families of trees or across families of lexical items; this makes them needlessly repetitive. We expect such limitations of the current system to diminish with future research.

3.1 Syntactic Constructions

Syntactic constructions are specified by four components in SPUD:

(19) a a NAME, an identifier under which other parts of the grammar refer to the construction;
b a set of PARAMETERS, open variables for referential indices in the definition (which are instantiated to discourse referents in a particular use of the construction);
c a PRAGMATIC CONDITION, which expresses a constraint that the construction imposes on the discourse context in terms of its parameters; and
d a SYNTACTIC STRUCTURE, which maps out the linguistic form of the construction.

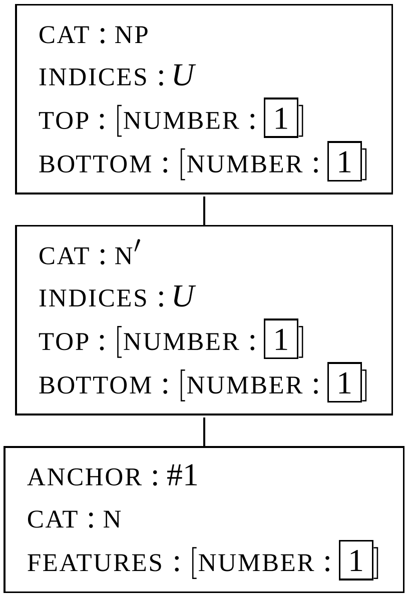
The syntactic structure is represented as a tree of compound nodes. Internal nodes in the tree bear the following attributes:

(20) a a CATEGORY, such as NP, V, etc.;
b INDICES, a list of the parameters that the node refers to and that additional syntactic material combined with this node may describe;
c a TOP FEATURE STRUCTURE, a list of attribute-value pairs (including variable values shared with other feature structures elsewhere in the tree) which describes the syntactic constraints imposed on this node from above; and
d a BOTTOM FEATURE STRUCTURE, another such list of attribute-value pairs which describes the syntactic constraints imposed on this node from below.

Leaves in the tree fall into one of four classes: SUBSTITUTION SITES, FOOT NODES, TERMINAL NODES and lexically-dependent word nodes or ANCHOR NODES. Like internal nodes, substitution sites and foot nodes are loci of syntactic operations and are associated with categories and indices. Any tree may have at most one foot node, and that foot node must have the same category and indices as the root. A terminal node includes a specific lexeme (typically a closed-class or function item) which appears explicitly in all uses of the construction. An anchor node is associated with an instruction to include a word retrieved from a specific lexical entry; trees may have multiple anchors and lexical entries may contain multiple words. In addition, all leaves are specified with a single feature structure which describes the constraints imposed on the node from above. Note that, in the case of anchor nodes, these constraints must be satisfied by the lexical items retrieved for the node.

(21) shows the tree structure for the zero definite noun phrase required in (3) for *coupling nut* and *sealing ring*.

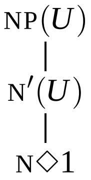
(21)


CAT : NP
INDICES : U
TOP : [NUMBER : $\boxed{1}$]
BOTTOM : [NUMBER : $\boxed{1}$]
CAT : N′
INDICES : U
TOP : [NUMBER : $\boxed{1}$]
BOTTOM : [NUMBER : $\boxed{1}$]
ANCHOR : #1
CAT : N
FEATURES : [NUMBER : $\boxed{1}$]

As (21) illustrates, the full specifications associated with such structures can be quite involved. For exposition, henceforth we will generally suppress feature structures. We will write internal nodes in the form CAT(INDICES); anchors, in the form CAT◇N (for the Nth token of a lexical item, a word of category CAT); substitution nodes, in the form CAT(INDICES) ↓; foot nodes, in the form CAT(INDICES)∗; and terminal nodes just by the words associated with them.

With these conventions, the syntactic entry for the zero definite construction (associated with *sealing-ring* for example) is given in (22).

(22) a NAME: zerodefnptree
b PARAMETERS: U
c PRAGMATICS: *zero-genre* ∧ *def*(U)
d TREE:



Observe that (21) appears simply as (22d).

3.2 Lexical Entries

SPUD lexical entries have the following structure.

(23) a a NAME, a list of the lexemes that anchor the entry (most entries have only one lexeme, but entries for idioms may have several);

b a set of PARAMETERS, open variables for referential indices in the definition (which are instantiated to discourse referents in a particular use of the entry);

c a TARGET, an expression constraining the category and indices of the node in a syntactic structure at which this lexical entry could be incorporated, and indicating whether the entry is added as a complement or as a modifier;

d a CONTENT CONDITION, a formula specifying a constraint on the parameters that the entry will assert when used to update the conversational record (note however that the content condition does not HAVE to be asserted with every use of the lexical item; sometimes the content condition may be presupposed);

e PRESUPPOSITION, a formula specifying a constraint on the parameters of the entry that the entry must presuppose;

f PRAGMATICS, a formula specifying a constraint on the status in the discourse of parameters of the entry;

g an ANCHORING FEATURE STRUCTURE, a list of attribute-value pairs that constrain the anchor nodes where lexical material from this entry is inserted into a syntactic construction; and

h a TREE LIST, specifying the trees that the lexical item can anchor by name and parameters (note that the tree list in fact determines what the target of the entry must be; we specify the target separately merely for convenience).

(24) gives an example of such a lexical item: the entry for *sealing-ring* as used, among other ways, with the zero definite noun phrase illustrated in (22).

(24) a NAME: sealing-ring

b PARAMETERS: N

c TARGET: NP(N) [complement]

d CONTENT: $sr(N)$

e PRESUPPOSITION: —

f PRAGMATICS: —

g ANCHOR FEATURES: $[\text{NUMBER} : \text{SINGULAR}]$

h TREE LIST: zerodefnptree$(N), \ldots$

3.3 Lexico-grammar

The basic elements of grammatical derivations are lexical entries used in specific syntactic constructions. These elements are declarative combinations of the two kinds of specifications presented in Sections 3.1 and 3.2. Abstractly, the combination of a lexical entry and a syntactic construction requires the following steps.

(25) a The parameters of the lexical entry are instantiated to suitable discourse referents.

b The parameters of the construction are instantiated to discourse referents as specified by the tree list of the lexical entry.

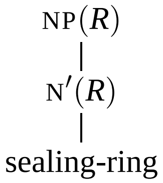
c Anchor nodes in the tree are replaced by corresponding terminal nodes constructed from the name of the lexical entry; and the top feature structures of anchor nodes are unified with the anchor features of the lexical entry to give the top features of the new terminal nodes.

d The assertion and the presupposition of the combined entry are determined, in one of two possible ways. In one possible case, the content condition of the lexical entry provides the assertion while the presupposition of the lexical entry provides the presupposition of the combined element. In the other, the content condition and any presupposition of the lexical entry are conjoined to give the presupposition of the combined element; in this case the element carries no assertion.[11]

e The pragmatics of the syntactic construction is conjoined with the pragmatics of the lexical entry.

Thus, abstractly, we can see the syntactic construction of (22) coming together with the lexical entry of (24) to yield the particular lexico-grammatical option described in (26).

(26) a TREE:



b ASSERTION: —

c PRESUPPOSITION: $sr(R)$

d PRAGMATICS: $def(R) \wedge zero\text{-}genre$

(Again, feature structures are suppressed here, but note that feature sharing ensures that each of the nodes in the tree is in fact marked with singular number.) This is the entry for *sealing-ring* which is used in deriving the communicative intent of Figure 6.

3.4 Morphological rules

We have seen that lexico-grammatical entries such as (26) contain not specific surface word-forms but merely lexemes labeled with features. This allows feature-values to be propagated through grammatical derivations. In this way, the derivation can select an appropriate realization for an underlying lexeme as a function of agreement processes in the language.

[11] In our current implementation, syntactic features can control which case applies, but we have found that this control has little impact on generation and is cumbersome to use. In most contexts only one case will make for a good microplanning move, depending on whether the content is in fact shared information or private information; so we can typically entertain both possibilities in search.

A database of morphological rules accomplishes this selection. This database is quite primitive. Each lexeme is simply paired with a list of feature-realization patterns. To determine the form to use in realizing a given lexeme at a node with given features F in a grammatical derivation, SPUD scans this list until the feature structure in a pattern subsumes F; SPUD uses the realization associated with this pattern.

For example, we might use (27) to determine the realization of *sealing-ring* in (26) as "sealing ring".

(27) a LEXEME: sealing-ring; PATTERNS:
b $[\text{NUMBER} : \text{SINGULAR}] \rightarrow$ sealing ring
c $[\text{NUMBER} : \text{PLURAL}] \rightarrow$ sealing rings

(Recall that the singular syntactic number feature here will be supplied by the lexical entry for *sealing ring* at the same stage at which the singular semantics is selected. In other cases, the morphological database accommodates agreement which is mediated by purely syntactic processes.)

4 Grammatical Derivation and Communicative Intent

To assemble communicative intent, SPUD deploys lexico-grammatical entries like (26) one by one, as depicted in Figure 7. As Section 2 suggested, these steps involve both grammatical inference to link linguistic structures together and contextual inference to link linguistic meanings to domain-specific representations. We now describe the specific form of these inferential processes in SPUD.

4.1 Grammatical Inference

In SPUD's grammar, the trees of entries like (26) describe a set of elementary structures for a feature-based lexicalized tree-adjoining grammar, or LTAG (Joshi et al., 1975; Vijay-Shanker, 1987; Schabes, 1990). In all TAG formalisms, entries can be combined into larger trees by two operations, called SUBSTITUTION and ADJOINING. Elementary trees without foot nodes are called INITIAL trees and can only substitute; trees with foot nodes are called AUXILIARY trees, and can only adjoin. The trees that these operations yield are called DERIVED trees; we regard the computation of derived trees as an inference about a complex structure that follows from a declarative specification of elementary structures. In a grammar with features, derived trees are completed by unifying the top and bottom features on each node.

In substitution, the root of an initial tree is identified with a leaf of another elementary or derived structure, called the SUBSTITUTION site. The top feature structure of the substitution site is unified with the top feature structure of the root of the initial tree. Figure 8 schematizes this operation.

Adjoining is a more complicated splicing operation, where an elementary structure DISPLACES some subtree of another elementary or derived structure. The node in this structure where the replacement applies is called the ADJUNCTION SITE; the excised subtree is then substituted back into the first tree at the distinguished FOOT node. As part of an adjoining operation, the top feature structure of the adjunction site is unified with the top feature structure of the root node of the

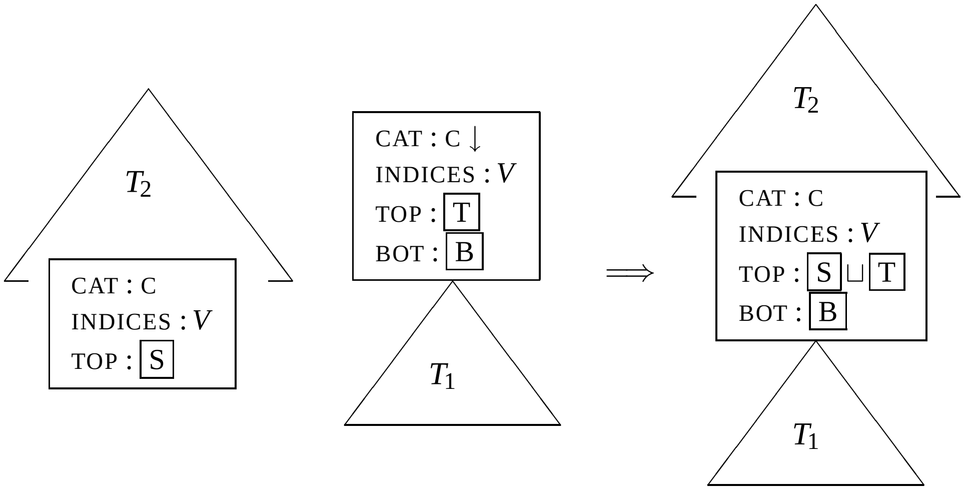


Figure 8: Substitution of T_1 into T_2.

auxiliary tree; the bottom feature structure of the adjunction site is unified with the feature structure of the foot node. After an adjoining operation, no further adjoining is possible at the foot node. This is schematized in Figure 9.

In substitution, the substitution site and the root node of the substituted tree must have the same category; likewise, in adjoining, the root node, the foot node and the adjunction site must all have the same category. Moreover, as our trees incorporate indices labeling the nodes, there is the further requirement that any nodes that are identified through substitution or adjoining must carry identical indices. In a Prolog implementation, these identities can be realized directly through unification.

The unification of indices determines the interface between syntax and semantics in SPUD. SPUD adopts an ontologically promiscuous semantics (Hobbs, 1985), in the sense that each entry used in the derivation of an utterance contributes a constraint to its overall semantics. Syntax determines when the constraints contributed by different grammatical entries describe the same variables or discourse anaphors. For example, take the phrase *slide the sleeve quickly*. Its lexical elements contribute constraints describing an event e in which agent x *slides* object y along path p; describing an individual z that is a *sleeve*; and describing an event e' that is *quick*. The syntax–semantics interface provides the guarantee that $y = z$ and $e = e'$ (i.e., that the sleeve is what is slid and that the sliding is what is quick). It does so by requiring that the index y of the object NP substitution site of *slide* unify with the index z of the root NP for *sleeve*, and by requiring that the index e of the VP adjunction site for *slide* unify with the index e' of the VP foot node for *quickly*. (See (Hobbs, 1985; Hobbs et al., 1993) for more details on ontologically promiscuous semantics.)

Note that this strategy contrasts with other approaches to LTAG semantics, such as (Candito and Kahane, 1998), which describe meanings primarily in terms of function-argument relations. (It is also possible to combine both function-argument and constraint semantics, as in (Joshi and Vijay-Shanker, 1999; Kallmeyer and Joshi, 1999).) Like Hobbs, we use semantic representations as a springboard to explore the relationships between sentence meaning, background knowledge

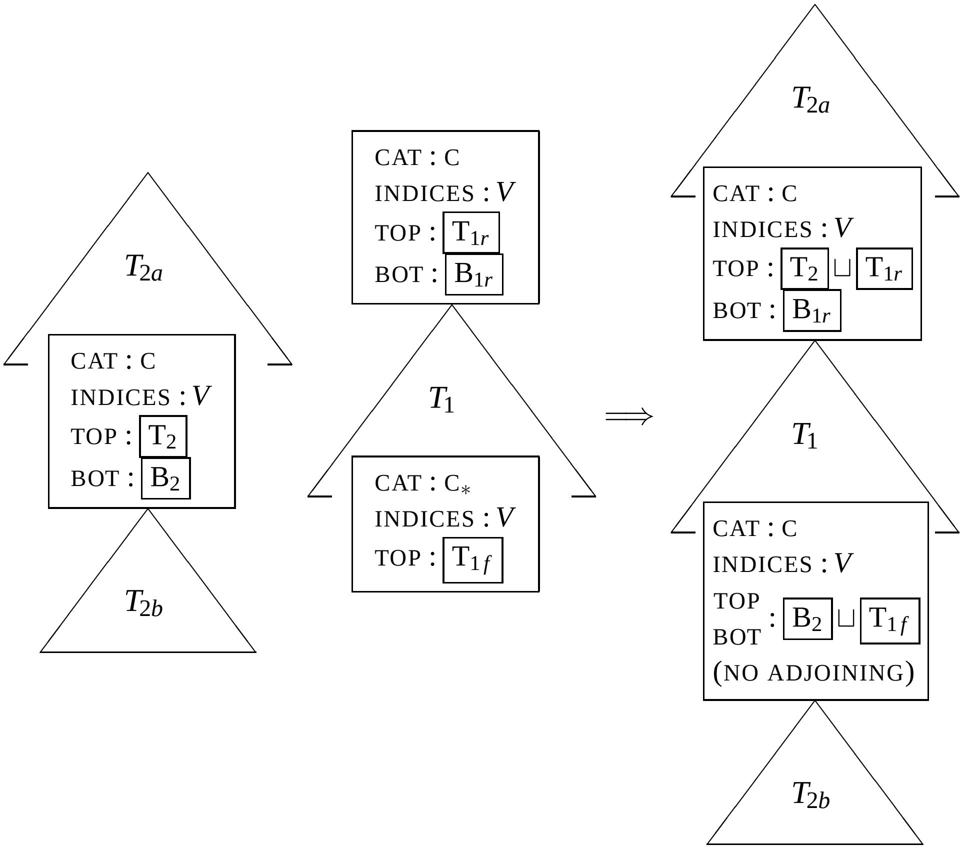
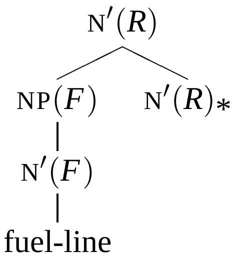


Figure 9: Adjunction of T_1 into T_2

and inference. These relationships are easiest to state in terms of constraints because they allow us to devise well-formed formulas that capture the assertions, presuppositions and pragmatics of incomplete utterances. In addition, the use of constraints harmonizes with our perspective that the essential microplanning task is to construct extended descriptions of individuals (Stone and Webber, 1998; Webber et al., 1999).

Let us illustrate the operations of grammatical inference by describing how the structure for *fuel-line* can combine with the structure for *sealing ring* by adjoining. *Fuel-line* will be associated with a combined lexico-syntactic realization as in (28).

(28) a TREE:



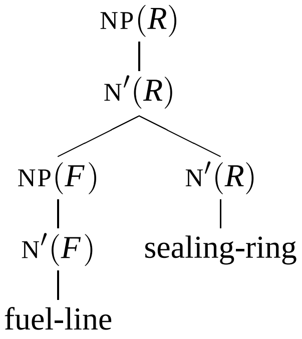
b ASSERTION: —

c PRESUPPOSITION: $fl(F) \wedge nn(R,F,X)$

d PRAGMATICS: $def(F)$

We can adjoin (28a) into (26a) using the N′(R) node as the adjunction site, to obtain the structure in (29).

(29)



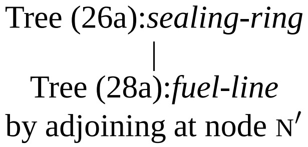
When we put together entries by TAG operations, we can represent the meaning of the combined structure as the component-wise conjunction of the meanings of its constituents. In the case of (26) and (28) this would yield:

(30) a ASSERTION: —
b PRESUPPOSITION: $sr(R) \wedge fl(F) \wedge nn(R,F,X)$
c PRAGMATICS: $def(R) \wedge zero\text{-}genre \wedge def(F)$

(As explained in the next section, we can also directly describe the joint interpretation of combined elements, in terms of intended links to the conversational record and intended updates to it.)

In addition to explicitly setting out the structure of a TAG derived tree as in (29), we can also describe a derived tree implicitly in terms of operations of substitution and adjoining which generate the derived tree. Such a description is called a TAG DERIVATION TREE (see (Vijay-Shanker, 1987) for a formal definition and discussion of TAG derivation trees). Each node in a derivation tree represents an elementary tree that contributes to the derived tree. Each edge in a derivation tree specifies a mode of combination: the child node is combined to the parent node by a specified TAG operation at a specified node in the structure. For example, (31) shows the derivation tree corresponding to (29).

(31)



Derivation trees indicate the decisions required to produce a sentence and outline the search space for the generation system more perspicuously than do derived trees. This makes derivation trees particularly attractive structures for describing an NLG system; for example, we can represent a TAG derivation tree for utterance (3) with a structure isomorphic to the the dependency tree (4).

4.2 Contextual Inference

SPUD assembles data structures such as (29)–(31) to exploit connections between linguistic meanings and domain-specific representations. For example, the presupposition (30b) connects the meaning of the constituent *fuel-line coupling nut* with shared referents $f4$ and $r11$ in the aircraft domain; SPUD might use the connection to identify these referents to the user.

SPUD's module for contextual inference determines the availability of such connections. The main resource for this module is a domain-specific knowledge base, specified as logical formulas. This knowledge base describes both the private information available to the system and the shared information that characterizes the state of the conversation. Tasks for contextual inference consult this knowledge base: SPUD first translates a potential connection between meaning and context into a theorem-proving query, and then confirms or rejects the connection by using logic programming search to evaluate the query against the contextual knowledge base. When the inferential connection is established, SPUD can record the inference as a constituent of its communicative intent.[12]

We now describe SPUD's knowledge base, SPUD's queries, and the inference procedure that evaluates them in more detail. SPUD's knowledge base is specified in first-order modal logic. First-order modal logic extends first-order classical logic by the addition of MODAL OPERATORS; these operators can be used to relativize the truth of a sentence to a particular time, context or information-state. We will use modal operators to refer to a particular body of knowledge. Thus, if p is a formula and $\Box$ is a modal operator, then $\Box p$ is a formula; $\Box p$ means that p follows from the body of knowledge associated with $\Box$.

For specifications in NLG, we use four such operators: $[\textsc{s}]$ represents the private knowledge of the generation system; $[\textsc{u}]$ represents the private knowledge of the other party to the conversation, the user; $[\textsc{cr}]$ represents the content of the conversational record; and finally $[\textsc{mp}]$ (for MEANING POSTULATES) represents a body of semantic information that follows just from the meanings of words. We regard the four sources of information as subject to the eleven axiom schemes presented in (32):

(32) a $[\textsc{s}]p \supset p.\;\; [\textsc{u}]p \supset p.\;\; [\textsc{cr}]p \supset p.\;\; [\textsc{mp}]p \supset p.$

b $[\textsc{s}]p \supset [\textsc{s}][\textsc{s}]p.\;\; [\textsc{u}]p \supset [\textsc{u}][\textsc{u}]p.\;\; [\textsc{cr}]p \supset [\textsc{cr}][\textsc{cr}]p.\;\; [\textsc{mp}]p \supset [\textsc{mp}][\textsc{mp}]p.$

c $[\textsc{mp}]p \supset [\textsc{cr}]p.\;\; [\textsc{cr}]p \supset [\textsc{s}]p.\;\; [\textsc{cr}]p \supset [\textsc{u}]p$

The system's information, the user's information, the conversational record and the background semantic information are all accurate, according to the idealization of (32a). The effect of (32b) is that hypothetical reasoning with respect to a body of knowledge retains access to all the information in it. Finally, (32c) ensures that semantic knowledge and the contents of the conversational

[12] Note that this strategy is strongly monotonic: SPUD's inference tasks are deductive and the links SPUD adds to communicative intent cannot be threatened by the addition of further information. Previous researchers have pointed out that much inference in interpretation is nonmonotonic (Lascarides and Asher, 1991; Hobbs et al., 1993). We take it as future work to extend SPUD's contextual inference, communicative-intent representations, and search strategy to this more general case.

record are in fact shared. (Stone, 1998) explores the relationship between the idealization of conversation implicit in these inference schemes and proposals for reasoning about dialogue context by Clark and Marshall (1981) and others. For current purposes, note that inferences using the schemes in (32) are not intended to characterize the explicit beliefs of participants in conversation veridically. Instead, the inferences contribute to a data structure, communicative intent, whose principal role is to support conversational processes such as plan recognition, coordination and negotiation.

In this paper, we consider specifications of domain knowledge and queries of domain knowledge that can be restricted to the logical fragment involving *definitions* of category D and *queries* of category Q defined by the following, mutually-recursive rules:

(33)
$$\begin{aligned} D &::= Q \;\mid\; Q \supset D \;\mid\; \forall x D \\ Q &::= [\textsc{cr}]D \;\mid\; [\textsc{s}]D \;\mid\; [\textsc{u}]D \;\mid\; Q \wedge Q \;\mid\; A \end{aligned}$$

A schematizes over any atomic formula; x schematizes over any bound variable. This fragment allows for the kind of clauses and facts that form the core of a logic programming language like Prolog.[13] In addition, these clauses and facts may make free use of modal operators; they may have nested implications and nested quantifiers within queries (e.g., in the antecedents of rules), provided they are immediately embedded under modal operators. There have been a number of proposals for logic programming languages along these lines, such as (Fariñas del Cerro, 1986; Debart et al., 1992; Baldoni et al., 1998). Our implementation follows (Stone, 1999), which also allows for more general specifications including disjunction and existential quantifiers. For a discussion of NLG inference using the more general modal specifications, see (Stone, 2000b).

SPUD's knowledge base in any generation task is a set of D formulas. We notate this knowledge base K. We use the notation $?K \longrightarrow q$ to denote the task of proving a Q-formula q as a query from K; we indicate by writing $K \longrightarrow q$ that this task results in the construction of a proof, and thus that the query succeeds. K provides all the information about the world and the conversation that SPUD can draw on to construct and to evaluate possible communicative intent. Concretely, for SPUD to construct communicative intent, the knowledge base must support any assertions, presuppositions and pragmatics that SPUD decides to appeal to in its utterance. Thus, the knowledge base should explicitly set up as system knowledge any information that SPUD may assert; if some intended update relates by inference to an assertion, the knowledge base must provide, as part of the conversational record, rules sufficient to infer the update from the assertion. Moreover, the knowledge base must provide, as part of the conversational record, formulas which entail the presuppositions and pragmatic conditions that SPUD may impose. Meanwhile, for SPUD to assess whether the hearer will interpret an utterance correctly, the knowledge base must describe the context richly enough to characterize not just the intended communicative intent for a provisional utterance, but also any potential alternatives to it.

For the communicative intent of Figure 6, then, the knowledge base must include the specific private facts that underlie the assertion in the instruction, as in (34):

[13] Note though that pure negative clauses such as $\neg p(X)$ or $\neg p(X) \vee \neg q(X)$ are not part of our fragment.

(34) $[\textsc{s}]\mathit{move}(a1,h0,n11,p(l(on,j2),l(on,e2))).$
$[\textsc{s}]\mathit{next}(a1).$
$[\textsc{s}]\mathit{purpose}(a1,a2).$
$[\textsc{s}]\mathit{uncover}(a2,h0,r11).$

(Recall that, in words, (34) describes the *next* action, a *move* event which takes the nut along a specified path and whose *purpose* is to *uncover* the sealing-ring.) For this communicative intent, no further specification is required for the links between assertions and updates. Updates are expressed in the same terms as meanings here, so the connection will follow without recourse to further axioms linking updates and meanings.

At the same time, the knowledge base must include the specific facts and rules that permit the presuppositions and pragmatics of the instruction to be recognized as part of the conversational record. (35a) spells out the instances that are simply listed in the conversational record; (35b) describes the rules and premises that allow the noun-noun compound and the spatial presuppositions to be interpreted by inference as in (14) and (15).

(35) a $[\textsc{cr}]\mathit{partic}(s0,h0).$ $[\textsc{cr}]\mathit{obl}(s0,h0).$
$[\textsc{cr}]\mathit{surf}(p(l(on,j2),l(on,e2))).$ $[\textsc{cr}]\mathit{zero\text{-}genre}.$
$[\textsc{cr}]\mathit{cn}(n11).$ $[\textsc{cr}]\mathit{def}(n11).$
$[\textsc{cr}]\mathit{el}(e2).$ $[\textsc{cr}]\mathit{def}(e2).$
$[\textsc{cr}]\mathit{sr}(r11).$ $[\textsc{cr}]\mathit{def}(r11).$
$[\textsc{cr}]\mathit{fl}(f4).$ $[\textsc{cr}]\mathit{def}(f4).$
b $[\textsc{cr}]\mathit{for}(r11,f4).$ $[\textsc{cr}]\forall ab(\mathit{for}(a,b)\supset \mathit{nn}(a,b,\mathit{for})).$
$[\textsc{cr}]\mathit{loc}(l(on,j2),n11)$
$[\textsc{cr}]\forall loe(\mathit{loc}(l,o)\supset \mathit{start\text{-}at}(p(l,e),o)).$ $[\textsc{cr}]\forall se(\mathit{end\text{-}on}(p(s,l(on,e)),e))$

(Again, with our conventions, (35a) spells out such facts as that *s*0 and *h*0 are the speaker and hearer *participating* in the current conversation, and that *s*0 is empowered to impose *obligations* on *h*0. Likewise, (35b) indicates that the ring is *for* the fuel-line, and that *for* is the right kind of relationship to interpret a noun-noun compound; that a path that starts where an object is *located starts at* the object; and that any path whose endpoint is *on* an object *ends on* the object.)

Of course, the knowledge base cannot be limited to just the facts that figure in this particular communicative intent. SPUD is designed to be supplied with a number of other facts, both private and shared, about the discourse referents evoked by the instruction. This way SPUD has substantive lexical choices that arise in achieving specified updates to the state of the conversation. SPUD also expects to be supplied with additional facts describing other discourse referents from the context. This way SPUD can consult the specification of the context to arrive at meaningful assessments of ambiguities in interpretation. For instance, the knowledge base must describe any other fuel lines and other sealing rings to settle whether there is any referential ambiguity in the phrase *fuel-line sealing ring*. For exposition, we note only the bare-bones alternatives required for SPUD to generate (3) given the task of describing the upcoming uncovering motion:

(36) a $[\textsc{cr}]sr(a_{r11})$
b $[\textsc{cr}]surf(p(l(on, j2), l(a_{on}, a_{e2})))$

There must be another sealing ring a_{r11} for SPUD to explicitly indicate $r11$ as the *fuel-line* sealing ring; and there must be another path to slide $n11$ along, for SPUD to explicitly describe the intended path as *onto elbow*.[14]

Now we consider the steps involved in linking grammatical structures such as (26) or (29)–(30) to domain-specific representations. As described in Section 4.1, the grammar delivers an assertion A, a presupposition P and pragmatics Q for each derivation tree. Links to domain-specific representations come as SPUD constructs a communicative intent for this derivation tree by reasoning from the context.

In doing this, SPUD must link up P and Q in a specific way with particular referents and propositions from the conversational record. We introduce an assignment σ taking variables to terms to indicate the correspondence between anaphors and intended referents. (We write out assignments as lists of the form $\{\dots V_i \leftarrow t_i \dots\}$ where each variable V_i is assigned term t_i as its value; for any structure E containing variables, and any assignment σ of values to those variables, we use $E\sigma$ to indicate the result of replacing the occurrences of variables in E by the terms assigned by σ.) In addition, SPUD must link up the assertion A with particular open questions in the discourse in virtue of the information it presents about particular individuals. We schematize any such update as a condition U.

These links between A, P and Q and the context constitute the presumptions that SPUD makes with its utterance; SPUD explicitly records them in its representation of communicative intent. Since these links are inferences, constructing them is a matter of proof. In SPUD, these proof tasks are carried out using logic programming inference and a modal specification of context.

- Checking that the intended instance of the assertion A is true corresponds to the proof task:

  $$?K \longrightarrow [\textsc{s}]A\sigma$$

  That is, does some instance of $A\sigma$ follow from the information available to the speaker? As usual in logic programming, if σ leaves open the values of some variables, then the proof actually describes a more specific instance $[\textsc{s}]A\sigma'$ where the substitution σ' possibly supplies values for these additional variables.

[14] SPUD requires such alternatives as motivation any time it uses presupposed material in an utterance; this is because the only effect of presupposition in interpretation in SPUD is to help identify individuals. Other goals have been proposed for including presupposed material in utterance, for example drawing the hearer's attention back to old information, or using it to highlight discourse and task structure; see (Walker, 1993; Jordan, 2000). The treatment of such goals in formal models of interpretation remains an important problem for future work. A practical expedient (not without theoretical merit) is to understand alternatives such as a_{r11} as modeling both the real objects present in the environment and any additional, hypothetical objects that the hearer might be prepared to find in the environment.

SPUD's greedy search also requires that this alternative path not end *on* anything, but instead end perhaps *around* or *over* its endpoint. The explanation for this depends on the results of Section 4.4 and Section 5, but briefly, SPUD will adjoin the modifier *onto* only if *onto* by itself rules out some path referents (and thus by itself helps the hearer to interpret the instruction).

- Checking that the intended instance of the assertion A leads to the update U corresponds to the proof task:

  $$?K \longrightarrow [\textsc{cr}]([\textsc{cr}]A\sigma \supset [\textsc{cr}]U)$$

  That is, considering only the content of the conversational record, can we show that when $A\sigma$ is added to the conversational record, U also becomes part of the conversational record? We assume that A and U do not share variables. Note that $[\textsc{cr}]([\textsc{cr}]p \supset [\textsc{cr}]p)$ is a valid formula of modal logic, for any p. Such a query always succeeds, regardless of the specification K.

- Checking that a presupposition P is met for an intended instance corresponds to the proof task:

  $$?K \longrightarrow [\textsc{cr}]P\sigma$$

  That is, does $P\sigma$ follow from the conversational record? More generally, determining the potential instances under which the presupposition P is met corresponds to the proof task:

  $$?K \longrightarrow [\textsc{cr}]P$$

  Each proof shows how the context supports a specific resolution σ' of underspecified elements in the meaning of the utterance, by deriving an instance $P\sigma'$. Such instances need not be just the one that the system intends. Checking that pragmatic conditions Q are met for an intended instance also corresponds to a query $?K \longrightarrow [\textsc{cr}]Q\sigma$.

Our logic programming inference framework allows queries and knowledge bases to be understood operationally as instructions for search, much as in Prolog; see (Miller et al., 1991). For example, a query $\Box p$ is an instruction to move to a new possible world and consider the query p there; a query $\forall x\, p$ is an instruction to consider a new arbitrary individual in place of x in proving p. A query $p \supset q$ is an instruction to assume p temporarily while considering the query q; a query $p \wedge q$ is an instruction to set up two subproblems for search: a query of p and a query of q. Logical connectives in knowledge-base clauses, meanwhile, are interpreted as describing matches for predicates, first-order terms, and possible worlds in atomic queries, and as setting up subproblems with additional queries of their own. Overall then, each theorem-proving problem initiates a recursive process where the inference engine breaks down complex queries into a collection of search problems for atomic queries, backward-chains against applicable clauses in the knowledge base to search for matches for atomic queries, and takes on any further queries that result from the matches.

As in Prolog, the course and complexity of the proof process can be determined from the form of the queries and the knowledge-base. Thus, when necessary, performance can be improved by astute changes in the representation and formalization of domain relationships. Proof search is no issue with (3), for example; inspection of the clauses in (34), (35) and (36) will confirm that logic programming search explores the full search space for generation queries for this instruction without having to reason recursively through implications.

4.3 Concrete Representations of Communicative Intent

We can now return to the communicative intent of Figure 6 to describe the concrete representations by which SPUD implements it. For reference, we repeat Figure 6 as Figure 10 here.

The grammar delivers a TAG derivation whose structure is isomorphic to the tree-structure of Figure 10. That derivation is associated with a meaning that we represent as the triple of conditions of (37a)–(37c); (37d) spells out the instantiation σ under which this meaning is to be linked to the communicative context:

(37) a Assertion: $move(a1,H,N,P) \wedge next(a1) \wedge purpose(a1,a2) \wedge uncover(a2,H,R)$
b Presupposition: $partic(S,H) \wedge start\text{-}at(P,N) \wedge surf(P) \wedge cn(N) \wedge end\text{-}on(P,E) \wedge el(E) \wedge sr(R) \wedge fl(F) \wedge nn(R,F,X)$
c Pragmatics: $obl(S,H) \wedge def(N) \wedge def(E) \wedge def(R) \wedge def(F) \wedge zero\text{-}genre$
d Instance: $\{H \leftarrow h0, S \leftarrow s0, N \leftarrow n11, P \leftarrow p(l(on,j2),l(on,e2)), R \leftarrow r11, E \leftarrow e2, F \leftarrow f4, X \leftarrow for\}$

(We abbreviate the assertion (37a) by M; and abbreviate the instance (37d) by σ.)

SPUD connects these meanings with domain-specific representations as schematized by the inference notation of Section 2.2 and as formalized by the modal logic queries described in Section 4.2. For example, an inference schematized in (10), repeated as (38), is required to justify the assertion-instance $move(a1,h0,n11,p(l(on,j2),l(on,e2))) = move(a1,H,N,P)\sigma$ and to link it with one of the system's goals for the instruction.

(38)
$$\begin{array}{c} \boxed{move(a1,h0,n11,p(l(on,j2),l(on,e2)))} \\ | \\ move(a1,H,N,P) \end{array}$$

Concretely this corresponds to two proofs which we obtain from the knowledge base K:

(39) a $K \longrightarrow [\textsc{s}]move(a1,H,N,P)\sigma$
b $K \longrightarrow [\textsc{cr}]([\textsc{cr}](M\sigma \supset [\textsc{cr}]move(a1,h0,n11,p(l(on,j2),l(on,e2)))$

The proof (39a) shows that the speaker knows about this motion; the proof (39b) shows that the overall assertion of the sentence will add the description of this motion to the conversational record. Note that (39b) relates the overall assertion of the utterance to the update achieved by a particular word, in this case *slide*. In general, we anticipate the possibility that a single domain-specific fact may be placed on the conversational record by combining the information expressed by multiple words. For example, one word may both provide an inference on its own and complete a complex inference in combination with words already in a sentence. We return to this possibility in Section 6.5.

Each conjunct of the assertion in (37a) contributes its inference to the system's communicative intent. In each case, SPUD represents the inference portrayed informally as a tree in Figure 10 as a pair of successful queries from K, as in (39).

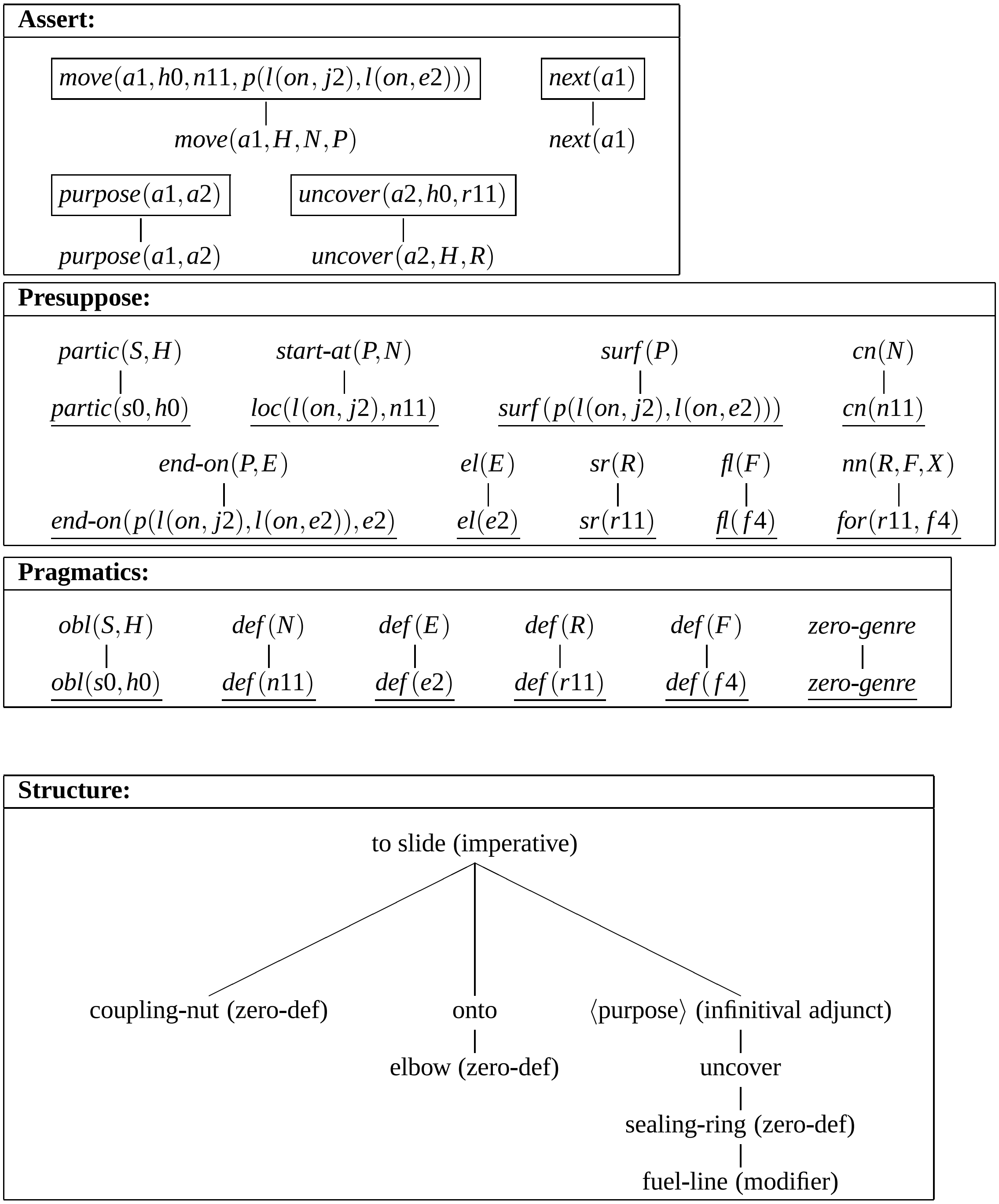


Figure 10: Communicative intent for (3). The grammar specifies meanings as follows: For *slide*, assertions *move* and *next*; for the infinitival adjunct, *purpose*; for *uncover*, *uncover*. For *slide*, presuppositions *partic*, *start-at* and *surf*; for *coupling-nut*, *cn*; for *onto*, *end-on*; for *elbow*, *el*; for *sealing-ring*, *cn*; for *fuel-line*, *fl* and *nn*. For *slide*, pragmatics *obl*; for other nouns, pragmatics *def* and *zero-genre*. The speaker's presumptions map out intended connections to discourse referents as follows: the speaker S, $s0$; the hearer H, $h0$; the nut N, $n11$; the path P, $p(l(on, j2), l(on, e2))$; the elbow E, $e2$; the ring R, $r11$; the fuel-line F, $f4$; the relation X, *for*. The fuel-line joint is $j2$.

Next, consider a presupposition, such as the general form $nn(R,F,X)$ and its concrete instance $nn(r11, f4, for) = nn(R,F,X)\sigma$. Corresponding to the informal inference of (40) we have the proof indicated in (41).

(40) $$\begin{array}{c} nn(R,F,X) \\ | \\ \underline{for(r11, f4)} \end{array}$$

(41) $$K \longrightarrow [\textsc{cr}]nn(R,F,X)\sigma$$

The proof of (41) proceeds by backward chaining using the axiom $[\textsc{cr}]\forall ab(for(a,b) \supset nn(a,b,for))$ and grounds out in the axiom $[\textsc{cr}]for(r11, f4)$; hence the correspondence with (40).

Each conjunct of the presupposition and each conjunct of the pragmatics requires a link to the shared context—an inference as in (40)—and in each case SPUD represents this link by a successful query as in (41).

Appendix A gives a grammar fragment sufficient to generate (3) in SPUD. By reference to the trees of this grammar, SPUD's complete representation of communicative intent for (3) is given in Figure 11.

4.4 *Recognition of Communicative Intent*

Recall from Section 2.2 that structures such as that of Figure 11 represent not only the interpretations that speakers intend for utterances but also interpretations that hearers can recognize for them; in the ideal case, an utterance achieves the updates to the conversation that the speaker intends because the hearer successfully recognizes the speaker's communicative intent. In generating an utterance, SPUD anticipates the hearer's recognition of its intent by consulting a final, inferential model. SPUD uses this model as described in Section 2.3 to keep track of the provisional status of incomplete utterances.

This model incorporates some simplifications that reflect the constrained domains and the constrained communicative settings in which NLG systems are appropriate. Each of these assumptions represents a starting point for further work to derive a more systematic and more general model of interpretation.

- We assume that the hearer can identify the intended lexical elements as contributing to the utterance, and can reconstruct the intended structural relationships among the elements. That is, we assume successful parsing and word-sense disambiguation. On this assumption, the hearer always has the correct syntactic structure for an utterance and a correct representation of its assertion, presupposition and pragmatics. For example, for utterance (3) as in Figure 11, the hearer gets the syntactic structure of the figure and the three conditions of meaning from (37a)–(37c).

- We assume that each update that the utterance is intended to achieve must either be an instance of an open question that has been explicitly raised by preceding discourse, or correspond to an assertion that is explicitly contributed by one of the lexical elements in the

Assert:

$K \longrightarrow [\mathrm{S}]move(a1,H,N,P)\sigma \quad K \longrightarrow [\mathrm{S}]next(a1)\sigma$
$K \longrightarrow [\mathrm{CR}]([\mathrm{CR}]M\sigma \supset [\mathrm{CR}]move(a1,h0,n11,p(l(on,j2),l(on,e2))))$
$K \longrightarrow [\mathrm{CR}]([\mathrm{CR}]M\sigma \supset [\mathrm{CR}]next(a1))$

$K \longrightarrow [\mathrm{S}]purpose(a1,a2)\sigma \qquad K \longrightarrow [\mathrm{S}]uncover(a2,H,R)\sigma$
$K \longrightarrow [\mathrm{CR}]([\mathrm{CR}]M\sigma \supset [\mathrm{CR}]purpose(a1,a2)) \qquad K \longrightarrow [\mathrm{CR}]([\mathrm{CR}]M\sigma \supset [\mathrm{CR}]uncover(a2,h0,r11))$

Presuppose:

$K \longrightarrow [\mathrm{CR}]partic(S,H)\sigma \quad K \longrightarrow [\mathrm{CR}]start\text{-}at(P,N)\sigma \quad K \longrightarrow [\mathrm{CR}]surf(P)\sigma$
$K \longrightarrow [\mathrm{CR}]cn(N)\sigma \quad K \longrightarrow [\mathrm{CR}]end\text{-}on(P,E)\sigma \quad K \longrightarrow [\mathrm{CR}]el(E)\sigma$
$K \longrightarrow [\mathrm{CR}]sr(R)\sigma \quad K \longrightarrow [\mathrm{CR}]fl(F)\sigma \quad K \longrightarrow [\mathrm{CR}]nn(R,F,X)\sigma$

Pragmatics:

$K \longrightarrow [\mathrm{CR}]obl(S,H)\sigma \quad K \longrightarrow [\mathrm{CR}]def(N)\sigma \quad K \longrightarrow [\mathrm{CR}]def(E)\sigma$
$K \longrightarrow [\mathrm{CR}]def(R)\sigma \quad K \longrightarrow [\mathrm{CR}]def(F)\sigma \quad K \longrightarrow [\mathrm{CR}]zero\text{-}genre\sigma$

Structure:

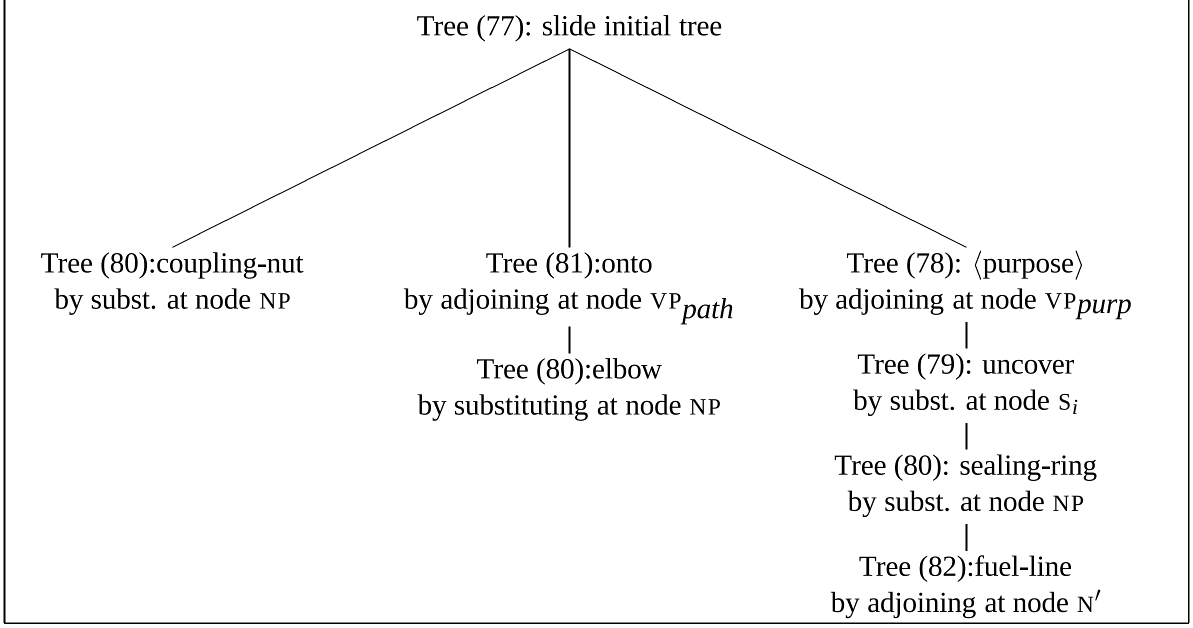


Figure 11: SPUD's representation of the communicative intent in Figure 10. Note two abbreviations for the figure:

$$M := move(a1,H,N,E) \wedge next(a1) \wedge purpose(a1,a2) \wedge uncover(a2,H,R)$$
$$\sigma := \{H \leftarrow h0, S \leftarrow s0, N \leftarrow n11, P \leftarrow p(l(on,j2),l(on,e2)), R \leftarrow r11, E \leftarrow e2, F \leftarrow f4, X \leftarrow for\}$$

Note also that K refers to the knowledge base specified in (34) and (35).

utterance itself. Once the hearer identifies the intended instance of the assertion $M\sigma$, the hearer can arrive at the intended update-inferences by carrying out a set of queries of the form $[\textsc{cr}]([\textsc{cr}]M\sigma \supset [\textsc{cr}]Q)$. Our assumption dictates that the set of possible formulas for Q is finite and is determined by the hearer's information; we make the further assumption that the domain inferences are sufficiently short and constrained that the search for each query is bounded (of course, the generator requires this to design its utterances—whether or not it assesses the hearer's interpretation). The two assumptions justify counting all updates as successfully recognized as long as the hearer can recognize the intended instance σ of the assertion.

- We assume that the hearer attempts to resolve the presupposition according to a shared ranking of SALIENCE. This ranking is formalized using the notion of a CONTEXT SET. Each REFERENT, e, comes with a context set $D(e)$ including it and its distractors; the context set for e determines all the referents that a hearer will consider as possible alternatives in resolving a variable X that the speaker intends to refer to e. This can represent a ranking because we can have $a \in D(b)$ without $b \in D(a)$; in this case a is more salient than b. During the reference resolution process, then, the hearer might have to run through the context set for a before expanding the search to include the context set for b. In practice, we simply assume that the hearer must recognize the context set successfully. That means that the hearer will consider a set of potential resolutions where variables are instantiated to elements of appropriate context sets; we represent this set of potential resolutions as a set of substitutions $D(\sigma)$ defined as follows:

  (42) $\sigma' \in D(\sigma)$ if and only if for each variable X that occurs in the presupposition of the utterance, $\sigma'(X) \in D(\sigma(X))$

  To make this assumption reasonable we have made limited use of gradations in salience.

- We assume that the hearer does not use the pragmatic conditions in order to determine the speaker's intended substitution σ. The hearer simply checks, once the hearer has resolved σ using the presupposition, that there is a unique inference that justifies the corresponding instance of the pragmatics. This assumption allows us to view pragmatic conditions purely as specifications for stylistic choices.

It follows from these assumptions that interpretation is a constraint-satisfaction problem, as in (Mellish, 1985; Haddock, 1989; Dale and Haddock, 1991). In particular, the key task that the hearer is charged with is to recognize the inferences associated with the presupposition of the utterance. That presupposition is an open formula P composed of the conjunction of the individual presupposition formulas P_i contributed by lexical elements. The resolutions compatible with the hearer's information about the utterances are the instances of P that fit the conversational record and the attentional state of the discourse. Formally, we can represent this as Σ' defined in (43).

(43) $\Sigma' := \{\sigma' \in D(\sigma) : K \longrightarrow [\textsc{cr}]P\sigma'\}$

Each of the formulas P_i determines a relation R_i on discourse referents that characterizes instances that the speaker may have intended; SPUD computes this relation by querying the knowledge base as in (44), and represents it compactly in terms of the free variables that occur in P_i.

(44) $R_i = \{\sigma' \in D(\sigma) : K \longrightarrow [\textsc{cr}]P_i\sigma'\}$

SPUD then uses an arc-consistency constraint-satisfaction heuristic on these relations to solve for Σ' (Mackworth, 1987). (This is a conservative but efficient strategy for eliminating assignments that are inconsistent with the constraints.) SPUD counts the inferences for the presupposition as successfully recognized when the arc-consistency computation leaves only a single possibility, namely the intended resolution σ.

5 Microplanning as a Search Task

The preceding sections have been leading up to a characterization of microplanning as a formal search task (Nilsson, 1971). We argued in Section 2 that a generator must represent the interpretation of an utterance as a data structure which records inferences that connect the structure of an utterance with its meaning, ground the meaning of an utterance in the current context, and draw on the meaning of the utterance to register specified information in the conversational record. In Section 3, we described the grammatical knowledge which defines the structure and meaning of utterances; in Section 4.2, we described the inferential mechanisms which encode the relationships between utterance meaning and an evolving conversational record. With these results, we obtain the specific data structure that SPUD uses to represent communicative intent, in the kinds of records schematized in Figure 11; and the concrete operations that SPUD uses to derive representations of communicative intent, by the steps of grammatical composition and contextual inference described in Sections 4.1, 4.2, and 4.4. Thus, we obtain a characterization of the microplanning problem as a SEARCH, whose RESULT is an appropriate communicative-intent data structure, and which PROCEEDS by steps of grammatical derivation and contextual inference.

5.1 A Formal Search Problem

In SPUD, the specification of a microplanning search problem consists of the following components:

(45) a a background specification of a GRAMMAR G describing the system's model of language (as outlined in Section 3) and a KNOWLEDGE BASE K describing the system's model of its domain, its user and the conversational record (as outlined in Section 4.2);

b a set of formulas, UPDATES, describing the specified facts that the utterance must add to the conversational record;

c a specification of the ROOT NODE of the syntactic tree corresponding to the utterance. This specification involves a syntactic category; variables specifying the indices of the root node; a substitution σ_0 describing the intended values that those variables must have; and a top feature structure, indicating syntactic constraints imposed on the utterance from the external context; cf. (20).

For instance, we might specify the task of describing the sliding action $a1$ by an instruction such as (3) as follows.

(46) a The GRAMMAR G outlined in Appendices A and B; the knowledge base outlined in (34), (35), and (36).
b Four UPDATES: $move(a1, h0, n11, p(l(on, j2), l(on, e2)))$; $next(a1)$; $purpose(a1, a2)$; $uncover(a2, h0, r11)$.
c A root node $\mathrm{S} \downarrow (E)$ with intended instance $\{E \leftarrow a1\}$.

The grammar and knowledge base of (45a) determine the search space for the NLG task. States in the search space are data structures for communicative intent, as argued for in Section 2 and as illustrated in Section 4.3. In particular, each state involves:

(47) a a syntactic structure T derived according to G and paired with a meaning $\langle A, P, Q \rangle$ giving the assertion, presupposition and pragmatics of T (respectively);
b a substitution σ determining the discourse referents intended for the variables in A, P, and Q;
c inferences $K \longrightarrow [\mathrm{S}]A\sigma$, $K \longrightarrow [\mathrm{CR}]P\sigma$, and $K \longrightarrow [\mathrm{CR}]Q\sigma$—such inferences show that the context supports use of this utterance to describe σ;
d inferences of the form $K \longrightarrow [\mathrm{CR}]([\mathrm{CR}]A\sigma \longrightarrow [\mathrm{CR}]F)$ where F is an update—such inferences witness that the utterance supplies needed information;
e a constraint network approximating $\Sigma' := \{\sigma' \in D(\sigma) : K \longrightarrow [\mathrm{CR}]P\sigma'\}$—this network represents the hearer's interpretation of reference resolution.

The INITIAL STATE for search is given in (48).

(48) a a syntactic structure consisting of a single substitution site matching the root node of the problem specification (45c) and paired with an empty meaning;
b the specified intended resolution σ_0 of variables in this syntactic structure;
c no inferences—a record that suffices to justify the empty meaning of the initial state but which shows that this state supplies no needed information;
d an unconstrained network realizing $\Sigma' := \{\sigma' \in D(\sigma_0)\}$.

A GOAL STATE for search is one where the three conditions of (49) are met.

(49) a The syntactic structure of the utterance must be complete: top and bottom features of all syntactic nodes must agree, and all substitution sites must be filled.
b For each update formula F, the communicative intent must include an update inference that establishes a substitution instance of F. More formally, on the assumption that M is the assertion of the utterance and that σ is the intended instance of M, the requirement is that the communicative intent include an inferential record of the form $K \longrightarrow [\mathrm{CR}]([\mathrm{CR}]M\sigma \supset [\mathrm{CR}]F\sigma')$. (We use σ and σ' to enforce that M and F cannot share variables.)

c The arc-consistency approximation to the key presupposition-recognition problem the hearer faces for the communicative intent, as defined in Section 4.4, identifies uniquely the intended substitution of knowledge-base discourse referents for discourse-anaphor variables in the utterance.

The requirements of (49) boil down simply to this: the generator's communicative intent must provide a complete sentence (49a) that says what is needed (49b) in a way the hearer will understand (49c). Observe that the communicative intent of Figure 11 fulfills the conditions in (49) for the microplanning problem of (46).

To derive a new state from an existing state as in (47) involves the steps outlined in (50).

(50) a Construct a lexico-grammatical element L, according to the steps of (25).

b Apply a syntactic operation combining L with the existing syntactic structure T (cf. Section 4.1); the result is a new structure T' and a new meaning $\langle A \wedge A', P \wedge P', Q \wedge Q' \rangle$ that takes into account the contribution $\langle A', P', Q' \rangle$ of L.

c Ensure that the use of this element is supported in context, by proving $K \longrightarrow [\textsc{s}]A'\sigma$, $K \longrightarrow [\textsc{cr}]P'\sigma$ and $K \longrightarrow [\textsc{cr}]Q'\sigma$; the result is a refined substitution σ' describing the intended instantiation not just of T but also of L.

d Record the communicative effects of the new structure in any inferences $K \longrightarrow [\textsc{cr}]([\textsc{cr}](A \wedge A')\sigma' \longrightarrow [\textsc{cr}]F)$ for outstanding updates F.

e Refine the constraint network to take into account the new constraint P'.

Any state so derived from a given state is called a NEIGHBOR of that state.

Because such searches begin at an initial substitution site and derive neighbors by incorporating single elements into the ongoing structure, this characterization of microplanning in terms of search builds in SPUD's head-first derivation strategy. On the other hand, it is compatible with any search algorithm, including brute-force exhaustive search, a traditional heuristic search method such as A* (Hart et al., 1968), or a stochastic optimization search (Mellish et al., 1998).

5.2 A Greedy Search Algorithm

We chose to implement a greedy search algorithm in SPUD. Greedy search applies iteratively to update a single state in the search space, the CURRENT STATE. In each iteration, greedy search first obtains all the neighbors of the current state. Greedy search then ranks the neighbors by a heuristic evaluation intended to assess progress towards reaching a goal state. The neighbor with the best heuristic evaluation is selected. If this state is a goal state, search terminates; otherwise this state becomes the current state for the following iteration.

In developing SPUD, we have identified a number of factors that give evidence of progress towards obtaining a complete, concise, natural utterance that conveys needed information unambiguously.

1. How many update formulas the utterance has conveyed. Other things being equal, if fewer updates remain unrealized, then fewer steps of lexical derivation will be required to convey this further required information.

2. How many alternative values the hearer could consider for each free variable which the system must resolve. Other things being equal, the fewer values remain for each variable, the fewer steps of lexical derivation will be required to supply content that eliminates the ambiguity for the hearer. The concrete measure for this factor in SPUD is a sorted list containing the number of possible values for each ambiguous variable in the constraint network; lists are compared by the lexicographic ordering.

3. How SALIENT the intended values for each free variable are. Other things being equal, an utterance referring to salient referents may prove more coherent and easier for the hearer to resolve (irrespective of its length). Again, the concrete measure for this factor in SPUD is a sorted list of counts, compared lexicographically; the counts here are the sizes of context sets for each intended referent.

4. How many FLAWS remain in the syntactic structure of the utterance. Flaws are open substitution sites and internal nodes whose top and bottom features do not unify. Each flaw can only be fixed by a separate step of grammatical derivation. Other things being equal, the fewer flaws remain, the fewer further syntactic operations will be required to obtain a complete grammatical utterance. We also prefer states in which an existing flaw has been corrected but new flaws have been introduced, over a structure with the same overall number of flaws but where the last step of derivation has not resolved any existing flaws.

5. How SPECIFIC the meanings for elements in the utterance are. In general, an element with a more specific assertion offers a more precise description for the hearer; an element with a more specific presupposition offers more precise constraints for identifying objects; an element with a more specific pragmatic conditions fits the context more precisely. We assess specificity off-line using the semantic information associated with the operator $[\text{MP}]$. If the query $?K \longrightarrow [\text{MP}](M \supset N)$ succeeds, we count formula M as at least as specific as N. We prefer words with more specific pragmatics; then (other things being equal) words with more specific presuppositions; then (other things being equal) words with more specific content; then (other things being equal) words in constructions with more specific pragmatics.

In our implementation of SPUD, we use all these criteria, prioritized as listed, to rank alternative options. That is, SPUD ranks option S ahead of option S' if one of these factors favors S over S' and all factors of higher priority are indifferent between S and S'.[15]

In designing SPUD with greedy search, we drew on the influential example of (Dale and Haddock, 1991), which used greedy search in referring expression generation; and on our own experience using greedy algorithms to design preliminary plans to achieve multiple goals (Webber et al., 1998). As described in Sections 6 and 7, we believe that our experience with SPUD supports our decision to use a sharply constrained search strategy; limited and predictable search makes it easier

[15] It happens that this is also the treatment of ranked constraints in optimality theory (Prince and Smolensky, 1997)!

to understand the behavior of the system and to design appropriate specifications for it. However, we do NOT claim that our experience offers a justification for the specific ranking we used beyond two very general preferences—a primary preference for adding lexical elements that make some progress on the generation task over those that make none (on syntactic, informational or referential grounds); and a secondary preference based on pragmatic specificity. In general, the relationships between search algorithms, specification development and output quality for microplanning based on communicative intent, remains an important matter for future research.

6 Solving NLG tasks with SPUD

In this section, we support our claims that decision-making based on communicative intent provides a uniform framework by which SPUD can simultaneously address all the subtasks of microplanning. We further argue that such a framework is essential for generating utterances that are EFFICIENT, in that they exploit the contribution of a single lexico-grammatical element to multiple goals and indeed to multiple microplanning subtasks. Throughout the section, we illustrate how SPUD's grammatical resources, inference processes, and search strategy combine to solve these problems together for instruction (3). Additional examples of using SPUD in generation can be found in (Bourne, 1999; Cassell et al., 2000); we also investigate these issues from the perspective of designing specifications for SPUD in Section 7.

6.1 Referring Expressions

The problem of generating a referring expression for a simple (i.e., non-event) discourse referent a is to devise a description that can be realized as a noun phrase by grammar G and that uniquely identifies a in context K. Such a problem can be posed to SPUD by the problem specification of (51).

(51) a the grammar G and context K
b no updates to achieve
c an initial node NP $\downarrow(X)$ and an initial substitution $\sigma_0 = \{X \leftarrow a\}$

By the criteria of (49), a solution to this task is a record of communicative intent which specifies a complete grammatical noun phrase and which determines a constraint-satisfaction network that identifies a unique intended substitution, including the assignment $X \leftarrow a$.

The following example demonstrates the close affinity between SPUD's strategy and the algorithm of (Dale and Haddock, 1991). In Figure 12, we portray a context K which supplies a number of salient individuals, including a rabbit $r1$ located in a hat $h1$; K records each individual with visual properties such as kind, size, and location. We consider the problem of generating a referring expression to identify $r1$.

With a suitable grammar, K allows us to construct the communicative intent schematized in Figure 13 for (52).

(52) the rabbit in the hat

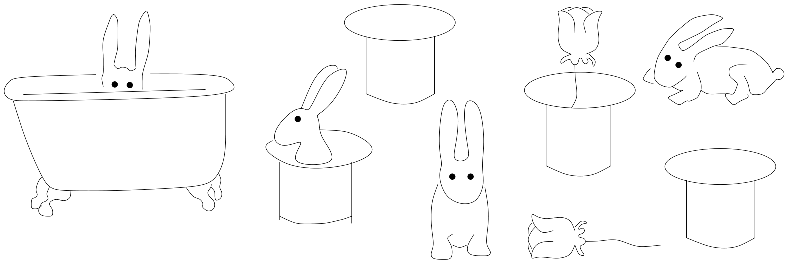


Figure 12: A representation of the context for a referring expression generation task

Assert:

(none)

Presuppose:

$$\begin{array}{ccc} \mathit{rabbit}(R) & \mathit{in}(R,H) & \mathit{hat}(H) \\ | & | & | \\ \underline{\mathit{rabbit}(r1)} & \underline{\mathit{in}(r1,h1)} & \underline{\mathit{hat}(h1)} \end{array}$$

Pragmatics:

$$\begin{array}{cc} \mathit{def}(R) & \mathit{def}(H) \\ | & | \\ \underline{\mathit{def}(r1)} & \underline{\mathit{def}(h1)} \end{array}$$

Structure:

rabbit (definite)
|
in (noun postmodifier)
|
hat (definite)

Figure 13: Communicative intent for *the rabbit in the hat*.

SPUD's model of interpretation, like Dale and Haddock's, predicts that the hearer successfully recognizes this communicative intent, because the context supplies a unique pair of values for variables R and H such that R is a rabbit, H is a hat, and R is in H. Thus, (52) represents a potential solution to the reference task both for SPUD and for Dale and Haddock.

In fact, in deriving *the rabbit in the hat*, the two algorithms would use parallel considerations to take comparable steps. SPUD's derivation, like Dale and Haddock's, consists of three steps in which specific content enriches a description: first *rabbit*, then *in* and finally *hat*. For both algorithms, the primary consideration for choosing each of these steps of derivation is that each narrows the domain of values for variables more than the available alternative steps.

We note three important contrasts between SPUD's approach and Dale and Haddock's, however. First, SPUD typically formulates referring expressions not in isolated subtasks as suggested in (51) but rather as part of a single, overall process of sentence formulation. SPUD's broader view is in fact necessary to generate instructions such as (3)—a point we return to in Section 6.5.

Second, SPUD's options at each step are determined by grammatical syntax, whereas Dale and Haddock's must be determined by a separate specification of possible conceptual combinations. For example, SPUD directly encodes the syntactic requirement that a description should have a head noun using the NP substitution site; for Dale and Haddock this requires an *ad hoc* restriction on what concepts may be included at certain stages of description.

Third, Dale and Haddock adopt a fixed, depth-first strategy for adding content to a description. Particularly since (Dale and Reiter, 1995), such fixed (and even domain-specific) strategies have become common for referring expressions made up of properties of a single individual. It is difficult to generalize a fixed strategy to relational descriptions, however. Indeed, Horacek (Horacek, 1995) challenges fixed strategies with examples that show the need for modification at multiple points in an NP, such as (53).

(53) the table with the apple and with the banana

In SPUD, the order of adding content is flexible. An LTAG derivation allows modifiers to adjoin at any node at any step of the derivation. This places descriptions such as (53) within SPUD's search space. (SPUD's flexibility also contrasts with a top-down derivation in a context-free grammar, where modifiers must be chosen before heads and there is a resulting tension between providing what the syntax requires and going beyond what the syntax requires. See (Elhadad and Robin, 1992) for discussion of the resulting difficulties in search.)

6.2 Syntactic Choice

The problem of syntactic choice is to select an appropriate grammatical construction in which to realize a given lexical item. For example, for English noun phrases, the problem is to select an appropriate determiner from among options including the indefinite marker *a*, the definite marker *the* and the demonstrative markers *this* and *that*. With main verbs in English sentences, the problem involves such decisions as the appropriate use of active or passive voice, and the appropriate fronting or preposing of marked argument constituents.

For SPUD, alternatives for such syntactic choices are represented as alternative states which SPUD's greedy search must consider at some stage of generation. All alternative syntactic entries whose pragmatic conditions are supported in the context will be available. Since these syntactic alternatives share a common lexical specification, their interpretations differ only by the contribution of the distinct pragmatic conditions. Recall that the pragmatics contributes neither to the updates that an utterance achieves nor to the resolution of referential ambiguity, in SPUD's model of interpretation. The pragmatic condition controls purely stylistic factors. Accordingly, SPUD's ranking of these alternatives is based only on the specificity of the pragmatic conditions. SPUD's strategy for syntactic choice is to select a licensed form whose pragmatic condition is maximally specific.

As an illustration of this strategy, consider the syntactic frame for the verb *slide* in instruction (3). The instruction exhibits the imperative frame *slide* NP. Recall that we associate this frame semantically with the condition that a sliding is the next action that the hearer should perform; we associate it with the pragmatic condition that the speaker is empowered to impose obligations for action on the hearer. This pragmatic condition distinguishes *slide* NP from other possible descriptions of this action. One such possibility is *you should slide* NP; we would represent this as a neutral alternative with an always true pragmatic condition. Thus, when SPUD considers both alternatives, it favors *slide* NP because of its specific pragmatics. (In (Stone and Doran, 1997), we consider choice of a topicalized frame, represented with the pragmatic conditions proposed for topicalization in (Ward, 1989), over an unmarked frame; we describe how the generation of *the syntax book, we have* follows from this specification under SPUD's preference for specificity.)

Syntactic frames for the noun phrases provide a similar illustration. Noun phrases in our aircraft maintenance manuals are realized in one of two frames: a zero definite realization for a unique referent, as in *coupling nut*, and a realization with an explicit numeral, used in the other cases (plural referents, such as *two coupling nuts*, and indefinite singular referents, such as *one coupling nut*). We associate the zero definite realization with a pragmatic condition, as in (22), requiring a definite referent and an appropriate linguistic genre; the realization with the explicit numeral is a default whose pragmatic conditions are always satisfied for this genre. The zero definite is chosen whenever applicable, by specificity. More generally, whichever of the two entries, the zero-definite noun phrase or the numerical noun phrase, best applies to a referent in the maintenance domain, SPUD will prefer that entry to the corresponding ordinary definite (*the*) or indefinite (*a*) noun-phrase entry. The genre-restricted entry carries a pragmatic condition on genre which the ordinary entry lacks; thus the genre-restricted entry is selected as more specific.

We credit to systemic linguistics the idea that choices in syntactic realization should be made incrementally, by consulting a model of the discourse and a specification of the functional consequences of grammatical choices. (Mathiessen, 1983) is a classic implementation for generation, while (Yang et al., 1991) explores the close connection between systemic linguistics and TAG. However, SPUD departs from the systemic approach in that pragmatic conditions are associated with individual constructions rather than linguistic systems; this departure also necessitates SPUD's criterion of specificity. Inspiration for both of these moves can be found in research on the discourse function of syntactic constructions, such as (Prince, 1986; Hirschberg, 1985; Ward, 1989; Gundel et al., 1993; Birner, 1992). More generally, as hinted in our contrast of zero-definite noun phrases versus *the* noun phrases, we hypothesize that pragmatically-conditioned constructions, selected in context by specificity, make for grammars that can incorporate general defaults in realization while also modeling the tendency of specific genres or sublanguages to adopt characteristic styles of communication (Kittredge et al., 1991). This hypothesis merits further detailed investigation.

6.3 Lexical Choice

Problems of lexical choice arise whenever a microplanner must apportion abstract content onto specific lexical items that carry this content (in context). Our model of this problem follows (Elhadad et al., 1997). According to this approach, in lexical choice, the microplanner must select words to contribute several independently-specified conditions to the conversational record. Some of these conditions characteristically "float", in that they tend to be realized across a range of syntactic constituents at different linguistic levels, and tend to be realized by lexical items that put other needed information on the record. We agree with the argument of Elhadad et al. that a solution to such problems depends on declarative conceptual and linguistic descriptions of lexical items and accurate assessments of the contribution of lexical items to interpretation. (We agree further that this lexical choice cannot be solved as an isolated microplanning subproblem, and must be solved concurrently with such other tasks as syntactic choice.)

Elhadad et al.'s example is (54); the sentence adopts an informal and concise style to describe an AI class for an academic help domain.

(54) AI requires six assignments.

The choice of verb *requires* here responds to two generation goals. First, it conveys simply that the AI class involves a given set of assignments. The generator has other lexical alternatives, such as *y has x* or *there are x in y*, that do the same. In addition, *requires* conveys that the assignments represent a significant demand that the class places on its students. This second feature distinguishes *requires* from alternative lexical items and accounts for the generator's selection of it.

Both for Elhadad et al. and for SPUD, the selection of *requires* for (54) depends on its lexical representation, which must spell out the two contributions the verb can make. In SPUD, these contributions can be represented as assertions made when using *require* to describe a state *S* associating a class *C* with assignments *A*, as in (55).

(55) Assertion: $\mathit{involve}(S,C,A) \wedge \mathit{demand}(A)$

Meanwhile, a microplanning task might begin with goals to convey two specific instances about the AI class, *c1*; its assignments, *a1*; and an eventuality, *s1*, as in (56).

(56) a $\mathit{involve}(s1,c1,a1)$
b $\mathit{demand}(a1)$

In a context which supplies the information in (56), SPUD can add an instance of *require* as in (55) to augment a sentence about $s1$; the instantiation σ has $\{S \leftarrow s1, C \leftarrow c1, A \leftarrow a1\}$. Using M to abbreviate the *require* assertion from (55), SPUD's assessment of interpretation now records the completed inferences in (57).

(57) a $[\textsc{cr}](M\sigma \supset \mathit{involve}(s1,c1,a1))$
b $[\textsc{cr}](M\sigma \supset \mathit{demand}(a1))$

Thus, SPUD recognizes the opportunistic dual contribution of *require*, and will therefore prefer *require* to other lexical alternatives that do not make a similar contribution.

Despite the high-level similarity, SPUD's mechanisms for grammatical and contextual inference are quite different to those of (Elhadad et al., 1997). Elhadad et al. achieve flexibility of search by logic-programming constructs that allow programmers to state meaningful dependencies and alternatives in the generator's decisions in constructing a context-free phrase structure by top-down traversal. For SPUD, dependencies and alternatives are represented using the extended domain of locality of LTAG; SPUD's strategy for updating decisions about the linguistic realization of floating constraints thus depends on its LTAG derivation and incremental interpretation.

Moreover, because SPUD's model of interpretation is broader, we account for more diverse interactions in microplanning; we explore this in more detail in Section 6.5 and explore its consequences for the design of SPUD specifications for lexical choice in Section 7.3.

6.4 Aggregation

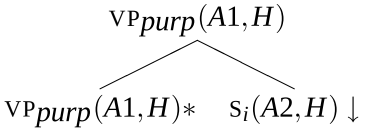
The microplanning process of aggregation constructs complex sentences in which assemblies of lexical items achieve multiple simultaneous updates to the conversational record. Instruction (3) represents a case of aggregation because the combination of *slide*, a infinitival purpose clause, and *uncover* conveys four updates to the conversational record with a single sentence: the *next* event is a *sliding* whose *purpose* is an *uncovering*.

Aggregation is so named because many microplanners produce complex sentences through syntactic operations that combine together, or aggregate, specifications of simple linguistic structures (Reiter and Dale, 2000). For example, such a system might derive instruction (3) by stitching together specifications for these simple sentences: *slide the coupling nut to the elbow*; *the sliding has a purpose*; *the purpose is uncovering the sealing ring*. Each of these sentence specifications directly corresponds to a single given update. The specifications can be combined by describing transformations that create embedded syntactic structures under appropriate syntactic, semantic and pragmatic conditions.

SPUD's architecture distinguishes aggregation at a rhetorical level, in which the content planner organizes related pieces of information that could potentially be expressed together, from aggregation at the surface level, in which a specific sentence is formulated to realize a constellation of communicative goals. SPUD handles surface-level aggregation directly. In SPUD, this process is not a distinct stage of microplanning that draws on idiosyncratic linguistic resources; instead, this aggregation arises as a natural consequence of the incremental elaboration of communicative intent using a grammar. Initial phases of lexicalization leave some updates unexpressed; for example, after SPUD's selection in (3) of the imperative transitive verb *slide*, SPUD still has the goals of updating the conversational record to the event's *purpose*, of *uncovering*. These lexical and syntactic decisions also trigger new grammatical entries that adjoin into SPUD's provisional linguistic structure and augment the provisional communicative intent. Such entries provide the grammatical resources by which SPUD's subsequent lexicalization decisions can directly contribute to complex sentences that achieve multiple communicative goals.

For example, in (3), *slide* introduces a VP$_{purp}$ node indexed by the sliding event $a1$ and its agent $h0$. This is a site where the lexico-syntactic entry in (58) could adjoin.

(58) a TREE:



b ASSERTION: $purpose(A1,A2)$

c PRESUPPOSITION and PRAGMATICS: —

(58) is a declarative description of the form and meaning of an English infinitival purpose construction, expressed in the general terms required for reasoning about the interpretation of assemblies of linguistic constructions in context. Specifically, (58a) assumes that the purpose clause modifies a specific VP node and subcategorizes for an infinitive S.[16]

At the same time, (58) also has an operational interpretation for generation, as a pattern of possible aggregation: (58) describes when and how a description of an event can be extended to include a characterization of the purpose of the event. This operational interpretation provides a complementary motivation for each of the constituents of (58). An aggregation pattern must indicate how new material can be incorporated into an existing sentence; this is determined by the syntactic tree in (58a). And it must indicate what updates are realized by the addition; this is the role of the assertion in (58b).

More generally, an aggregation pattern must indicate how the syntactic realization of aggregated material depends on its subordination to or coordination with other linguistic structure. Languages generally offer lightweight constructs, such as participles and prepositional phrases, which augment a sentence with less than another full clause. Syntactic trees such as that in (58a) provide a natural specification of these constructs. Finally, the pattern must characterize the idiosyncratic interpretive constraints that favor one aggregated realization over another. Not all realizations are equally good; alternatives may require specific informational or discourse relationships, such as the inferrability between events that some adjuncts demand (Cheng and Mellish, 2000). As an aggregation pattern, (58) represents such characterizations of requirements on context by appropriate pragmatic conditions or presuppositions.

Selecting entry (58) is SPUD's analogue of an aggregation process; by using it, SPUD derives a provisional sentence including *slide* and requiring a further infinitive clause. SPUD substitutes *to uncover* for the infinitive sentence in the purpose clause in a subsequent step of lexicalization. This grammatical derivation results in a single complex sentence that achieves four updates to the conversational record.

6.5 Interactions in Microplanning

SPUD is capable of achieving specified behavior on isolated microplanning tasks, but a key strength of SPUD is its ability to model INTERACTIONS among the requirements of microplanning.

[16] Lexicalization purists could add a covert subordinating conjunction to head the tree in (58a), but SPUD does not require it.

Different requirements can usually be satisfied in isolation by assembling appropriate syntactic constituents—for example, by identifying an individual using a noun phrase that refers to it or by communicating a desired property of an action using a verb phrase that asserts it. However, many sentences exhibit an alternative, more efficient strategy which we have called TEXTUAL ECONOMY: the sentences satisfy some microplanning objectives implicitly, by exploiting the hearer's (or reader's) recognition of inferential links to material elsewhere in the sentence that is there for independent reasons (Stone and Webber, 1998). Such material is therefore *overloaded* in the sense of (Pollack, 1991).[17]

The main clause of (3), repeated as (59), is in fact illustrative of textual economy that exploits interactions among problems of referring expression generation and lexical choice within a single clause.

(59) Slide coupling nut onto elbow.

Consider the broader context in which (59) will be used to instruct the action the depicted in Figure 3. Given the frequent use of coupling nuts and sealing rings to join vents together in aircraft, we cannot expect this context to supply a single, unique coupling nut. Indeed, diagrams associated with instructions in our aircraft manuals sometimes explicitly labeled multiple similar parts. Allocating tasks of verb choice and referring expression generation to independent constituents in such circumstances would therefore lead to unnecessarily verbose utterances like (60).

(60) Slide coupling nut that is over fuel-line sealing ring onto elbow.

Instead, it is common to find instructions such as (59), in which these parts are identified by abbreviated descriptions; and such instructions seem to pose no difficulty in interpretation. Intuitively, the hearer can identify the intended nut from (59) because of the choice of verb: one of the semantic features of the verb *slide* is the constraint that its object (here, the coupling nut) moves in contact along a surface to reach its destination (here, the elbow). Identifying the elbow directs the hearer to the coupling nut on the fuel line, since that coupling nut alone lies along a common surface with the elbow.

The formal representation of communicative intent in Figure 11 implements this explanation. It associates the verb *slide* with proofs $K \longrightarrow [\textsc{cr}]\mathit{surf}(P)$, $K \longrightarrow [\textsc{cr}]\mathit{start\text{-}at}(P,N)$ and $K \longrightarrow [\textsc{cr}]\mathit{end\text{-}on}(P,E)$ which together require the context to establish that the nut lie on a common surface with the elbow. Accordingly, the constraint-network model of communicative-intent recognition described in Section 4.4 uses this requirement in determining candidate values for N and E. The network will heuristically identify coupling nuts that lie on a common surface with an elbow. In this case, the constraints suffice jointly to determine the arguments in the action. Thus, when SPUD constructs the communicative intent in Figure 11, it models and exploits an interaction between the microplanning tasks of referring expression generation and lexical choice.

[17] Pollack used the term *overloading* to refer to cases where a single intention to act is used to wholly or partially satisfy several of an agent's goals simultaneously.

In (Stone and Webber, 1998), we make a similar point by analyzing the instruction (61) in the context depicted in Figure 12.

(61) Remove the rabbit from the hat.

From (61), the hearer should be able to identify the intended rabbit and the intended hat—even though the context supplies several rabbits, several hats, and even a rabbit in a bathtub and a flower in a hat. The verb *remove* presupposes that its object (here, the rabbit) starts out in the source (here, the hat), and this distinguishes the intended rabbit and hat in Figure 12 from the other ones.

Where instructions such as (59) exploit interactions between referring expression generation and lexical choice, instructions exhibiting PRAGMATIC OVERLOADING exploit interactions between aggregation and lexical choice (Di Eugenio and Webber, 1996). DiEugenio and Webber characterize the interpretation of instructions with multiple clauses that describe complex actions, such as (62).

(62) a Hold the cup under the spigot—
b —to fill it with coffee.

Here, the two clauses (62a) and (62b) are related by enablement, a kind of purpose relation. Because of this relation, the description in (62b) forms the basis of a constrained inference that provides additional information about the action described in (62a). That is, while (62a) itself does not specify the orientation of the cup under the spigot, its purpose (62b) can lead the hearer to an appropriate choice. To fill a cup with coffee, the cup must be held vertically, with its concavity pointing upwards. As noted in (Di Eugenio and Webber, 1996), this inference depends on the information available about the action in (62a) and its purpose in (62b). The purpose specified in (63) does not constrain cup orientation in the same way:

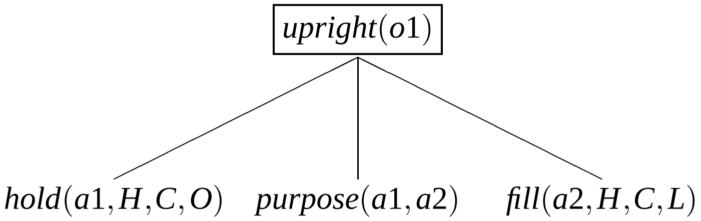
(63) Hold the cup under the faucet to wash it.

In a representation of communicative intent, the pragmatic overloading of (62) manifests itself in an update to the conversational record that is achieved by inference. Suppose that we represent the cup as $c1$, the action of holding it under the spigot as $a1$, and the needed spatial location and orientation as $o1$; at the same time, we may represent the filling as action $a2$, and the coffee as liquid $l1$. We contribute by inference that the orientation is upright—$\mathit{upright}(o1)$—because we assert that $a1$ is an action where the hearer $h1$ holds $c1$ in $o1$—$\mathit{hold}(a1,h1,c1,o1)$—whose purpose is the action $a2$ of filling $c1$ with $l1$—$\mathit{purpose}(a1,a2) \wedge \mathit{fill}(a2,h1,c1,l1)$; and because we count on the hearer to recognize that an event in which something is held to be filled must involve an upright orientation—in symbols:

(64) $[\textsc{cr}]\forall ee'xcol[\mathit{hold}(e,x,c,o) \wedge \mathit{purpose}(e,e') \wedge \mathit{fill}(e',x,c,l) \supset \mathit{upright}(o)]$

The notation of Section 2.2 records this inference as in (65), a constituent of the communicative intent for (62).

(65)



Because SPUD assesses the interpretations of utterances by looking for inferential possibilities such as (65), it can recognize the textual economy in utterances such as (62). Moreover, because SPUD interleaves reasoning for aggregation and lexical choice (and referring expression generation), SPUD can orchestrate the lexical content of clauses in order to take advantage of inferential links like that of (65).

Thus, suppose that SPUD starts with the goal of describing the holding action in the main clause, describing the filling action, and indicating the purpose relation between them. For the holding action, SPUD's goals include making sure that the sentence communicates where the cup will be held and how it will be held (i.e., *upright*). SPUD first selects an appropriate lexico-syntactic tree for imperative *hold*; SPUD can choose to adjoin in the purpose clause next, in an aggregation move, and then to make the appropriate lexico-syntactic choice of *fill*. After this substitution, the semantic contributions of the sentence describe an action of *holding an object* which *can bring about* an action of *filling that object*. As shown in (Di Eugenio and Webber, 1996), and as formalized in (64), these are the premises of an inference that the object is held upright during the filling. When SPUD assesses the interpretation of this utterance, using logical queries about the updates it could achieve, it finds that the utterance has in fact conveyed how the cup is to be held. SPUD has no reason to describe the orientation of the cup with additional content.

7 Building specifications

We have seen how SPUD plans sentences not by a modular pipeline of subtasks, but by general reasoning that draws on detailed linguistic models and a rich characterization of interpretation. While this generality makes for an elegant uniformity in microplanning, it also poses substantial obstacles to the development of SPUD specifications. Because of SPUD's general reasoning, changes to any lexical and syntactic entry have far-reaching and indirect consequences on generation results.

In response to this challenge, we have developed a methodology for constructing lexicalized grammatical resources for generation systems such as SPUD. Our methodology involves guidelines for the construction of syntactic structures, for semantic representations and for the interface between them. In this section, we describe this methodology in detail, and show, by reference to a case study in a specific instruction-generation domain, how this methodology helps ensure that SPUD deploys its lexical and syntactic options as observed in a corpus of desired output. In the future, we hope that this methodology can serve as a starting point for automatic techniques of specification development and validation from possibly paired corpora of syntactic and semantic representations——a problem that has begun to draw attention from the perspective of interpretation as well (Hockenmaier et al., 2003; Kingsbury and Palmer, 2002; Kingsbury and Kipper, 2003; Gildea and Hockenmaier, 2003).

The basic principle behind all of our guidelines is this: THE REPRESENTATION OF A GRAMMATICAL ENTRY MUST MAKE IT AS EASY AS POSSIBLE FOR THE GENERATOR TO EXPLOIT ITS CONTRIBUTION IN CARRYING OUT FURTHER PLANNING. This principle responds to two concerns. First, SPUD is currently constrained to greedy or incremental search for reasons of efficiency. At each step, SPUD picks the entry whose interpretation goes furthest towards achieving its communicative goals. As the generator uses its grammar to build on these greedy choices, our principle facilitates the generator in arriving at a satisfactory overall utterance. More generally, we saw in Section 6 many characteristic uses of language in which separate lexico-syntactic elements jointly ensure needed features of communicative intent. This is an important way in which any generator needs to be able to exploit the contribution of an entry it has already used, in line with our principle.

7.1 Syntax

Our first set of guidelines describes the elementary trees that we specify as syntactic structures for lexical items (including lexical items that involve a semantically-opaque combination of words).

1. The grammar must associate each item with its observed range of complements and modifiers, in the observed orders. This constraint is common to any effort in grammar development; it is sufficiently well-understood to allow induction of LTAGs from treebanks (Chen and Vijay-Shanker, 2000; Sarkar, 2001).

2. All syntactically optional elements, regardless of interpretation, must be represented in the syntax as modifiers, using the LTAG operation of adjunction. This allows the generator to select an optional element when it is needed to achieve updates not otherwise conveyed by its provisional utterance. Recall that, in LTAG, a substitution site indicates a constituent that must be supplied syntactically to obtain a grammatical sentence; we call a constituent so provided a SYNTACTIC ARGUMENT. The alternative is to rewrite a node so as to include additional material (generally optional) specified by an auxiliary tree; we call material so provided a SYNTACTIC ADJUNCT. If optional elements are represented as syntactic adjuncts, it is straightforward to select one whenever its potential benefit is recognized. With other representations—for example, having a set of syntactic entries, each of which has a different number of syntactic arguments—the representation can result in artificial dependencies in the search space in generation, or even dead-end states in which the grammar does not offer a way to more precisely specify an ambiguous reference. To use this representation successfully, a greedy generator such as SPUD would have to anticipate how the sentence would be fleshed out later in order to select the right entry early on.

3. The desired linear surface order of complements and modifiers for an entry must be represented using hierarchies of nodes in its elementary tree. In constructions with fixed word-order (the typical case for English), the nodes we add reflect different semantic classes which tend to be realized in a particular order. In constructions with free word-order (the typical

case in many other languages), node-ordering would instead reflect the information-structure status of constituents. Introducing hierarchies of nodes to encode linear surface order decouples the generator's search space of derivations from the overt output word-order. It allows the generator to select complements and modifiers in any search order, while still realizing the complements and modifiers with their correct surface order. This is important for SPUD's greedy search; alternative designs—representing word-order in the derivation itself or in features that clash when elements appear in the wrong order—introduce dependencies into the search space for generation that make it more difficult for the generator to build on its earlier choices successfully. However, for a generator which explores multiple search paths, the more flexible search space will offer more than one path to the same final structure, and additional checks will be required to avoid duplicate results.

Because of strong parallels in natural language syntax across categories (see for example (Jackendoff, 1977)), we anticipate that these guidelines apply for all constructions in a similar way. Here we will illustrate them with verbs, a challenging first case that we have investigated in detail; other categories, particularly complex adjectives, adverbials and discourse connectives, merit further investigation.

We collected occurrences of the verbs *slide*, *rotate*, *push*, *pull*, *lift*, *connect*, *disconnect*, *remove*, *position* and *place* in the maintenance manual for the fuel system of the American F16 aircraft. We eliminated instructions involving plural and coordinated noun phrases, because SPUD does not yet have a model of plural reference, but otherwise included the remainder of the examples in our investigation. We characterized the form and meaning of the verbs in these examples in sufficient detail to allow SPUD to generate an instruction with similar syntactic and semantic frame from suitable knowledge base and communicative goals. (We did not construct such knowledge bases and goals for all examples, however.) This broad analysis was possible because each operation is described simply, consistently and precisely in the manual.

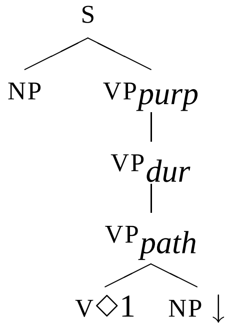
Syntactic analysis of instructions in the corpus and the application of standard tests allowed us to cluster the uses of these verbs into four syntactic classes; these classes are consistent with each verb's membership in a distinct Levin class (Levin, 1993). Differences among these classes include whether the verb lexicalizes a path of motion (*rotate*), a resulting location (*position*), or a change of state (*disconnect*); and whether a spatial complement is optional (as with the verbs just given) or obligatory (*place*). The sentences from our corpus in (66) illustrate these alternatives.

(66) a Rotate valve one-fourth turn clockwise. [Path]
b Rotate halon tube to provide access. [Path, unspecified]
c Position one fire extinguisher near aircraft servicing connection point. [Resulting location]
d Position drain tube. [Resulting location, unspecified]
e Disconnect generator set cable from ground power receptacle. [Change of state, specified source]
f Disconnect coupling. [Change of state, unspecified source]

g Place grommet on test set vacuum adapter. [Resulting location, required]

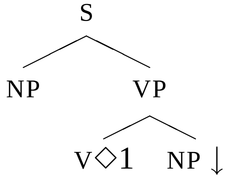
We used our guidelines to craft SPUD syntactic entries for these verbs. For example, we associate *slide* with the tree in (67). The structure reflects the optionality of the *path* constituent and makes explicit the observed characteristic order of three kinds of modifiers: those specifying path, such as *onto elbow*, which adjoin at VP_{path}; those specifying duration, such as *until it is released*, which adjoin at VP_{dur}; and those specifying purpose, such as *to uncover sealing ring*, which adjoin at VP_{purp}.

(67)



The requirements of generation in SPUD induce certain differences between our trees and other LTAG grammars for English, such as the XTAG grammar (Doran et al., 1994; The XTAG-Group, 1995), even in cases when the XTAG trees do describe our corpus. For example, the XTAG grammar represents *slide* simply as in (68).

(68)



The XTAG grammar does not attempt to encode the different orders of modifiers, nor to assign any special status to path PPs with motion verbs.

7.2 Semantic Arguments and Compositional Semantics

Recall that, to express the semantic links between multiple entries in a derivation, we associate each node in a syntactic tree with indices representing individuals. When one tree combines with another, and a node in one tree is identified with a node in the other tree, the corresponding indices are unified. Thus, the central problem of designing the compositional semantics for a given entry is to decide which referents to explicitly represent in the tree and how to distribute those referents as indices across the different nodes in the tree. (Of course, these decisions also inform subsequent specification of lexical semantics.)

We refer to the collection of all indices that label nodes in an entry as the SEMANTIC ARGUMENTS of the entry. This notion of semantic argument is clearly distinguished from the notion of syntactic argument that we used in Section 7.1 to characterize the syntactic structure of entries. Each syntactic argument position corresponds to one semantic argument (or more), since the syntactic argument position is a node in the tree which is associated with some indices: semantic arguments. However, semantic arguments need not be associated with syntactic argument positions. For example, in a verb entry, we do not have a substitution site that realizes the eventuality that the verb describes. But we treat this eventuality as a semantic argument to implement a Davidsonian account of event modifiers, cf. (Davidson, 2002). Because we count these implicit and unexpressed referents as semantic arguments, our notion is broader than that of (Candito and Kahane, 1998) and is more similar to Palmer's essential arguments (Palmer, 1990).

Our strategy for specifying semantic arguments is as follows. We always include at least one implicit argument that the structure as a whole describes; these are the MAJOR ARGUMENTS of the structure. (This is common in linguistics, e.g. (Jackendoff, 1990), and in computational linguistics, e.g. (Joshi and Vijay-Shanker, 1999).) Moreover, since complements require semantic arguments, we have found the treatment of complements relatively straightforward—we simply introduce appropriate arguments.

The treatment of optional constituents, however, is more problematic, and requires special guidelines. Often, it seems that we might express the semantic relationship between a head *h* and a modifier *m* in two ways, as schematized in (69).

(69) a $h(R,A,\ldots) \wedge m(A,\ldots)$
b $h(R,\ldots) \wedge m(R,A,\ldots)$

In (69a), we represent the head as relating its major argument *R* to another semantic argument *A*; we interpret the modifier *m* as specifying *A* further, perhaps in relation to additional entities. In this case, we must provide *A* as an index at the node where *m* adjoins. In contrast, in (69b), we interpret the modifier *m* as relating the major argument *R* of the head directly to *A*. In this case, *A* need not be a semantic argument of *h*, and we need only provide *R* as an index at the node where *m* adjoins.

We treat the case (69b) as a default, and we require specific distributional evidence before we adopt a representation such as (69a). If a class of modifiers such as *m* passes any of the three tests below, we represent the key entity *A* as a semantic argument of the associated head *h*, and include *A* as an index of the node to which *m* adjoins.

1. The PRESUPPOSITION TEST requires us to compare the interpretation of a sentence with a modifier *m*, in which the head *h* contributes an update, to the interpretation of a corresponding sentence without the modifier. If the referent *A* specified by the modifier can be identified implicitly as discourse-bound—so that the sentence without the modifier can have the same interpretation as the sentence with the modifier—then the modifier must specify *A* as a semantic argument of the head *h*. In fact, *A* must figure in the presupposition of *h*. This is only a partial diagnostic, because semantic arguments need not always be presupposed.

   (70) illustrates an application of the presupposition test for the locative modifier of the verb *disconnect*.

(70) a (Find the power cable.) Disconnect it from the power adaptor.
b (The power cable is attached to the power adaptor.) Disconnect it.

In (70b), it is understood that the power cable is to be disconnected *from the power adaptor*; the modifier in (70a) makes this explicit. Thus *disconnect* and *from the power adaptor* pass the presupposition test.

The motivation for the presupposition test is as follows. In SPUD, implicit discourse-bound references can occur in an entry *h* used for an update, only when the presupposition of *h* evokes a salient referent from the conversational record, as suggested by (Saeboe, 1996). In (70b), for example, this referent is the power adaptor and the presupposition is that the power cable is connected to it. The representation of such presuppositions must feature a variable for the referent—we might have a variable *A* for the adaptor of (70b). Accordingly, in SPUD's model of interpretation, the speaker and hearer coordinate on the value for this variable (that *A* is the power adaptor, say) by reasoning from the presupposed constraints on the value of this variable. To guarantee successful interpretation (again using greedy search), SPUD needs to be able to carry out further steps of grammatical derivation that add additional constraints on these variables. (For example, SPUD might derive (70a) from (70b) by adjoining *from the power adaptor* to describe *A*.) But this is possible only if the variable is represented as a semantic argument. Otherwise, the inclusion of an additional modifier will only introduce new discourse anaphors and so will not help the hearer to resolve the existing ones.

2. The NO-CONSTITUENT-ELLIPSIS TEST looks at the grammaticality and interpretation of cases of constituent ellipsis—certain anaphoric constructions that go proxy for a major argument of the head *h*. If modifiers in the same class as *m* cannot be varied across constituent ellipsis, then these modifiers must characterize semantic arguments other than the major argument of *h*.

   For verbs, *do so* is one case of constituent ellipsis. The temporal modifiers *at first* and *later* in (71a) (from the *Wall Street Journal*), which successfully contrast different times at which Mr. Azoff might produce films, fail the no-constituent-ellipsis test for *do so*. However, the locative PPs in the constructed variant (71b) cannot be taken to describe two alternative destinations; so they pass the no-constituent-ellipsis test.

(71) a Although Mr. Azoff won't produce films at first, it is possible that he could do so later.
b *Although Mr. Azoff won't move his operation to Hollywood, it is possible that he could do so to Toronto.

A successful test here suggests that *m* contributes a description of a referent that is not related to the event by a general semantic predicate—in other words, that *m* specifies some semantic argument. Its meaning should therefore be represented in the form $m(A,\ldots)$. For (71b), for example, the modifier is *to*, the referent is the path *P* (a semantic argument of the verb), and we can use a constraint $to(P,O)$ indicating that the path *P* goes to the landmark *O*.

A failed test here suggests that *m* directly describes a complete event. Its meaning should therefore be represented in the form $m(R,A,\ldots)$, where *m* is some relational constraint and *R* is an event variable. For *at first* in (71a), for example, we relate the event variable *E* of the main clause to an understood temporal reference point *R*: $at\text{-}first(E,R)$.

A theoretical justification for the no-constituent-ellipsis test depends on the assumption that material recovered from context in constituent ellipsis is invisible to operations of syntactic combination. (For example, the material might be supplied atomically as discourse referent, as in (Kehler and Ward, 1999; Hardt, 1999), where *do so* recovers a property or action discourse referent that has been introduced by an earlier predicate on events.) Then a phrase that describes the major argument *R* can combine with the ellipsis, but phrases that describe any another implicit referent *A* cannot; these implicit referents are syntactically invisible.

3. The TRANSFORMATION TEST looks at how modifiers are realized across different syntactic frames for *h*; it is particularly useful when *m* is headed by a closed-class item. If some frames for *h* permit *m* to be realized as a discontinuous constituent with an apparent “long-distance” dependency, then the modifier *m* specifies a semantic argument. (Note that failure of the transformation test would be inconclusive in cases where syntax independently ruled out the alternative realization.)

   For verbs, *wh*-extraction constructions illustrate the transformation test:

   (72) a What/Where did you remove the rabbit from? (A: the hat)
   b *What/*Where did you remove the rabbit at? (A: the magic show)

   In these cases, a modifier is realized effectively in two parts: *what...from* in (72a) and *what...at* in (72b). Intuitively, we have a case of extraction of the NP describing *A* from within *m*. When this is grammatical, as in (72a), it suggests that *m* specifies *A* as a semantic argument of the head; when it is not, as in (72b), the test fails.

   In LTAG, a transformation is interpreted as a relation among trees in a tree family that have essentially the same meaning and differ only in syntax. (In one formalization (Xia et al., 1998), these relationships between trees are realized as descriptions of structure to add to elementary trees.) A transformation that introduces the referent *A* in the syntax–semantics interface and relates *A* to the available referent *R* in the semantics cannot be represented this way. However, if some semantic argument *A* is referenced in the original tree, the transformed analogue to this tree can easily realize *A* differently. If we describe the source location as the semantic argument *A* in (72a) for example, the new realization involves an initial *wh*-NP substitution site describing the source *A*, and the corresponding stranded structure of the PP *from t*.

Of course, these tests are not perfect and have on occasion revealed difficult or ambiguous cases; here too, further research remains in adapting these tests to categories of constituents that did not require intensive investigation in our corpus.

We have combined these tests to designing the syntax–semantics interface for verbs in our generation grammar. In the case of *slide*, these tests show that the path of motion is a semantic argument but a syntactic modifier. (73) presents our diagnostics: extraction is good, *do so* substitution is degraded, and *slide* can make a presupposition about the path of motion that helps to identify both the object and the path.

(73) a What did you slide the sleeve onto?
b *Mary slid a sleeve onto the elbow and John did so onto the pressure sense tube.
c Slide sleeve onto elbow [acceptable in a context with many sleeves, but only one connected on a surface with the elbow].

Suppose we describe an event *A* in which *H* slides object *O* along path *P*. We label the nodes of (67) with these indices as in (74).

(74) a subject NP: *H*
b object NP: *O*
c S, VP$_{dur}$: *A*
d VP$_{purp}$: *A*, *H*
e VP$_{path}$: *A*, *O*, *P*

This labeling is motivated by patterns of modification we observed in maintenance instructions. In particular, the index *H* for (74d) allows us to represent the control requirement that the subject of the purpose clause is understood as the subject of the main sentence; meanwhile, the indices *O* and *P* for (74e) allow us to represent the semantics of path particles such as *back*; *back* presupposes an event or state preceding *A* in time in which object *O* was located at the endpoint of path *P*.

7.3 Lexical Semantics

To complete a SPUD specification, after following the methods outlined in Sections 7.1 and 7.2, we have only to specify the meanings of individual lexical items. This task always brings potential difficulties. However, the preceding decisions and the independent effects of SPUD's specifications of content, presupposition and pragmatics greatly constrain what needs to be specified.

By specifying syntax and compositional semantics already, we have determined what lexicalized derivation trees the generator will consider; this maps out the search space for generation. Moreover, our strategy for doing so keeps open as many options as possible for extending a description of an entity we have introduced; it allows entries to be added incrementally to an incomplete sentence in any order, subject only to the constraint that a head must be present before we propose to modify it. Syntactic specifications guarantee correct word order in the result, while the syntax–semantics interface ensures correct connections among the interpretations of combined elements. Thus, all that remains is to describe the communicative intent that we associate with the utterances in this search space.

The communicative intent of an utterance is made up of records for assertion, presupposition and pragmatics that depend on independent specifications from lexical items. The content condition determines the generator's strategy for contributing needed information to the hearer; the presupposition determines, inter alia, reference resolution; the pragmatics determines other contextual links. Thus we can consider these specifications separately and base each specification on clearly delineated evidence. In what follows we will describe this process for the motion verbs we studied.

We begin with the content condition. We know the kind of relationship that this condition must express from the verb's syntactic distribution (i.e., for *slide*, the frames of (66) that lexicalize an optional path of motion), and from the participants in the event identified as semantic arguments of the verb (i.e., *slide*, the event itself and its agent, object and path). To identify the particular relationship, we consider what basic information we learn from discovering that an event of this type occurred in a situation where the possibility of this event was known. For verbs in our domain, we found just four contrasts:

(75) a Whether the event merely involves a pure change of state, perhaps involving the spatial location of an object but with no specified path; e.g., *remove* but not *move*.

b Whether the event must involve an agent moving an object from one place to another along a specified path; e.g., *move* but not *remove*.

c Whether the event must involve the application of force by the agent; e.g., *push* but not *move*.

d Whether the event must be brought about directly through the agent's bodily action (and not through mechanical assistance or other indirect agency); e.g., *place* but not *position*.

Obviously, such contrasts are quite familiar from such research in lexical semantics as (Talmy, 1988; Jackendoff, 1990); they have also been explored successfully in action representation for animation (Badler et al., 1999; Badler et al., 2000), based on VerbNet representations (Kipper et al., 2000a; Kipper et al., 2000b; Dang et al., 2000).

Many sets of verbs are identical in content by these features. One such set contains the verbs *move*, *slide*, *rotate* and *turn*; these verbs contribute just that the event involves an agent moving an object along a given path. If the path is familiar, the utterance must identify it, just as the utterance must identify a familiar agent or a familiar object. Different verbs identify this path using different constraints; this is an important part of the meaning of different verbs, and we represent this meaning separately, as a presupposition.

To specify the presupposition and pragmatics of a verb, we must characterize the links that the verbs impose between the action and what is known in the context about the environment in which the action is to be performed. In some cases, these links are common across verb classes. For instance, all motion verbs presuppose a current location for the object, which they assert to be the beginning of the path traveled. In other cases, these links accompany particular lexical items; an example is the presupposition of *slide*, that the path of motion maintains contact with some surface.

In specifying these links, important evidence comes from the uses of lexical items observed in a corpus. The following illustration is representative. In the aircraft vent system, pipes may be sealed together using a sleeve, which fits snugly over the ends of adjacent pipes, and a coupling, which snaps shut around the sleeve and holds it in place. At the start of maintenance, one *removes* the coupling and *slides* the sleeve away from the junction between the pipes. Afterwards, one *(re-)positions* the sleeve at the junction and *(re-)installs* the coupling around it. In the F16 corpus, these actions are always described using these verbs.

This use of verbs reflects not only the motions themselves but also the general design and function of the equipment. For example, the verb *position* is used to describe a motion that leaves its object in some definite location in which the object will be able to perform some intended function. In the case of the sleeve, it would only be IN POSITION when straddling the pipes whose junction it seals. Identifying such distinctions in a corpus thus points to the specification required for correct lexical choice. In this case, we represent *position* as presupposing some “position” where the object carries out its intended function.

These specifications now directly control how SPUD realizes the alternation. To start, SPUD’s strategy of linking the presupposition and pragmatics to a knowledge base of shared information restricts what verbs are applicable in any microplanning task. For example, when the sleeve is moved away from the junction, we can only describe it by *slide* and not by *position*, because the presupposition of *position* is not met.

At the same time, in contexts which support the presupposition and pragmatics of several alternatives, SPUD selects among them based on the contribution to communicative intent of presupposition and pragmatics. We can illustrate this with *slide* and *position*. We can settle on a syntactic tree for each verb that best fits the context; and we have designed these trees so that either choice can be fleshed out by further constituents into a satisfactory utterance. Similarly, these items are alike in that their assertions both specify the motion that the instruction must convey to the hearer.[18] The syntax, the syntax–semantics interface, and the assertion put *slide* and *position* on an equal footing, and only the presupposition and pragmatics could distinguish the two.

With differences in presuppositions come differences in possible resolutions of discourse anaphors to discourse referents; the differences depend on the properties of salient objects in the common ground. The fewer resolutions that there are after selecting a verb, the more the verb assists the hearer in identifying the needed action. This gives a reason to prefer one verb over another. In general, we elect to specify a constraint on context as a presupposition exactly when we must model its effects on reference resolution.

In our example, general background indicates that each sleeve only has a single place where

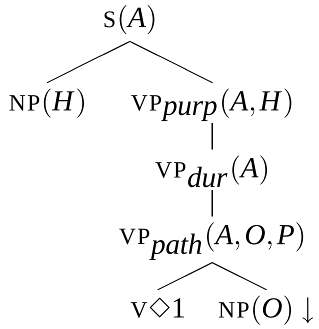
[18] Note that if the assertions were different in some relevant respect, the difference would provide a decisive reason for SPUD to prefer one entry over another. SPUD’s top priority is to achieve its updates. For example, SPUD would prefer an entry if its assertion achieved a specified update by describing manner of motion and alternative entries did not.

it belongs, at the joint; meanwhile, there may be many “way points” along the pipe to slide the sleeve to. This makes the anaphoric interpretation of *position* less ambiguous than that of *slide*; to obtain an equally constrained interpretation with *slide*, an additional identifying modifier like *into its position* would be needed. This favors *position* over *slide*—exactly what we observe in our corpus of instructions. The example illustrates how SPUD’s meaning specifications can be developed step by step, with a close connection between the semantic distinctions we introduce in lexical entries and their consequences for generation.

With differences in pragmatics come differences in the fit between utterance and context. The more specific the pragmatics the better the fit; this gives another reason to prefer one verb over another. We did not find such cases among the motion verbs we studied, because the contextual links we identified all had effects on reference resolution and thus were specified as presuppositions. However, we anticipate that pragmatics will prove important when differences in meaning involve the perspective taken by the speaker on an event, as in the contrast of *buy* and *sell*.

Appendix B details our results for the ten verbs we studied; (76) presents the final sample entry for *slide*. The tree gives the syntax for one element in the tree family associated with *slide*, with its associated semantic indices; the associated formulas describe the semantics of the entry in terms of presuppositions and assertions about the individuals referenced in the tree.

(76) a Syntax and syntax–semantics interface:



b Assertion: $move(A,H,O,P)$

c Presupposition: $start\text{-}at(P,O) \wedge surf(P)$

Of course, the corresponding entries (77) and (83) that we used in assembling concrete communicative intent for (3) in Figure 11 refine (76) only in adopting the specific syntactic and semantic refinements of an imperative use of the verb. The entries are provided as (77) and (83) in Appendix A.

8 Previous Work

In the discussion so far, we have been able to contrast SPUD with a range of research from the sentence planning literature. In particular, as catalogued in Section 6, SPUD captures the essence of techniques for referring expression generation, such as (Dale and Haddock, 1991); for syntactic choice, such as (Mathiessen, 1983; Yang et al., 1991); for lexical choice, such as (Nogier and

Zock, 1991; Elhadad et al., 1997; Stede, 1998); and for aggregation, such as (Dalianis, 1996; Shaw, 1998). At the same time, SPUD goes beyond these pipelined approaches in modeling and exploiting interactions among microplanning subtasks, and SPUD captures these efficiencies using a uniform model of communicative intent.

Our characterization of sentence planning is not so far from that of Appelt (Appelt, 1985), McDonald (McDonald, 1992) and Thomason and colleagues (Thomason et al., 1996; Thomason and Hobbs, 1997). All these researchers propose to guide use representations of interpretation to solve microplanning problems flexibly and interactively. Nevertheless, SPUD is distinguished by its uniform decision-making, and its streamlined computational representations. For example, Appelt's planning formalism includes plan-critics that can detect and collapse redundancies in sentence plans (Appelt, 1985). This framework treats subproblems in generation as independent by default; and writing tractable and general critics is hampered by the absence of abstractions like those used in SPUD to simultaneously model the syntax and the interpretation of a whole sentence. (Appelt takes a speech-act view of communicative action, and uses logic to attempt to characterize the changing mental states of interlocutors in dialogue. That means, for example, that to achieve interpretations with pragmatic overloading Appelt would have to axiomatize the reasoning by which a hearer recognizes the multiple goals behind the choice of a word.)

Meanwhile, in (McDonald, 1992), McDonald considers descriptions of events in domains which impose strong constraints on what information about events is semantically relevant. He shows that such material should and can be omitted, if it is both syntactically optional and inferentially derivable:

> FAIRCHILD Corporation (Chantilly VA) Donald E Miller was named senior vice president and general counsel, succeeding Dominic A Petito, who resigned in November, at this aerospace business. Mr. Miller, 43 years old, was previously principal attorney for Temkin & Miller Ltd., Providence RI.

Here, McDonald points out that one does not need to explicitly mention the position that Petito resigned from in specifying the resignation sub-event, since it must be the same as the one that Miller has been appointed to. Whereas McDonald adopts special-purpose module to handle this, we regard it as a special case of pragmatic overloading.

The closest approach is the work of Thomason and colleagues (Thomason et al., 1996; Thomason and Hobbs, 1997) in the interpretation-as-abduction framework (Hobbs et al., 1993). In interpretation as abduction, a speaker's communicative intention is represented as a unified inference that links up with and extends the shared information in the context; speaker and hearer coordinate their understanding of one another by drawing on matching preferences to construct matching interpretations from utterances. Abstractly, then, our two approaches share a commitment to inference as a representation of a broad notion of interpretation, and a commitment to coordinated processes that keep generation and understanding in synch. We claim that our realization of these commitments has better properties for computation and knowledge engineering. Abductive reasoning involves complex management of assumptions, and so it does not lend itself to computational techniques such as constraint-satisfaction. Abductive interpretations in particular do not distinguish among presuppositions, assertions, and pragmatic conditions, so they offer designers fewer guidelines for the formalization of linguistic resources and less control over their use in generation.

More generally, like many sentence planners, SPUD achieves a flexible association between the content input to a sentence planner and the meaning that comes out. Other researchers (Nicolov et al., 1995; Rubinoff, 1992) have assumed that this flexibility comes from a mismatch between input content and grammatical options. In SPUD, such differences arise from the referential requirements and inferential opportunities that are encountered.

Previous authors (McDonald and Pustejovsky, 1985; Joshi, 1987) have noted that TAG has many advantages for generation as a syntactic formalism, because of its localization of argument structure. (Joshi, 1987) states that adjunction is a powerful tool for elaborating descriptions. These aspects of TAGs are crucial for us; for example, lexicalization allows us to easily specify local semantic and pragmatic constraints imposed by the lexical item in a particular syntactic frame.

Various efforts at using TAG for generation (McDonald and Pustejovsky, 1985; Joshi, 1987; Yang et al., 1991; Danlos, 2000; Nicolov et al., 1995; Wahlster et al., 1991) enjoy many of these advantages. They vary in the organization of the linguistic resources, the input semantics and how they evaluate and assemble alternatives. Furthermore, (Shieber et al., 1990; Shieber, 1991; Prevost and Steedman, 1993; Hoffman, 1994) exploit similar benefits of lexicalization and localization. Our approach is distinguished by its declarative synthesis of a representation of communicative intent, which allows SPUD to construct a sentence and its interpretation simultaneously.

9 Conclusion

Most generation systems pipeline pragmatic, semantic, lexical and syntactic decisions (Reiter, 1994). With the right formalism—an explicit, declarative representation of COMMUNICATIVE INTENT—it is easier and better to construct pragmatics, semantics and syntax simultaneously. The approach elegantly captures the interaction between pragmatic and syntactic constraints on descriptions in a sentence, and the inferential interactions between multiple descriptions in a sentence. At the same time, it exploits linguistically motivated, declarative specifications of the discourse functions of syntactic constructions to make contextually appropriate syntactic choices.

Realizing a microplanner based on communicative intent involves challenges in implementation and specification. In the past (Appelt, 1985), these challenges may have made communicative-intent–based microplanning seem overwhelming. Nevertheless, in this paper, we have described an effective implementation, SPUD, that constructs representations of communicative intent through top-down LTAG derivation, logic-programming and constraint-satisfaction models of interpretation, and greedy search; and we have described a systematic, step-by-step methodology for designing generation grammars for SPUD.

With these results, the challenges that remain for the program of microplanning based on communicative intent offer fertile ground for further research. SPUD's model of interpretation omits important features of natural language, such as plurality (Stone, 2000a), discourse connectivity (Webber et al., 1999) and such defeasible aspects of interpretation as presupposition-accommodation

(Lewis, 1979). SPUD's search procedure is simplistic, and is vulnerable to stalled states where lookahead is required to recognize the descriptive effect of a combination of lexical items. (Gardent and Striegnitz, 2001) illustrate how refinements in SPUD's models of interpretation and search can lead to interesting new possibilities for NLG. At the same time, the construction of lexicalized grammars for generation with effective representations of semantics calls out for automation, using techniques that make lighter demands on developers and make better use of machine learning.

Acknowledgments

This paper has been brewing for a long time—during which the investigation has been supported by: NSF and IRCS graduate fellowships, a RuCCS postdoctoral fellowship, NSF grant NSF-STC SBR 8920230, NSF Research Instrumentation award 9818322, ARPA grant N6600194C6-043, and ARO grant DAAH04-94-G0426. We are grateful for discussion and comments from anonymous reviewers, Valerie Barr, Gann Bierner, Betty Birner, Julie Bourne, Justine Cassell, Doug DeCarlo, David DeVault, Aravind Joshi, Alistair Knott, Ellen Prince, Owen Rambow, Joseph Rosenzweig, Anoop Sarkar, Mark Steedman, Paul Tepper, Mike White and Hao Yan.

References

Appelt, D. (1985). *Planning English Sentences*. Cambridge University Press.

Badler, N., Bindiganavale, R., Allbeck, J., Schuler, W., Zhao, L., Lee, S.-J., Shin, H., and Palmer, M. (2000). Parameterized action representation and natural language instructions for dynamic behavior modification of embodied agents. In *Proceedings of the Fourth International Conference on Autonomous Agents*.

Badler, N., Palmer, M., and Bindiganavale, R. (1999). Animation control for real-time virtual humans. *Communications of the ACM*, 42(7):65–73.

Baldoni, M., Giordano, L., and Martelli, A. (1998). A modal extension of logic programming: Modularity, beliefs and hypothetical reasoning. *Journal of Logic and Computation*, 8(5):597–635.

Birner, B. (1992). *The Discourse Function of Inversion in English*. PhD thesis, Northwestern University.

Bourne, J. C. (1999). *Generating Effective Natural Language Instructions based on Agent Expertise*. PhD thesis, University of Pennsylvania.

Brachman, R., McGuinness, D., Schneider, P. P., Resnick, L. A., and Borgida, A. (1990). Living with CLASSIC: when and how to use a KL-ONE-like language. In Sowa, J., editor, *Principles of Semantic Networks*. Morgan Kaufmann.

Bratman, M. E. (1987). *Intention, Plans, and Practical Reason*. Harvard University Press.

Butt, M., King, T. H., Nino, M.-E., and Segond, F. (1999). *A Grammar Writer's Cookbook.* CSLI.

Cahill, L. and Reape, M. (1999). Component tasks in applied NLG systems. Technical Report ITRI-99-05, ITRI, University of Brighton.

Candito, M. and Kahane, S. (1998). Can the TAG derivation tree represent a semantic graph? an answer in the light of Meaning-Text Theory. In *Workshop on Tree-Adjoining Grammar and Related Formalisms (TAG+4)*.

Cassell, J. (2000). Embodied conversational interface agents. *Communications of the ACM*, 43(4):70–78.

Cassell, J., Stone, M., and Yan, H. (2000). Coordination and context-dependence in the generation of embodied conversation. In *First International Conference on Natural Language Generation*, pages 171–178.

Chen, J. and Vijay-Shanker, K. (2000). Automated extraction of TAGs from the Penn treebank. In *Proceedings of the 6th International Workshop on Parsing Technologies*, Trento, Italy.

Cheng, H. and Mellish, C. (2000). An empirical analysis of constructing non-restrictive NP components to express semantic relations. In *First International Conference on Natural Language Generation*, pages 108–115.

Clark, H. H. (1996). *Using Language*. Cambridge University Press.

Clark, H. H. and Marshall, C. R. (1981). Definite reference and mutual knowledge. In Joshi, A. K., Webber, B. L., and Sag, I., editors, *Elements of Discourse Understanding*, pages 10–63. Cambridge University Press.

Dale, R. (1992). *Generating Referring Expressions: Constructing Descriptions in a Domain of Objects and Processes*. MIT Press.

Dale, R. and Haddock, N. (1991). Content determination in the generation of referring expressions. *Computational Intelligence*, 7(4):252–265.

Dale, R. and Reiter, E. (1995). Computational interpretations of the Gricean maxims in the generation of referring expressions. *Cognitive Science*, 18:233–263.

Dalianis, H. (1996). *Concise Natural Language Generation from Formal Specifications*. PhD thesis, Royal Institute of Technology, Stockholm. Department of Computer and Systems Sciences.

Dang, H. T., Kipper, K., and Palmer, M. (2000). Integrating compositional semantics into a verb lexicon. In *COLING-2000 Eighteenth International Conference on Computational Linguistics*.

Danlos, L. (2000). G-TAG: A lexicalized formalism for text generation inspired by tree adjoining grammar. In Abeillé, A., and Rambow, O., editors, *Tree Adjoining Grammars: Formalisms, Linguistic Analysis and Processing*, pages 343–370. CSLI.

Davidson, D. (2002). The logical form of action sentences. In *Essays on Actions and Events*, pages 105–121. Clarendon Press, Oxford.

Debart, F., Enjalbert, P., and Lescot, M. (1992). Multimodal logic programming using equational and order-sorted logic. *Theoretical Computer Science*, 105:141–166.

Di Eugenio, B. and Webber, B. (1996). Pragmatic overloading in natural language instructions. *Internationl Journal of Expert Systems*, 9(2):53–84.

Doran, C., Egedi, D., Hockey, B. A., Srinivas, B., and Zaidel, M. (1994). XTAG System - a wide coverage grammar for English. In *Proceedings of COLING*, pages 922–928.

Doran, C., Hockey, B. A., Sarkar, A., Srinivas, B., and Xia, F. (2000). Evolution of the XTAG system. In Abeillé, A. and Rambow, O., editors, *Tree Adjoining Grammars: Formal, Computational and Linguistic Aspects*, pages 371–403. CSLI.

Elhadad, M., McKeown, K., and Robin, J. (1997). Floating constraints in lexical choice. *Computational Linguistics*, 23(2):195–240.

Elhadad, M. and Robin, J. (1992). Controlling content realization with functional unification grammars. In Dale, R., Hovy, E., Rösner, D., and Stock, O., editors, *Aspects of Automated Natural Language Generation: 6th International Workshop on Natural Language Generation*, Lecture Notes in Artificial Intelligence 587, pages 89–104. Springer Verlag, Berlin.

Fariñas del Cerro, L. (1986). MOLOG: A system that extends PROLOG with modal logic. *New Generation Computing*, 4:35–50.

Gardent, C. and Striegnitz, K. (2001). Generating indirect anaphora. In *Proceedings of the International Workshop on Computational Semantics*, pages 138–155.

Gildea, D. and Hockenmaier, J. (2003). Identifying semantic roles using combinatory categorial grammar. In *Empirical Methods in Natural Language Processing*.

Grice, H. P. (1957). Meaning. *The Philosophical Review*, 66(3):377–388.

Gundel, J. K., Hedberg, N., and Zacharski, R. (1993). Cognitive status and the form of referring expressions in discourse. *Language*, 69(2):274–307.

Haddock, N. (1989). *Incremental Semantics and Interactive Syntactic Processing*. PhD thesis, University of Edinburgh.

Hardt, D. (1999). Dynamic interpretation of verb phrase ellipsis. *Linguistics and Philosophy*, 22(2):187–221.

Hart, P. E., Nilsson, N. J., and Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on SSC*, 4:100–107. Correction in *SIGART Newsletter* 37:28–29.

Hirschberg, J. (1985). *A Theory of Scalar Implicature*. PhD thesis, University of Pennsylvania.

Hobbs, J., Stickel, M., Appelt, D., and Martin, P. (1993). Interpretation as abduction. *Artificial Intelligence*, 63:69–142.

Hobbs, J. R. (1985). Ontological promiscuity. In *Proceedings of ACL*, pages 61–69.

Hobbs, J. R., Stickel, M., Appelt, D., and Martin, P. (1988). Interpretation as abduction. In *Proceedings of ACL*, pages 95–103.

Hockenmaier, J., Bierner, G., and Baldridge, J. (2003). Extending the coverage of a CCG system. *Research on Language and Computation*, 1(4).

Hoffman, B. (1994). Generating context-appropriate word orders in Turkish. In *Proceedings of the Seventh International Generation Workshop*.

Horacek, H. (1995). More on generating referring expressions. In *Proceedings of the Fifth European Workshop on Natural Language Generation*, pages 43–58.

Jackendoff, R. (1977). *$\bar{X}$ Syntax: A Study of Phrase Structure*. MIT Press.

Jackendoff, R. S. (1990). *Semantic Structures*. MIT Press.

Jordan, P. W. (2000). *Influences on Attribute Selection in Redescriptions: Evidence from an Empirical Study*. PhD thesis, University of Pittsburgh.

Joshi, A. and Vijay-Shanker, K. (1999). Compositional semantics with lexicalized tree-adjoining grammar (LTAG). In *International Workshop on Computational Semantics*, pages 131–145.

Joshi, A. K. (1987). The relevance of tree adjoining grammar to generation. In Kempen, G., editor, *Natural Language Generation*, pages 233–252. Martinus Nijhoff Press, Dordrecht.

Joshi, A. K., Levy, L., and Takahashi, M. (1975). Tree adjunct grammars. *Journal of the Computer and System Sciences*, 10:136–163.

Kallmeyer, L. and Joshi, A. (1999). Factoring predicate argument and scope semantics: underspecified semantics with LTAG. In *12th Amsterdam Colloquium*.

Kamp, H. and Rossdeutscher, A. (1994). DRS-construction and lexically driven inference. *Theoretical Linguistics*, 20:97–164.

Kehler, A. and Ward, G. (1999). On the semantics and pragmatics of ‘identifier *so*’. In Turner, K., editor, *The Semantics/Pragmatics Interface from Different Points of View*, pages 233–256. Elsevier.

Kingsbury, P. and Kipper, K. (2003). Deriving verb-meaning clusters from syntactic structure. In *Workshop on Text Meaning, held in conjunction with HLT–NAACL 2003*.

Kingsbury, P. and Palmer, M. (2002). From treebank to propbank. In *Third International Conference on Language Resources and Evaluation*.

Kipper, K., Dang, H. T., and Palmer, M. (2000a). Class based construction of a verb lexicon. In *AAAI 2002 Seventeenth National Conference on Artificial Intelligence*.

Kipper, K., Dang, H. T., Schuler, W., and Palmer, M. (2000b). Building a class based verb lexicon using TAGs. In *TAG+5 Fifth International Workshop on Tree Adjoining Grammars and Related Formalisms*, pages 147–154.

Kittredge, R., Korelsky, T., and Rambow, O. (1991). On the need for domain communication knowledge. *Computational Intelligence*, 7(4):305–314.

Lascarides, A. and Asher, N. (1991). Discourse relations and defeasible knowledge. In *Proceedings of ACL*, pages 55–62.

Levelt, W. J. M. (1989). *Speaking*. MIT Press.

Levin, B. (1993). *English Verb Classes and Alternations: A preliminary investigation*. University of Chicago Press.

Lewis, D. (1979). Scorekeeping in a language game. In Bäuerle, R., Egli, U., and von Stechow, A., editors, *Semantics from Different Points of View*, pages 172–187. Springer Verlag, Berlin.

Mackworth, A. (1987). Constraint satisfaction. In Shapiro, S., editor, *Encyclopedia of Artificial Intelligence*, pages 205–211. John Wiley and Sons.

Mathiessen, C. M. I. M. (1983). Systemic grammar in computation: the Nigel case. In *Proceedings of EACL*, pages 155–164.

McDonald, D. (1992). Type-driven suppression of redundancy in the generation of inference-rich reports. In Dale, R., Hovy, E., Rösner, D., and Stock, O., editors, *Aspects of Automated Natural Language Generation: 6th International Workshop on Natural Language Generation*, Lecture Notes in Artificial Intelligence 587, pages 73–88. Springer Verlag, Berlin.

McDonald, D. D. and Pustejovsky, J. D. (1985). TAG’s as a grammatical formalism for generation. In *Proceedings of ACL*, pages 94–103.

Mellish, C., O'Donnell, M., Oberlander, J., and Knott, A. (1998). An architecture for opportunistic text generation. In *9th International Workshop on Natural Language Generation*.

Mellish, C. S. (1985). *Computer Interpretation of Natural Language Descriptions*. Ellis Horwood, Chichester, UK.

Meteer, M. W. (1991). Bridging the generation gap between text planning and linguistic realization. *Computational Intelligence*, 7(4):296–304.

Miller, D., Nadathur, G., Pfenning, F., and Scedrov, A. (1991). Uniform proofs as a foundation for logic programming. *Annals of Pure and Applied Logic*, 51:125–157.

Nicolov, N., Mellish, C., and Ritchie, G. (1995). Sentence generation from conceptual graphs. In G. Ellis, R. Levinson, W. R. and Sowa, F., editors, *Conceptual Structures: Applications, Implementation and Theory (Proceedings of Third International Conference on Conceptual Structures)*, pages 74–88. Springer.

Nilsson, N. (1971). *Problem-solving Methods in Artificial Intelligence*. McGraw-Hill, New York.

Nogier, J. and Zock, M. (1991). Lexical choice as pattern-matching. In Nagle, T., Nagle, J., Gerholz, L., and Eklund, P., editors, *Current Directions in Conceptual Structures Research*. Springer.

Palmer, M. (1990). *Semantic Processing for Finite Domains*. Cambridge Univeristy Press.

Pereira, F. C. N. and Shieber, S. M. (1987). *Prolog and Natural Language Analysis*. CSLI.

Pollack, M. (1991). Overloading intentions for efficient practical reasoning. *Noûs*, 25:513–536.

Pollack, M. E. (1992). The uses of plans. *Artificial Intelligence*, 57:43–68.

Prevost, S. and Steedman, M. (1993). Generating contextually appropriate intonation. In *Proceedings of EACL*, pages 332–340.

Prince, A. and Smolensky, P. (1997). Optimality: From neural networks to universal grammar. *Science*, 275:1604–1610.

Prince, E. (1986). On the syntactic marking of presupposed open propositions. In *Proceedings of the 22nd Annual Meeting of the Chicago Linguistic Society*, pages 208–222, Chicago. CLS.

Rambow, O. and Korelsky, T. (1992). Applied text generation. In *Applied Natural Language Processing Conference*, pages 40–47.

Reiter, E. (1994). Has a consensus NL generation architecture appeared, and is it psycholinguistically plausible? In *Seventh International Workshop on Natural Language Generation*, pages 163–170.

Reiter, E. and Dale, R. (2000). *Building Natural Language Generation Systems*. Cambridge University Press.

Rubinoff, R. (1992). Integrating text planning and linguistic choice by annotating linguistic structures. In Dale, R., Hovy, E., Rösner, D., and Stock, O., editors, *Aspects of Automated Natural Language Generation: 6th International Workshop on Natural Language Generation*, Lecture Notes in Artificial Intelligence 587, pages 45–56. Springer Verlag, Berlin.

Saeboe, K. J. (1996). Anaphoric presuppositions and zero anaphora. *Linguistics and Philosophy*, 19(2):187–209.

Sarkar, A. (2001). Applying co-training methods to statistical parsing. In *Proceedings of the North Americal Association for Computational Linguistics*, pages 175–182.

Schabes, Y. (1990). *Mathematical and Computational Aspects of Lexicalized Grammars*. PhD thesis, Computer Science Department, University of Pennsylvania.

Shaw, J. (1998). Clause aggregation using linguistic knowledge. In *Ninth International Workshop on Natural Language Generation*, pages 138–148.

Shieber, S., van Noord, G., Pereira, F., and Moore, R. (1990). Semantic-head-driven generation. *Computational Linguistics*, 16:30–42.

Shieber, S. M. (1991). A uniform architecture for parsing and generation. In *Proceedings of International Conference on Logic Programming*, pages 614–619.

Stede, M. (1998). A generative perspective on verb alternations. *Computational Linguistics*, 24(3):401–430.

Steedman, M. (1997). Temporality. In van Benthem, J. and ter Meulen, A., editors, *Handbook of Logic and Language*, pages 895–935. Elsevier.

Stone, M. (1998). *Modality in Dialogue: Planning, Pragmatics and Computation*. PhD thesis, University of Pennsylvania.

Stone, M. (1999). Indefinite information in modal logic programming. Technical Report RUCCS Report 56, Rutgers University. Submitted.

Stone, M. (2000a). On identifying sets. In *First International Confernence on Natural Language Generation*, pages 116–123.

Stone, M. (2000b). Towards a computational account of knowledge, action and inference in instructions. *Journal of Language and Computation*, 1:231–246.

Stone, M. (2002). Lexicalized grammar 101. In *ACL Workshop on Effective Tools and Methodologies for Teaching NLP and CL*, pages 76–83.

Stone, M. (2003a). Communicative intentions and conversational processes in human-human and human-computer dialogue. In Trueswell, J. and Tanenhaus, M., editors, *World-Situated Language Use: Psychological, Linguistic and Computational Perspectives on Bridging Product and Action Traditions*. MIT Press.

Stone, M. (2003b). Natural language generation in computational theories of conversation. *Submitted*.

Stone, M., Bleam, T., Doran, C., and Palmer, M. (2000). Lexicalized grammar and the description of motion events. In *TAG+5: Workshop on Tree-Adjoining Grammar and Related Formalisms*, pages 199–206.

Stone, M. and Doran, C. (1996). Paying heed to collocations. In *Proceedings of the International Natural Language Generation Workshop*, pages 91–100.

Stone, M. and Doran, C. (1997). Sentence planning as description using tree-adjoining grammar. In *Proceedings of ACL*, pages 198–205.

Stone, M. and Webber, B. (1998). Textual economy through close coupling of syntax and semantics. In *Proceedings of International Natural Language Generation Workshop*, pages 178–187.

Talmy, L. (1988). Force dynamics in language and cognition. *Cognitive Science*, 12:49–100.

The XTAG-Group (1995). A Lexicalized Tree Adjoining Grammar for English. Technical Report IRCS 95-03, University of Pennsylvania. Updated version available at http://www.cis.upenn.edu/˜xtag/tr/tech-report.html.

Thomason, R. H. (1990). Accommodation, meaning and implicature. In Cohen, P. R., Morgan, J., and Pollack, M. E., editors, *Intentions in Communication*, pages 325–363. MIT Press.

Thomason, R. H., Hobbs, J., and Moore, J. (1996). Communicative goals. In *ECAI Workshop on Gaps and Bridges: New Directions in Planning and Natural Language Generation*.

Thomason, R. H. and Hobbs, J. R. (1997). Interrelating interpretation and generation in an abductive framework. In *AAAI Fall Symposium on Communicative Action*.

USAF (1988). *Organizational Maintenance Job Guide (Fuel System Distribution, USAF Series F-16C/D Aircraft)*. Technical Order Manual. United States Air Force.

van der Sandt, R. (1992). Presupposition projection as anaphora resolution. *Journal of Semantics*, 9(2):333–377.

Vijay-Shanker, K. (1987). *A Study of Tree Adjoining Grammars*. PhD thesis, Department of Computer and Information Science, University of Pennsylvania.

Wahlster, W., André, E., Bandyopadhyay, S., Graf, W., and Rist, T. (1991). WIP: The coordinated generation of multimodal presentations from a common representation. In Stock, O., Slack, J., and Ortony, A., editors, *Computational Theories of Communication and their Applications*. Berlin: Springer Verlag.

Walker, M. A. (1993). *Informational redundancy and resource bounds in dialogue*. PhD thesis, Department of Computer & Information Science, University of Pennsylvania. Institute for Research in Cognitive Science report IRCS-93-45.

Wanner, L. and Hovy, E. (1996). The HealthDoc sentence planner. In *Seventh International Workshop on Natural Language Generation*, pages 1–10.

Ward, G. (1988). *The Semantics and Pragmatics of Preposing*. Outstanding Dissertations in Linguistics, Garland.

Webber, B., Knott, A., Stone, M., and Joshi, A. (1999). Discourse relations: A structural and presuppositional account using lexicalised TAG. In *Proceedings of ACL*, pages 41–48.

Webber, B. L. (1988). Tense as discourse anaphor. *Computational Linguistics*, 14(2):61–73.

Webber, B. L., Carberry, S., Clarke, J. R., Gertner, A., Harvey, T., Rymon, R., and Washington, R. (1998). Exploiting multiple goals and intentions in decision support for the management of multiple trauma: A review of the TraumAID project. *Artificial Intelligence*, 105:263–293.

Xia, F., Palmer, M., Vijay-Shanker, K., and Rosenzweig, J. (1998). Consistent grammar development using partial tree-descriptions for lexicalized tree-adjoining grammars. In *Workshop on Tree-Adjoining Grammar and Related Formalisms (TAG+4)*.

Yan, H. (2000). Paired speech and gesture generation in embodied conversational agents. Master's thesis, Media Lab, MIT.

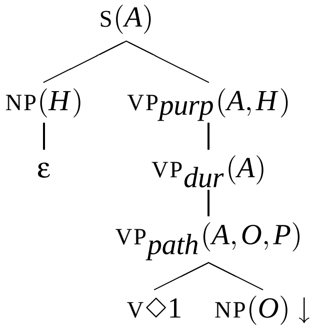
Yang, G., McCoy, K. F., and Vijay-Shanker, K. (1991). From functional specification to syntactic structures: systemic grammar and tree-adjoining grammar. *Computational Intelligence*, 7(4):207–219.

A Instruction Grammar Fragment

A.1 Syntactic Constructions

(77) a NAME: axnpVnpopp
b PARAMETERS: A,H,O,P,S
c PRAGMATICS: $obl(S,H)$

d TREE:

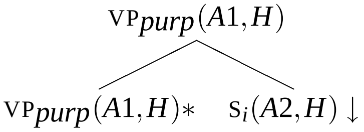


(78) a NAME: bvpPsinf

b PARAMETERS: $A1, H, A2$

c PRAGMATICS: —

d TREE:

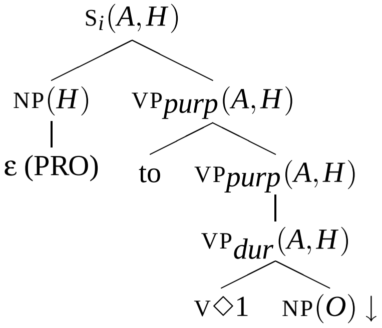


(79) a NAME: anpxVinp

b PARAMETERS: A, H, O

c PRAGMATICS: —

d TREE:

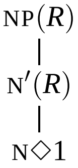


(80) a NAME zeroDefNP

b PARAMETERS: R

c PRAGMATICS: $\textit{zero-genre} \wedge \textit{def}(R)$

d TREE:

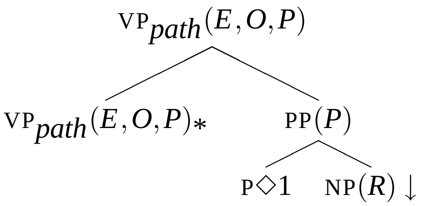


(81) a NAME: bvpPnp

b PARAMETERS: E, O, P, R

c PRAGMATICS: $\textit{zero-genre} \wedge \textit{def}(R)$

d TREE:

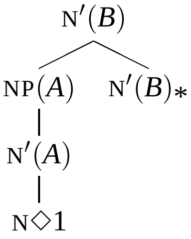


(82) a NAME: bNnn

b PARAMETERS: A,B

c PRAGMATICS: $def(A)$

d TREE:



A.2 Lexical Entries

(83) a NAME: slide

b PARAMETERS: A,H,O,P,S

c CONTENT: $move(A,H,O,P) \wedge next(A)$

d PRESUPPOSITION: $start\text{-}at(P,O) \wedge surf(P) \wedge partic(S,H)$

e PRAGMATICS: —

f TARGET: S(A) [complement]

g TREE LIST: axnpVnpopp(A,H,O,P,S)

(84) a NAME: ⟨purpose⟩

b PARAMETERS: $A1,H,A2$

c CONTENT: $purpose(A1,A2)$

d PRESUPPOSITION: —

e PRAGMATICS: —

f TARGET: VP$_2(A1,H)$ [modifier]

g TREE LIST: bvpPsinf$(A1,H,A2)$

(85) a NAME: uncover

b PARAMETERS: A,H,O

c CONTENT: $uncover(A,H,O)$

d PRESUPPOSITION: —

e PRAGMATICS: —

f TARGET: S$_i(A,H)$

g TREE LIST: anpxVinp(A,H,O)

(86) a NAME: sealing-ring
b PARAMETERS: N
c CONTENT: $sr(N)$
d PRESUPPOSITION: —
e PRAGMATICS: —
f TARGET: NP(N) [complement]
g TREE LIST: zerodefnptree(N)

(87) a NAME: coupling-nut
b PARAMETERS: N
c CONTENT: $cn(N)$
d PRESUPPOSITION: —
e PRAGMATICS: —
f TARGET: NP(N) [complement]
g TREE LIST: zerodefnptree(N)

(88) a NAME: onto
b PARAMETERS: E,O,P,R
c CONTENT: $end\text{-}on(P,R)$
d PRESUPPOSITION: —
e PRAGMATICS: —
f TARGET: VP$_{path(E,O,P)}$ [modifier]
g TREE LIST: bvpPnp(E,O,P,R)

(89) a NAME: elbow
b PARAMETERS: N
c CONTENT: $el(N)$
d PRESUPPOSITION: —
e PRAGMATICS: —
f TARGET: NP(N) [complement]
g TREE LIST: zerodefnptree(N)

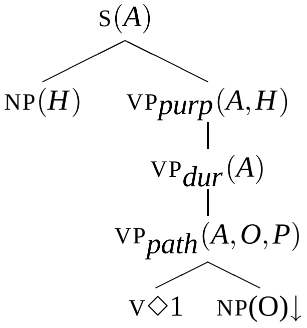
(90) a NAME: fuel-line
b PARAMETERS: N,R,X
c CONTENT: $fl(N) \wedge nn(R,N,X)$
d PRESUPPOSITION: —
e PRAGMATICS: —
f TARGET: N$'(R)$ [modifier]
g TREE LIST: bNnn(N)

B Motion Verb Entries

B.1 Pure Motion Verbs

The verbs *slide*, *rotate*, *turn*, *push*, *pull*, and *lift* all share a use in which they describe an event A in which some agent H moves an object O along a path P. Our analysis of this use was presented in detail in Section 7. (91) gives the syntactic frame for this class.

(91)



Semantically, *slide*, *rotate* and *turn* all assert simple motions; the verbs differ in that *slide* presupposes motion along a surface while *turn* presupposes a circular or helical path around an axis by which an object can pivot and *rotate* presupposes a circular path around an axis through the center of an object. (92) represents this.

(92) a slide: assert $move(A,H,O,P)$; presuppose $start\text{-}at(P,O) \wedge surf(P)$
b turn: assert $move(A,H,O,P)$; presuppose $start\text{-}at(P,O) \wedge around(P,X) \wedge pivot(O,X)$
c rotate: assert $move(A,H,O,P)$; presuppose $start\text{-}at(P,O) \wedge around(P,X) \wedge center(O,X)$

The verbs *push*, *pull* and *lift* involve force as well as motion; they differ in presuppositions about the direction of force and motion: for *push*, it is away from the agent; for *pull*, it is towards the agent; *lift* has an upward component:

(93) a push: assert $forced\text{-}move(A,H,O,P)$; presuppose $start\text{-}at(P,O) \wedge away(P,H)$
b pull: assert $forced\text{-}move(A,H,O,P)$; presuppose $start\text{-}at(P,O) \wedge towards(P,H)$
c lift: assert $forced\text{-}move(A,H,O,P)$; presuppose $start\text{-}at(P,O) \wedge upwards(P)$

B.2 Pure Change-of-state Verbs

This category of verbs describes an event A in which an agent H changes of state of an object O; these verbs appeal to a single optional semantic argument U which helps to specify what the change of state is. Examples of this class are *remove [from U]*, *disconnect [from U]* and *connect [to U]*; U is a landmark object and the change-of-state involves a spatial or connection relation between O and U.

Our diagnostic tests give a number of reasons to think of the parameter U as a semantic argument that is referenced in the tree but described by syntactic adjuncts. Here are illustrations of these tests for the case of *disconnect*. It is possible to extract from it, and impossible to supply it by *do so* substitution.

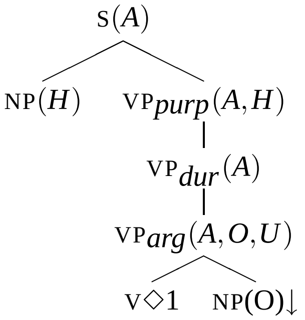
(94) a What did you disconnect the cable from ε?
b ?Mary disconnected a coupling from system A, and John did so from system B.

It is possible to take the initial connection between O and U as presupposed, and to factor in this constraint in identifying O and U. Thus, with many systems and couplings, we might still find:

(95) Disconnect the coupling from system A.

These considerations lead to the syntactic frame of (96).

(96)



Note that syntactic features can allow the verb to determine which preposition is used to specify the optional argument. That is, we can use lexical entries for verbs that indicate that they impose feature-value constraints on the syntactic features of the anchor V◇ node.

In order to characterize the semantics of change-of-state verbs, we introduce a predicate $\textit{caused-event}(A,H,O)$ indicating that A is an event in which H has a causal effect on O; and an operator $\textit{result}(A,p)$ indicating that the proposition p holds in the state that results from doing A. (For more on this ontology, see (Steedman, 1997).) (97) uses this notation to describe *connect*, *disconnect* and *remove*.

(97) a connect: assert $\textit{caused-event}(A,H,O) \wedge \textit{result}(A,\textit{connected}(O,U))$; presuppose $\textit{free}(O,U)$
b disconnect: assert $\textit{caused-event}(A,H,O) \wedge \textit{result}(A,\textit{free}(O,U))$; presuppose $\textit{connected}(O,U)$
c remove: assert $\textit{caused-event}(A,H,O) \wedge \textit{result}(A,\textit{free}(O,U))$; presuppose $\textit{dependent}(O,U)$

That is, *connecting* causes O to be connected to the optional argument U where O is presupposed to be presently spatially independent of, or *free* of, U; *disconnecting*, conversely, causes O to be *free* of U, where O is presupposed to be connected to U. Finally, *remove* is more general than *disconnect*. It presupposes only that there is some *dependent* spatial relation between O and U; O may be attached to U, supported by U, contained in U, etc.

B.3 Near-motion Verbs

Distinct from motion verbs and ordinary change-of-state verbs is a further class which we have called near-motion verbs: near-motion verbs are change-of-state verbs that encode a spatial change by evoking the final location where an object comes to rest. Semantically, they involve arguments A, H, O, and L—the fourth, spatial argument L represents a spatial configuration rather than a path (as in the case of motion verbs). The canonical near-motion verb is *position*; others are *reposition* and *install*. According to our judgments, *turn* and *rotate* can be used as near-motion verbs as well as genuine motion verbs, whereas *slide*, *push*, *pull* and *lift* cannot.

Now, whenever there is a change of location, there must be motion (in our domain); and whenever an object moves to a new place, there is a change of location. This semantic correspondence between motion verbs and near-motion verbs is mirrored in similar syntactic realizations with prepositional phrases that describe a final location. So we find both:

(98) a Push the coupling on the sleeve.
b Position the coupling on the sleeve.

The difference between motion verbs and near-motion verbs is that motion verbs permit an explicit description of the PATH the object takes during the motion, while near-motion verbs do not:

(99) a Push the coupling to the sleeve.
b *Position the coupling to the sleeve.

Another way to substantiate the contrast is to consider the interpretation of ambiguous modifiers. In (100a), *downward* modifies the path by describing the direction of motion in the event. In (100b), with the near-motion verb, this path interpretation is not available: the reading of *downward* instead is that it describes the final orientation of the object that is manipulated.

(100) a Push handle downward.
b Position handle downward.

These readings are paraphrased in (101).

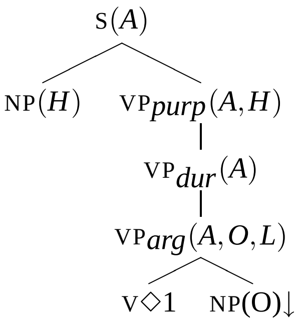
(101) a Push handle in a downward direction.
b Position handle so that it is oriented downward.

The natural *wh*-questions associated with the two constructions are also different:

(102) a { In which direction, *How } did you push the handle? Downward.
b { *In which direction, How } did you position the handle? Downward.

(103) schematizes the syntax of near-motion verbs.

(103)



Like motion verbs, near-motion verbs share a common assertion—there is an event *A* of *H* acting on *O* whose result is that *O* is located at place *L*. The differences among near-motion verbs lie in their presuppositions: *position* presupposes that *L* is a position in which *O* will be able to perform its intended function, as in (104a); *reposition* further presupposes a state preceding *A* where *O* was located at *L*—we write this as $back(A,O,L)$ in (104b); finally, *install* presupposes that the spatial position for *O* is one which fastens *O* tightly, as in (104c).

(104) a position: assert $\textit{caused-event}(A,H,O) \wedge result(A,loc(L,O))$; presuppose $\textit{position-for}(L,O)$

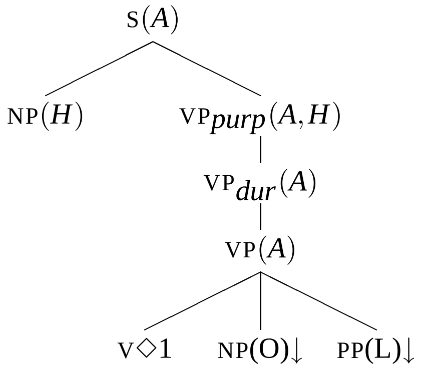
b reposition: assert $\textit{caused-event}(A,H,O) \wedge result(A,loc(L,O))$; presuppose $\textit{position-for}(L,O) \wedge back(A,O,L)$

c install: assert $\textit{caused-event}(A,H,O) \wedge result(A,loc(L,O))$; presuppose $\textit{position-for}(L,O) \wedge fastening(L,O)$

B.4 Put Verbs

Closely related to the near-motion verbs are the *put* verbs. These differ from near-motion verbs only in that *put* verbs take the configuration PP as a syntactic complement—rather than as an optional syntactic modifier.

(105)



Verbs in this class include not only *put*, but also *place*.

(106) a put: assert $\textit{caused-event}(A,H,O) \wedge result(A,loc(L,O))$

b place: assert $\textit{body-caused-event}(A,H,O) \wedge result(A,loc(L,O))$; presuppose $\textit{place-for}(L,O)$

Note that a placement must be performed by hand; the presupposition that *L* be a *place* for *O* signifies that *O*'s specific location at *L* is required for the success of future actions or events. (*Place* contrasts with *position* in that places depend on the action of an agent on the object in a particular activity whereas positions are enduring regions that depend on the functional properties of the object itself; contrast *working place* and *working position*.)