Proceedings of the Ninth National Conference on Artificial Intelligence, 1991, pages 117–124.

A Cognitively Plausible Approach to Understanding Complex Syntax

Claire Cardie and **Wendy Lehnert**
Department of Computer and Information Science
University of Massachusetts
Amherst, MA 01003
E-mail: cardie@cs.umass.edu, lehnert@cs.umass.edu

Abstract

This paper describes a cognitively plausible mechanism for systematically handling complex syntactic constructions within a semantic parser. More specifically, we show how these constructions are handled without a global syntactic grammar or syntactic parse tree representations and without sacrificing the benefits of semantically-oriented parsing. We evaluate the psychological validity of our architecture and conclude that it is a plausible computational model of human processing for an important class of embedded clause constructions. As a result, we achieve robust sentence processing capabilities not found in other parsers of its class.

Introduction

People seem to understand syntactically complex sentences without noticeable effort. Consider, for example, the following sentences:

(a) John asked Bill to eat the leftovers.

(b) That's the gentleman that the woman invited to go to the show.

(c) That's the gentleman that the woman declined to go to the show with.

Recent experiments in psycholinguistics show that human processing of complicated nested clause constructions like (a) through (c) is quite efficient [Fodor, 1989] and there is documented evidence that children understand these constructs by the age of ten [Chomsky, 1969].

Embedded clause constructions have consistently been troublesome for natural language processing systems, however. Understanding them requires that the parser infer the existence of an invisible or phonetically null constituent in the embedded clause and then associate the missing constituent with an antecedent phrase that may be arbitrarily distant from it. In (a), for example, the parser should infer that "Bill" is the phonetically null subject of "eat"; in (b), "gentleman" is the direct object of "invited" as well as the subject of "go"; and in (c), "woman" is the phonetically null subject of "go" while "gentleman" is the prepositional object of "with".

Syntactically-oriented parsers typically handle embedded clauses using context-free grammars and similar formalisms that, in theory, easily conquer the recursive structure of these constructs (see, for example, [Kay, 1980]). In practice, however, avoiding massive ambiguity while still allowing the hypothesis of missing constituents is difficult and the problem of finding the correct antecedent remains even when the syntactic structure has been determined. In addition, this class of parser often focuses on producing just a syntactic representation of the input. Semantically-oriented parsers, on the other hand, ([Riesbeck, 1975], [Birnbaum and Selfridge, 1981], [Riesbeck and Martin, 1985], [Wilks *et al.*, 1985], and [Cullingford, 1986]) produce a semantic representation of the input but traditionally avoid syntactically complicated sentences altogether.[1]

This paper describes a cognitively plausible mechanism for systematically handling complex syntactic constructions within a semantic parser called CIRCUS [Lehnert, 1990]. Through the use of this mechanism, CIRCUS achieves the desired balance between syntactic and semantic concerns during sentence processing and does so without a global syntactic grammar, without syntactic parse tree representations, without massive syntactic ambiguity, and without sacrificing the benefits of semantically-oriented parsing. Instead, we define a small number of lexically-indexed control kernels (LICKs) for processing embedded clause constructions and allow individual words to selectively trigger the LICK that will correctly handle the current clause. We also evaluate the psychological validity of the LICK processing mechanism and conclude that it is a plausible computational model of human process-

[1] One possible exception is [Lytinen, 1984].

ing for nested clause constructions. As a result, CIRCUS achieves robust sentence processing capabilities not found in other parsers of its class.

Because of length limitations and because the majority of recent psycholinguistic studies of embedded clauses have addressed wh-constructions, we will focus our attention on wh-phrases. We would like to make clear, however, that we use LICKs to understand additional classes of nested clause constructions. In particular, the mechanism handles sentential complements (e.g., "John thought Mary broke the toy"), infinitive complements (e.g., "John asked Bill to eat the leftovers"), and interactions between wh-phrases and complement clauses (e.g., "John asked Bill what to eat").

The remainder of the paper is organized into four sections. The next section presents an overview of the syntactic and semantic processing in CIRCUS. It is followed by a brief introduction to the LICK formalism that handles nested clause constructions. The last sections examine CIRCUS' processing of wh-constructions and then evaluate it with respect to data from recent experiments in psycholinguistics that show how people process these constructs.

An Overview of Syntax and Semantics in CIRCUS

CIRCUS [Lehnert, 1990] is a conceptual analyzer that produces a semantic case frame representation of an input sentence using a stack-oriented control for syntactic processing and a marker-passing mechanism for predictive preference semantics[2]. CIRCUS has been used successfully to provide natural language processing capabilities for a variety of projects including the analysis of citation sentences in research papers [Lehnert *et al.*, 1990] and understanding wire service texts about Latin American terrorism[3]. Although space does not permit us to give a full technical description of CIRCUS, this section presents the overall parser design.

In the tradition of conceptual analyzers, CIRCUS' syntactic component produces no parse tree of the input and employs no global syntactic grammar. It is based on the McEli parser [Schank and Riesbeck, 1981] and uses lexically-indexed local syntactic knowledge to segment incoming text into noun phrases, prepositional phrases, and verb phrases. These constituents are stored in global buffers that track the subject, verb, direct object, indirect object, and prepositional phrases of a sentence. Because we restrict the buffer contents to simple syntactic structures with a strongly "local" sense of the sentence, larger constituents like clauses are not explicitly recognized by the syntactic component.

Figure 1, for example, depicts the state of the McEli syntactic module after processing the phrase "John brought". McEli recognizes "John" as the subject

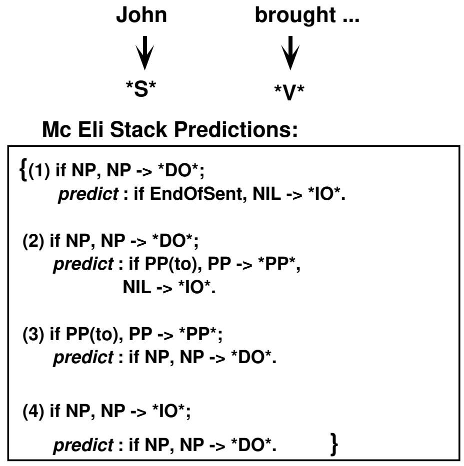

Figure 1: **McEli Status After "John brought..."**

(*S*) and "brought" as the verb (*V*). In addition, the current McEli stack contains a single packet encoding the syntactic expectations associated with "brought"[4]. This verb predicts (1) a direct object, (2) a direct object followed by a "to" prepositional phrase, (3) a "to" prepositional phrase followed by a direct object, or (4) an indirect object followed by a direct object. If the next word in the sentence were the noun phrase "Mary", for example, McEli would assign "Mary" to *both* the direct object and the indirect object buffers and update its stack of syntactic expectations. These new predictions resolve the momentary syntactic ambiguity by overwriting the contents of either *DO* or *IO* depending on the next phrase in the sentence.

As soon as McEli recognizes a syntactic constituent, that constituent is made available to the predictive semantics module (PSM). PSM is responsible for making case role assignments. In CIRCUS, this consists of top-down slot-filling for any active semantic case frames. Whenever a syntactic constituent becomes available in one of the global buffers, PSM examines any active case frame that expects a slot filler in that buffer. PSM then fills the slot if the constituent satisfies the slot's semantic constraints. CIRCUS allows both hard and soft slot constraints. A hard constraint is a predicate

[2] CIRCUS also employs a numerical relaxation algorithm to perform bottom-up insertion of unpredicted slots into case frames. This module is not important for the purposes of this paper, however.

[3] CIRCUS was selected as one of about a dozen systems to participate in the DARPA-sponsored Third Message Understanding System Evaluation and Message Understanding Conference (MUC-3).

[4] Each prediction in a packet is called a request. Whenever one request in the topmost packet on the stack is satisfied, the entire packet containing the request is popped from the stack and all subsequent predictions associated with the request are pushed onto the stack in a new packet.

that *must* be satisfied. In contrast, a soft constraint defines a preference for a slot filler rather than a predicate that blocks slot-filling when it is not satisfied. Consider, for example, the semantic case frame for a PTRANS event triggered by the word "brought" in the phrase "John brought" (see Figure 2).[5] The case frame

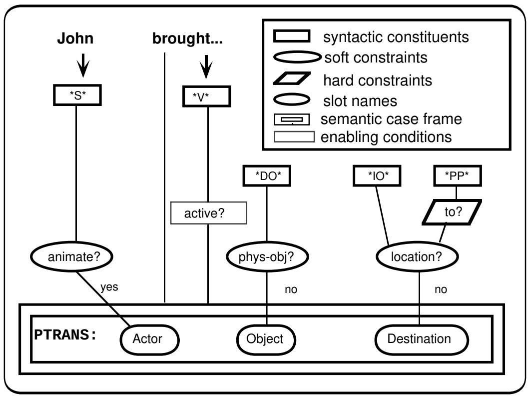

Figure 2: **PSM Status After "John brought..."**

definition indicates the mapping between surface constituents and case frame slots: subject → Actor, direct object → Object, prepositional phrase or indirect object → Destination.[6] In addition, it depicts the hard and soft constraints associated with each slot. Namely, the Actor should be animate, the Object should be a physical object, the Destination should be a location, and the prepositional phrase filling the Destination slot must begin with the preposition "to".[7] At this point in the parse, PSM successfully fills the Actor slot with "John" because "John" is the subject of the sentence and is animate. All of the other slots in the PTRANS frame remain empty.

When a frame satisfies certain instantiation criteria, PSM "freezes" the case frame with its assigned slot fillers. Any instantiated case frames then become part of the semantic representation CIRCUS derives for the sentence. Figure 3, for example, shows the PTRANS case frame instantiation returned by CIRCUS after parsing "John brought Mary to Manhattan".

PTRANS

Actor **: John**
Object **: Mary**
Destination **: Manhattan**

Figure 3: **Semantic Case Frame for "John brought Mary to Manhattan"**

Lexically-Indexed Control Kernels

When sentences become more complicated, we have to "partition" the stack processing in a way that recognizes embedded syntactic structures as well as conceptual dependencies. This is accomplished with lexically-indexed control kernels (LICKs). We view the top-level McEli stack as a single control kernel whose expectations and binding instructions change in response to specific lexical items as we move through the sentence. When we come to a subordinate clause, the top-level kernel creates a subkernel that takes over to process the interior clause. In other words, when a subordinate clause is first encountered, the parent LICK spawns a child LICK, passes control over to the child, and later recovers control from the child when the subordinate clause is completed.

Each control kernel essentially creates a new parsing environment with its own set of bindings for the syntactic buffers, its own copy of the main McEli stack, and its own predictive semantics module. To understand the behavior of multiple LICKs, we need only specify rules for passing control among LICKs and rules for passing variable bindings across LICKs:

<u>Inter-LICK Control Rules</u>:

1. An existing LICK can create a new LICK at which time control moves from the parent LICK to the child LICK.
2. When a child LICK relinquishes control, control reverts back to the parent LICK.

<u>Inter-LICK Communication Rules</u>:

1. When moving from a parent LICK to a child LICK, all syntactic buffers in the child LICK can be initialized by the parent LICK.
2. When moving from a child LICK to a parent LICK, the only buffer that can be initialized or reassigned in the parent LICK is the *LB* buffer.

LB (lick buffer) is a special syntactic buffer used only for inter-LICK communication. Typically, the conceptual representation for an entire subordinate clause is stored in *LB* until it can be incorporated into the representation being constructed by a parent control kernel.

LICKs, then, embody the basic control mechanism of ATN's [Woods, 1970] but enforce a much stricter set of communication rules. In addition, CIRCUS' use of LICKs differs tremendously from the pervasive recursion of ATN's — CIRCUS employs the LICK mechanism only at the clause level and selectively triggers the mechanism via lexically-indexed signals. Unlike ATN's, the parsing of constituents within a clause remains deterministic and strictly bottom-up.

[5] PTRANS is a primitive act in conceptual dependency describing a physical transfer (see [Schank, 1975]). The PTRANS case frame actually has a fourth slot — the original location or Source of the object. For the purposes of this example, however, we will ignore this slot.

[6] As in lexical-functional grammar (LFG) [Bresnan, 1982], a different case frame definition would be needed to handle a passive sentence construction.

[7] This is a hard constraint.

The next section walks through a specific example using LICKs to parse a sentence containing an embedded wh-phrase. It is followed by an evaluation of the psychological validity of this mechanism.

Understanding Wh-Constructions

In this section we show how sentences containing embedded wh-phrases are handled by local syntactic predictions and interactions between cooperating LICKs. Consider the following sentence:

(1) The policeman saw the boy who the crowd at the party accused of the crime.

Figure 4 shows the state of the parser after the word "who". The LICK processing the main clause has triggered a semantic case frame for SAW and has successfully filled its Actor and Object slots. In addition, the lexicon entry for "who" indicates that processing of the main clause should be temporarily suspended and a child LICK spawned. Because the antecedent for "who" can bind to one of four possible syntactic constituents within the subordinate clause, CIRCUS initializes each of the child *S*, *DO*, *IO*, and *PP* buffers with "boy". When the child completes a semantic case frame instantiation, only the buffer associated with the *gap* (i.e., the missing or phonetically null constituent) should hold the *filler* (i.e., the antecedent). The other buffers initialized with the antecedent will either be overwritten with actual phrases from the embedded clause or ruled out as possible gaps by syntactic information associated with the verb. In any case, few case frame definitions will access all four buffers. As indicated in Figure 4, "who" also sets up syntactic predictions for either a verb phrase or a subject-verb sequence before passing control to the embedded clause LICK.

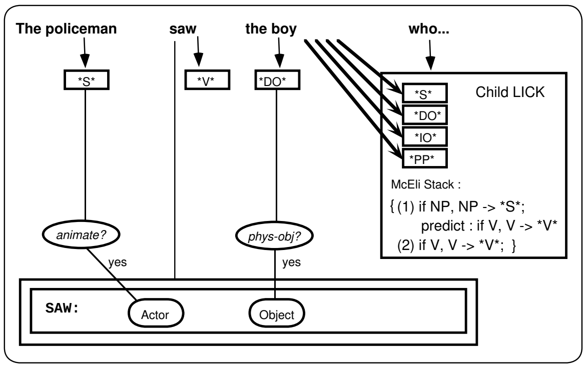

Figure 4: **PSM Status After "The policeman saw the boy who..."**

Figure 5 shows the state of the child LICK just after processing "accused". "Crowd" has overwritten *S* and "party" has overwritten *PP*.[8] In addition, "accused" triggers activation of a case frame and makes initial slot assignments based on the case frame definition: Actor = crowd and Patient = boy. The Accusation slot remains empty even though we have a prepositional phrase because the hard constraint that the preposition be "of" is violated. Note that although both *IO* and *DO* contain the antecedent "boy", *IO* does not interfere with the semantic representation because the ACCUSE case frame does not access that buffer.

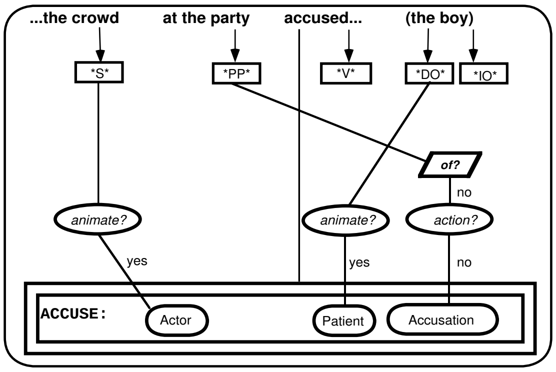

Figure 5: **"The policeman saw the boy who the crowd at the party accused..."**

Figure 6 shows the state of the child LICK at the end of the embedded clause: Actor = crowd, Patient = boy, Accusation = crime. At this point, CIRCUS freezes the ACCUSE case frame, assigns the instantiated representation to the *LB* buffer, exits the child LICK, and returns control to the main clause where *LB* is attached to the antecedent "boy".

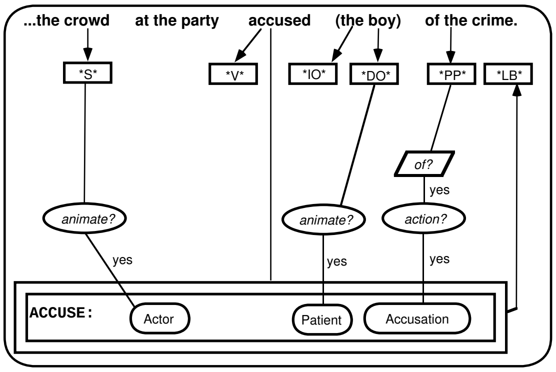

Figure 6: **"The policeman saw the boy who the crowd at the party accused of the crime."**

[8] Currently CIRCUS has only one *PP* buffer. The implication is that the parser only has access to the most recent prepositional phrase. Clearly, we should be using multiple buffers or a stack of *PP* buffers.

Psycholinguistic Studies of Wh-Constructions

Many recent experiments in psycholinguistics have addressed the human processing of wh-constructions. We will discuss a few experiments that focus on processing phenomena, but space limitations have forced us to omit many equally interesting studies, e.g., [Clifton *et al.*, 1984], [Frazier and Clifton, 1989], and a series of experiments from [Nicol, 1988].

Reactivation Effects

A Swinney, Ford, Frauenfelder, and Bresnan study [Swinney *et al.*, 1988] determined that people "reactivate" the meaning of a wh-phrase antecedent at the position of its gap in the embedded clause.

(1) The policeman saw the boy who the crowd at the party accused # of the crime.

At # in sentence (1), for example, subjects respond faster to a word semantically related to "boy" (e.g., "girl") than to a control word or to words associated with "policeman" and "crowd".[9] This result implies that people have integrated the meaning of the filler into the current semantic representation of the sentence at the point of the missing constituent. CIRCUS is consistent with this finding. Reactivation occurs in CIRCUS when the syntactic constituent currently expected according to the McEli stack is found to contain the antecedent. In (1), for example, syntactic knowledge stored with "accused" sets up the McEli stack to expect a direct object to follow and CIRCUS reactivates "boy" immediately following "accused" because the next constituent expected by McEli is the direct object, and *DO* already contains the antecedent "boy".

Although Swinney's study tested for priming of the antecedent only in the direct object position, CIRCUS predicts that priming of the antecedent should also occur in the subject position. In sentence (1), for example, the LICK associated with "who" assigns "boy" to the *S*, *DO*, *IO*, and *PP* buffers and predicts either a subject-verb or just a verb phrase for the embedded clause (see Figure 4). Because *S* is the next constituent expected by McEli and *S* has already been filled by the antecedent "boy", CIRCUS reactivates "boy" immediately after processing "who". To our knowledge, no psycholinguistic studies have tested for antecedent priming in the subject position. Therefore, further experimentation is required before the predictions made by CIRCUS can be confirmed or refuted.

[9] In the [Swinney *et al.*, 1988] study, the target word was briefly flashed at some point during aural presentation of the sentence. Subjects were asked to decide whether or not the visually presented word was a real word and press the appropriate button. Faster response to "girl" than "policeman" or "crowd" is attributed to priming by the semantically related word "boy".

Finally, the [Swinney *et al.*, 1988] study found reactivation only for the **correct** antecedent at # in (1). He found no reactivation of "crowd" or "policeman". CIRCUS also reactivates *only* the correct antecedent because the LICK formalism makes "boy" the only matrix clause constituent accessible to the embedded clause. No other noun phrases in the sentence (e.g., "policeman", "crowd", "party") are considered as antecedents of "who".

Thus, CIRCUS seems to employ a psychologically valid mechanism for reactivation of antecedents in wh-phrases: it reactivates the antecedent at the point of the gap and it reactivates only the correct antecedent.

Filled Gap Effects

Studies in [Crain and Fodor, 1985] and [Stowe, 1986] have produced evidence for a phenomenon called the *Filled Gap Effect.* This occurs when the processor has postulated a gap for an antecedent, but then discovers it has made a mistake — it finds that the hypothesized gap position is not actually empty in the input string. Their experiments showed an increase in reading time at the point of the filled gap. Consider, for example, these sentences from [Stowe, 1986]:

(2) My brother wanted to know who_i Ruth will bring us home to (i) at Christmas.

(3) My brother wanted to know who_i Ruth will bring (i) home to Mom at Christmas.

The position after "bring" in (2) constitutes a filled gap. It seems that the processor has noticed the antecedent "who" and anticipates its filling the object position. Instead, it finds "us" as the direct object, is momentarily confused, and is forced to reconsider its hypothesis that "who" is the direct object. The extra processing load at "us" causes the increase in reading time at this point in the sentence. Sentence (3), on the other hand, does not exhibit the Filled Gap Effect because there is no direct object in the sentence that might conflict with the processor's initial hypothesis that "who" should be the direct object.

CIRCUS is consistent with these findings. The LICK triggered to process the embedded clauses in (2) and (3) initially assigns "who" to the *S*, *DO*, *IO*, and *PP* buffers. By the word "bring", however, the subject buffer has been overwritten with "Ruth". A processing slowdown occurs in (2) when CIRCUS finds "us" as the real direct object and is forced to change *DO* from "who" to "us". As a side effect, this syntactic modification changes the Patient role of "bring" from "who" to "us". The slowdown caused by this reanalysis is consistent with the results of [Crain and Fodor, 1985] and [Stowe, 1986]. There is no such reanalysis required by CIRCUS at the same position in sentence (3).

Most analyses of the [Crain and Fodor, 1985] and [Stowe, 1986] studies attribute the filled gap effect to the *syntactic* reanalysis required at the direct object position. In CIRCUS, however, it is not clear whether the slowdown is due to syntactic or semantic reanalysis at the object position, since both occur: McEli overwrites the contents of a syntactic buffer and PSM recomputes the slot fillers for any active case frames. To investigate the possibility that semantics plays a role in the slowdown, we would like to hypothesize the following constraint on the Filled Gap Effect:

The Filled Gap Effect should only occur when the overwritten constituent had been a plausible role filler in an active case frame.

No Filled Gap Effect is expected when the antecedent is an implausible filler in a case frame because PSM prefers syntactic constituents that satisfy all of the slot's semantic constraints, i.e., it prefers plausible fillers over implausible ones. Hence, the revised Filled Gap Effect predicts a slowdown at "me" in (4), but not in (5) below:

(4) The district attorney found out *which witness*$_i$ the reporter asked me anxiously about (i).

(5) The district attorney found out *which building*$_i$ the reporter asked me anxiously about (i).

As a semantically-driven parser, CIRCUS is consistent with the revised Filled Gap Effect. At the word "asked" in (4), the following role assignments exist in the embedded clause LICK: Actor (*S*) = reporter, Patient (*IO*) = witness. Because "witness" is a plausible filler for the Patient role of "ask", the processor should be reluctant to change this role assignment. When the real indirect object is recognized, however, CIRCUS is forced to reanalyze the current clause: "me" overwrites *IO* and, as a side effect, bumps "which witness" from the Patient slot. Implausible role fillers require no such reanalysis. In the embedded clause in (5), for example, the following role assignments exist at "asked": Actor (*S*) = reporter, Patient (*IO*) = building. "Building", however, is marked as an implausible filler of the Patient role because it does not satisfy the soft constraints associated with the Patient slot. Since the processor has not successfully filled the Patient role with a semantically valid candidate, no Filled Gap Effect is expected at "me".

While the predictions of the revised Filled Gap Effect have not been confirmed, a study described in [Tanenhaus *et al.*, 1989a] found that the Filled Gap Effect disappears for implausible fillers for at least one class of verbs. This study used a continuous make-sense-judgment task[10] to evaluate the Filled Gap Effect in sentences containing verbs that expect both a direct object and infinitive complement, e.g., remind:

(6) *Which movie*$_i$ did Mark remind them to watch (i) this evening?

(7) *Which child*$_i$ did Mark remind them to watch (i) this evening?

The Filled Gap Effect for the direct object does not appear in sentences like (6) where the antecedent ("movie") is an implausible object of "remind". It does occur in sentences like (7) where the antecedent ("child") satisfies the semantic constraints associated with the object slot of "remind". The embedded clauses in the [Tanenhaus *et al.*, 1989a] study, however, only contained verbs that require a direct object and an infinitive complement (e.g., remind, tell). Because we claim that the revised Filled Gap Effect will hold for all classes of verbs, further experimentation is required to confirm our more general hypothesis.

The [Stowe, 1986] study, however, indirectly contributes evidence supporting our claim that the Filled Gap Effect is dependent on the semantics of the sentence — she found no Filled Gap Effect in the subject position of embedded clauses. This result supports our claim because the Filled Gap Effect can only occur when there is an active semantic case frame. In most cases, CIRCUS does not trigger a case frame until it encounters the verb. In addition, the study described in the next section supplies evidence related to our hypothesis that the Filled Gap Effect is at least partially a semantically-driven processing phenomenon.

Thematic Role Effects

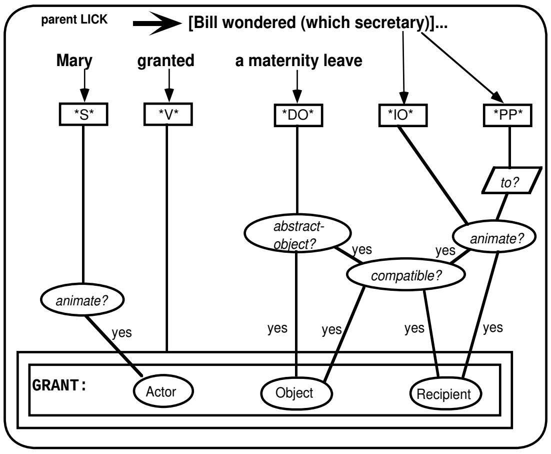

Figure 7: **"Bill wondered which secretary..."**

A study described in [Tanenhaus *et al.*, 1989b] contradicts the syntax-first theory of parsing often espoused by syntactically-oriented approaches to wh-phrases, while supporting the computational architecture of CIRCUS. The study used a continuous make-sense-judgment task on sentences of the following types:

(8) Bill wondered *which secretary*$_i$ Mary granted a maternity leave to (i).

(9) Bill wondered *which bachelor*$_i$ Mary granted a maternity leave to (i).

[10] In this type of experiment, subjects are asked to continuously push one of two buttons indicating whether or not the sentence currently makes sense.

Subjects indicated ungrammaticality at "a maternity leave" in (9) and also took longer to make a response at this point. Neither of these effects occurred at "a maternity leave" in (8). [Tanenhaus *et al.*, 1989b] interpreted these results to mean that the processor is assigning an antecedent to a gap based on case role *before the gap ever appears in the sentence.* Parsers that rely on a global syntactic grammar for postulating gaps are inconsistent with this finding.

CIRCUS, on the other hand, is completely consistent with the Tanenhaus results because PSM assigns case roles to syntactic constituents as soon as the constituents become available. Figure 7 shows the state of the parser for sentence (8) after processing "maternity leave": Actor = Mary, Recipient = which secretary, Object = maternity leave. All hard and soft slot constraints are satisfied. Figure 8, on the other

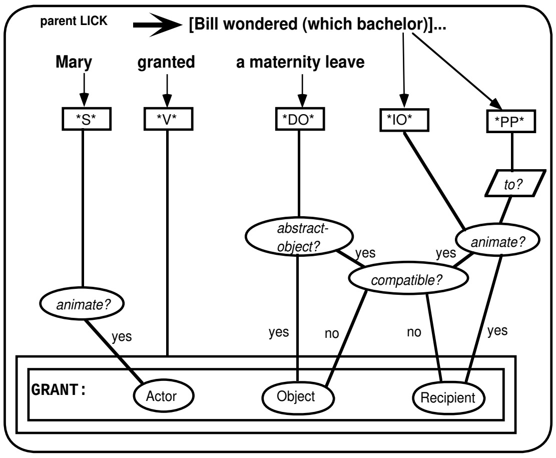

Figure 8: **"Bill wondered which bachelor..."**

hand, indicates inconsistencies in the meaning representation at the same point in (9). Because maternity leaves and bachelors are not compatible, the soft constraints for the Object and Recipient slots are violated and CIRCUS tags both slot fillers as "semantic failures". Thus, CIRCUS mimics the Tanenhaus results in that sentence (9) does not make sense to CIRCUS beginning at "maternity leave".

The Tanenhaus study presented an additional result: they found no increase in reading time after "to" in either sentence. Parsers that manipulate or produce a syntactic parse tree representation of a sentence find this result particularly troublesome. These parsers predict an increase in reading time after "to" because the syntactic parse tree has to be rearranged at this point when the false gap is discovered.[11] If one assumes that syntactic changes that have no semantic effects require no additional time, CIRCUS is consistent with this study.[12] At "to" in sentences (8) and (9), McEli removes the antecedent from the false gap buffer *IO*, leaving it only in *PP*. However, this syntactic change is actually unnecessary in that it provokes no semantic reanalysis. As depicted in Figures 7 and 8, either *IO* or *PP* provide the Recipient role for GRANT. Emptying *IO* at "to" forces *PP* to fill the Recipient slot rather than *IO*, but because **both** buffers held the same antecedent, there is no change to the meaning representation.

Based on our comparisons of CIRCUS with the results of psycholinguistic studies of reactivation, filled gap and thematic role effects, we conclude that CIRCUS employs cognitively plausible processing mechanisms in its interpretation of wh-constructions.

Conclusions

Because space limitations prevent us from providing a more exhaustive presentation of the various embedded clause constructs handled by the LICK mechanism, we have focused on the use of LICKs to understand wh-phrases. However, we currently use LICKs to handle all types of embedded clauses in the 1300 newswire stories of the MUC-3 development corpus. In particular, the LICK mechanism can infer the missing subject of infinitival complement clauses: e.g., 1) "The terrorist tried to kidnap the businessman", and 2) "The terrorist asked the drug mafias to kidnap the businessman". In sentences like 1, CIRCUS spawns a child LICK that initializes the subject of the embedded clause with the subject of the main clause (i.e., "terrorist" becomes the actor of "to kidnap"). In sentences like 2, however, CIRCUS spawns a child LICK that initializes the embedded clause subject with the direct object of the main clause (i.e., "drug traffickers" becomes the actor of "to kidnap"). In addition, both the subject-controlled LICK and the object-controlled LICK set up McEli stack expectations for the infinitive form of a verb to begin the embedded clause. Still other LICKs are responsible for handling sentential complements (e.g., "The peasants thought the president had been assassinated") and interactions between wh-phrases and complement clauses (e.g., "The president told the people what to do in case of bombing").

However, we also understand that some embedded clause problems cannot be resolved by the simple inter-LICK control rules and communication rules described here. For example, a reduced relative clause presents an ambiguity that must be resolved by either a parent LICK (in the case of an active past tense verb form) or a child LICK (in the case of a passive past participle verb form). The control kernel formalism encourages us to view this disambiguation problem in terms of

[11] The verb phrase of the embedded clause changes from V-NP-NP to V-NP-PP.

[12] Without this assumption, further experimentation would be required to determine whether minimal syntactic "reanalysis" (i.e., emptying a syntactic buffer) takes a significant amount of time when compared to the time it takes for semantic reanalysis.

competition for control, but does not suggest how that competition should be resolved.

Our approach to syntactic/semantic interactions recasts the problems of embedded constructions as issues concerning communication across scoping environments. We propose lexically-indexed control kernels as a formalism in which these scoping problems are handled naturally without additional machinery specific to individual syntactic constructs. We know of no other semantically-oriented parser that handles embedded clause constructs in such a systematic manner, and we know of no syntactically-oriented parser that handles these constructs without manipulating syntactic parse trees or using a global syntactic grammar.

In addition, we have shown that CIRCUS and the LICK formalism offer a cognitively plausible mechanism for understanding a subset of embedded clause constructions. We believe that this adherence to a plausible computational model of human processing allows CIRCUS to achieve robust sentence processing capabilities not found in other parsers of its class.

Acknowledgments

This research was supported by the Advanced Research Projects Agency of the Department of Defense, monitored by the Office of Naval Research under contract #N00014-87-K-0238, the Air Force Office of Scientific Research under Contract No. F49620-88-C-0058, the Office of Naval Research under a University Research Initiative Grant, Contract #N00014-86-K-0764, and NSF Presidential Young Investigators Award NSFIST-8351863. The United States Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright notation hereon.

References

Birnbaum, L. and Selfridge, M. Conceptual Analysis of Natural Language. In Schank, R. and Riesbeck, C., editors, *Inside Computer Understanding*. Lawrence Erlbaum Associates, Hillsdale, NJ, 1981.

Bresnan, J. *The Mental Representation of Grammatical Relations*. MIT Press, Cambridge, 1982.

Chomsky, C. *The Acquisition of Syntax in Children from 5 to 10*. MIT Press, Cambridge, 1969.

Clifton, C., Frazier, L., and Connine, C. Lexical expectations in sentence comprehension. *Journal of Verbal Learning and Verbal Behavior*, 23, 1984.

Crain, S. and Fodor, J. How can grammars help parsers? In Dowty, D., Kartunen, L., and Zwicky, A., editors, *Natural Language Parsing: Psychological, computational, and theoretical perspectives*. Cambridge University Press, 1985.

Cullingford, R. *Natural Language Processing*. Rowman and Littlefield, Totowa, NJ, 1986.

Fodor, J. Empty Categories in Sentence Processing. *Language and Cognitive Processes*, 4, 1989.

Frazier, L. and Clifton, C. Successive cyclicity in the grammar and the parser. *Language and Cognitive Processes*, 4, 1989.

Kay, M. Algorithm schemata and data structures in syntactic processing. In *Proceedings of the Symposium on Text Processing*. Nobel Academy, 1980.

Lehnert, W., Cardie, C., and Riloff, E. Analyzing Research Papers Using Citation Sentences. In *Proceedings of the Twelfth Annual Conference of the Cognitive Science Society*, pages 511–518, Cambridge, MA, July 1990. Lawrence Erlbaum Associates.

Lehnert, W. Symbolic/Subsymbolic Sentence Analysis: Exploiting the Best of Two Worlds. In Barnden, J. and Pollack, J., editors, *Advances in Connectionist and Neural Computation Theory*, pages 135–164. Ablex Publishers, Norwood, NJ, 1990.

Lytinen, S. *The Organization of Knowledge in a Multi-lingual, Integrated Parser*. PhD thesis, Yale University, 1984. Also available as Tech Report YALEU/CSD/RR #340.

Nicol, J. *Coreference Processing During Sentence Comprehension*. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, 1988.

Riesbeck, C. and Martin, C. Direct Memory Access Parsing. In Riesbeck, C. and Kolodner, J., editors, *Experience, Memory and Reasoning*. Lawrence Erlbaum, Hillsdale, NJ, 1985.

Riesbeck, C. Conceptual Analysis. In Schank, R., editor, *Conceptual Information Processing*. North Holland, Amsterdam, 1975.

Schank, R. and Riesbeck, C. *Inside Computer Understanding: Five Programs Plus Miniatures*. Lawrence Erlbaum, Hillsdale, NJ, 1981.

Schank, R. *Conceptual Information Processing*. North Holland, Amsterdam, 1975.

Stowe, L. A. WH-constructions: evidence for on-line gap location. *Language and Cognitive Processes*, 3, 1986.

Swinney, D., Ford, M., Bresnan, J., and Frauenfelder, U. Coreference assignment during sentence processing. In Macken, M., editor, *Language structure and processing*. CSLI, Stanford, Palo Alto, CA, 1988.

Tanenhaus, M., Boland, J., Garnsey, S., and Carlson, G. Lexical Structure in Parsing Long-Distance Dependencies. *Journal of Psycholinguistic Research*, 18(1), 1989.

Tanenhaus, M., Carlson, G., and Trueswell, J. C. The Role of Thematic Structures in Interpretation and Parsing. *Language and Cognitive Processes*, 4, 1989.

Wilks, Y., Huang, X., and Fass, D. Syntax, Preference, and Right Attachment. In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, Los Angeles, CA, August 1985.

Woods, W. Transition network grammars for natural language analysis. *CACM*, 13(10), 1970.