Using Real-World Reference to Improve Spoken Language Understanding

Scott C. Stoness, James Allen, Greg Aist and **Mary Swift**
Department of Computer Science, University of Rochester
{stoness, james, gaist, swift}@cs.rochester.edu

Abstract

Humans understand spoken language in a continuous manner, incorporating complex semantic and contextual constraints at all levels of language processing on a word-by-word basis, but the standard paradigm for computational processing of language remains sentence-at-a-time, and does not demonstrate the tight integration of interpretations at various levels of processing that humans do. We introduce the fruit carts task domain, which has been specifically designed to elicit language that requires this sort of *continuous understanding*. A system architecture that incrementally incorporates feedback from a real-world reference resolution module into the parser is presented as a major step towards a continuous understanding system. A preliminary proof in principle shows that real-world knowledge can help resolve certain parsing ambiguities, thus improving accuracy, and that the efficiency of the parser, as measured by the number of constituents built, improves by upwards of 30% on certain example sentences with multiple attachment ambiguities. A 26% efficiency improvement was achieved for a dialogue transcript taken from those collected for the fruit carts task domain. We also argue that real-world reference information can help resolve ambiguities in speech recognition.

Continuous Understanding of Spoken Language

There are a number of speech-to-intention dialogue systems which undertake the task of understanding and/or interperting spoken language, such as Verbmobil (Kasper *et al.* 1996; Noth *et al.* 2000; Pinkal, Rupp, & Worm 2000), Gemini (Dowding *et al.* 1993; 1994; Moore *et al.* 1995) and TRIPS (Allen *et al.* 1996; Ferguson, Allen, & Miller 1996; Ferguson & Allen 1998). The traditional control flow of such systems is a layered sequential processing where the output of speech recognition for the entire sentence is passed to a parser, which in turn hands its analysis to higher-level modules; only after the entire sentence has been parsed do modules such as reference resolution and intention recognition have a chance to analyze the sentence, and the scope of their activity is limited to that set of analyses which the parser has deemed fit to pass forward.

This layered sequential model is certainly appropriate for processing written text where the entire sentence is available at once, but the task of understanding speech is inherently incremental because the input is a stream rather than a string. Humans take advantage of this fact, incorporating all levels of language analysis simultaneously, from speech recognition to semantics, reference, and intention recognition, at the granularity of the word rather than that of the sentence, as has been shown in the eye-tracking literature (Cooper 1974; Tanenhaus & Spivey 1996; Allopenna, Magnuson, & Tanenhaus 1998; Sedivy *et al.* 1999; Phillips 1996). In the context of a natural language understanding system we refer to this immediate incorporation of higher-level information as *continuous understanding*.

While it need not be the case that computers best process language in the same fashion as humans, we believe that continuous understanding models have several significant computational advantages, including:

- improving the accuracy and efficiency of real-time spoken language understanding;
- better supporting the understanding of spontaneous human speech, which does not proceed in well-formed complete utterances as in written text; and
- enabling better human-computer spoken language interfaces (for example, confirming and/or clarifying understanding as the user is speaking).

The major computational advantage of continuous understanding is that high-level expectations and feedback can be used to influence the search of lower level processes, thus leading to a focussed search through hypotheses that are plausible at all levels of processing. Note that it is the incorporation of feedback that makes a model continuous, not the concurrent processing. Incorporating feedback from higher levels allows a module to intelligently prune its own search space to remove those hypotheses which can be rejected by other modules with access to different information; this pruning both speeds the search and improves the accuracy of the system, because globally unlikely hypotheses become dispreferred. Thus, not only is the accuracy of the module improved, but so is the accuracy of the system as a whole since it is less prone to "truncation" errors, where

the n-best output at one level does not include the globally best hypothesis.

A continuous understanding system can more easily understand spoken language, because spontaneous speech is conveyed incrementally in segments that often do not resemble well-formed sentences of written text. This makes the understanding process considerably more difficult and requires a much greater reliance on contextual interpretation: ensuring that this contextual interpretation is available simultaneously at all levels is the very essence of continuous understanding.

Finally, human conversation is a highly interactive activity, involving a fine-grained interaction within turns. This type of interaction enables more efficient communication, which ideally human-computer interfaces could mimic. While a computer might not interact in exactly the same way as people do, grounding mechanisms are needed for effective human-computer interaction (Horvitz & Paek 1999).

Parsing for Continuous Understanding

In order to have a true end-to-end continuous understanding system, it is necessary that all of the modules operate in an incremental fashion, able to send forward hypotheses on a word-by-word basis, and to receive feedback about those hypotheses from a higher-level module.

Obviously, in any spoken language understanding system, the speech recognizer is the starting point of processing, and a significant amount of research has been done to incrementally incorporate higher-level information into speech modelling, often using an incremental parser to provide additional grammatical and/or semantic context beyond that which is available from the standard n-gram models (Brill *et al.* 1998; Jelinek & Chelba 1999; Wang & Harper 2004). This work is important not only for its ability to improve performance on the speech recognition task; it also models the necessary interactions between speech recognition and parsing in a continuous understanding system. Our research attempts to further the quest for continuous understanding by moving one step up the hierarchy, building an incremental parser which receives advice rather than providing it.[1]

The robust semantic processing literature provides us with a number of examples of systems which contain incremental parsers, but not continuous understanding parsers (Pinkal, Rupp, & Worm 2000; Rose 2000; Worm 1998; Zechner 1998). These systems pass forward partial parses as they are constructed, enabling the robust semantic analysis component to begin its work without waiting for the parser to finish. However, none of the systems above enable the semantic analysis component to affect the parsing, so they are not truly continuous understanding parsers.

Schuler (2002) describes a semantic parser that uses the denotations of constituents in the environment to inform parsing decisions, and is an excellent first step towards a continuous understanding parser, but is limited both in that the feedback consists of a single bit of information (essentially the presence or absence of an object in the world), and is not a component of a speech-to-intention system.

[1]Of course, an eventual system might incorporate both aspects of an incremental parser, providing advice to the speech recognition component while itself receiving adivce about its hypotheses from more information-rich modules, but for the moment we are focussing on the parser as advisee to the exclusion of parser as advisor.

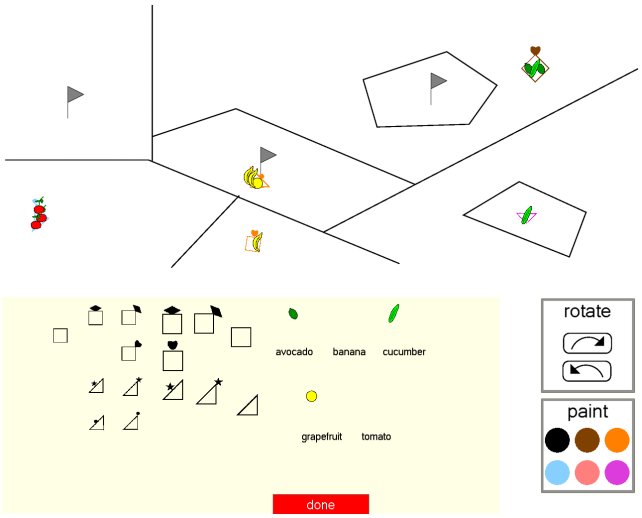

Figure 1: A Screen Capture of a Fruit Cart Session

We have built an initial attempt at a continuous understanding parser (Stoness, Tetreault, & Allen 2004) in the context of an existing task-oriented dialogue system, TRIPS. The system's bottom-up parser was modified to communicate incrementally with a reference resolution module, and the feedback from the higher-level system was incorporated into the chart so as to inform future parsing decisions.

The resulting system showed that the general architecture for incremental interaction outlined in the paper could be productively instantiated, and resulted in reasonably significant gains in parsing accuracy (5.8%, 8.2% and 9.3% error reduction), and modest improvements in the number of constituents built by the parser (a reduction of 4.6%, 4.0% and 3.6% in workload).

However, it also became clear that the domain was not particularly suited to initial experiments in continuous understanding, because processing dialogues is heavily dependent on accurately resolving references into the discourse context, a notoriously difficult proposition (Byron 2000). This suggests that the relatively modest accuracy and performance improvements reported could be improved upon in a more suitable domain, especially one which relied on real-world reference resolution to a knowledge base rather than discourse context.

The Fruit Carts Domain

TRIPS is a modular task-oriented dialogue system which isolates the information dependent on the task to relatively few modules; thus, when moving to a new domain, the bulk of the system remains unchanged, which in part accounts for the wide range of domains that the TRIPS system has been used in, including interactive planning and advice giving

(e.g., a medication advisor), robot control, and task learning (e.g., learning to perform tasks on the web). Even though the tasks and domains differ dramatically, these applications use the same set of core understanding components.

The fruit carts domain is designed specifically to research continuous understanding, encouraging both continuous-style language on the part of the user and multi-modal continuous response from the system.

The system displays a map consisting of several regions, and beneath it a variety of fruit and a number of shapes in which the fruit can be placed (the carts). These square and triangular carts come in various sizes, and some are decorated with smaller shapes (hearts, stars, diamonds) on the edges and corners.

Initially, the map is empty, and the user is given a card depicting the ideal final state of the map that should be achieved.[2] The user then speaks to the system, asking it to move and rotate carts or fruit, paint carts various colours, put fruit in carts or carts in regions on the map, or simply select an object for a later command.

During dialogue collection[3] the user spoke to a computer displaying the interface shown in Figure 1, and a (human) confederate silently manipulated objects on the screen in a natural (i.e. continuous) way. Thus, objects would be selected the moment they were understood, and actions would begin when it was natural to do so.

For example, "move a square to central park" would likely result in the confederate selecting a square and beginning to drag it toward the map before the destination "central park" has even been heard.

This incremental system response resulted in a significant amount of continuous language. Because the system responded incrementally, much as another human (in this case, the confederate) would, language requesting the rotation of an object ranged from full commands such as "and you're going to rotate the square like 30 degrees to the right" to the much more continuous utterance "rotate it *pause* a little bit more *pause* right", which is only interpretable in light of the fact that the confederate began actions as soon as the user's intentions were understood.

The decorated objects also evoke complicated descriptions that generate multiple attachment ambiguities for a parser trying to analyse the language. For example, "put the square in central park right underneath the flag and color it magenta" and "take a uh small triangle with a star on the corner and put it um so that the hypoteneuse should be horizontal and it should be in uh pine tree mountain" offer numerous attachment options, and even genuine ambiguities.

The ultimate goal of the fruit carts domain is to build a continuous understanding system atop the pre-existing TRIPS framework which responds to user commands in much the same way that our confederates did: immediately and incrementally, enabled by an ability to analyse language at all levels simultaneously. In the meantime, the 52 dialogue transcripts, representing 4 trials each from 13 speakers, provide a rich source of language which requires continuous understanding to be interpreted, and the construction of the system provides a research platform for investigating numerous aspects (and complications) of the move towards a continuous understanding system.

[2]Note that the names of the regions appear only on the user's card, rather than on the map itself; names on the map elicit speech such as "put the banana near the 'a' in 'park"'.

[3]Something of a misnomer, as only the user speaks.

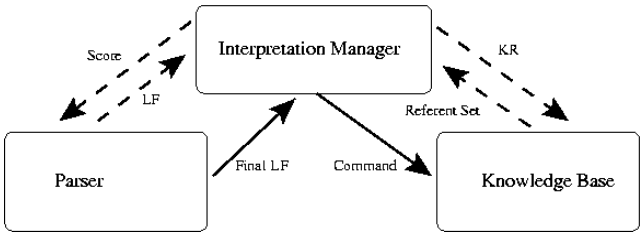

Figure 2: Key Components for Continuous Real-World Reference

The Fruit Carts System

As mentioned above, one of the initial goals of the system is to incorporate higher-level feedback into the parser, in this case, specifically feedback from real-world reference, albeit the "real world" of the virtual fruit cart domain.

Providing the parser with information from real-world reference is anticipated to improve both the efficiency and accuracy of the parser, because this domain-specific grounded information will aid the parser in resolving attachment ambiguities, not only by guiding the parser towards the correct interpretation, but also by steering the parser away from implausible ones.

Consider the sentence "put the square near the flag in the park." A traditional parsing viewpoint would be that this sentence is inherently ambiguous between (a) moving some well-specified square to a location close to "the flag in the park" and (b) putting the square which is near the flag into the park. However, in a task-oriented domain in which a user is giving instructions, it would be decidedly odd for a user to deliberately utter an ambiguous phrase. While accidents may occur, it is a productive strategy to use continuous understanding to take into account real-world reference while assuming that the user's speech is somehow "appropriate" to the world.

For example, if a square is currently selected on the screen, it is very likely that the correct interpretation of the sentence above involves moving the selected square. Moreover, one can assume in general that any definite noun phrase (NP) that could refer to the selected object in the world probably does in the absence of information to the contrary.

Similarly, most referring NPs should probably refer to some real-world entity; if in the real world no square is close to a flag, then "the square near the flag" is very unlikely to be a component of the final parse, barring surprising circumstances in the user's construction of the utterance: such gems as "the square near the flag does not exist", while certainly linguistically possible, are, pragmatically speaking, quite unlikely in a task-oriented domain.

TRIPS is a flexible framework that requires little modifi-

Keyword	Article	Reference Context
GREAT	the	SELECTED
GOOD	the	SET, length = 1
BAD	the	SET, length > 1
TERRIBLE	the	SET, length = 0
BAD	a	SET, length = 1
GOOD	a	SET, length > 1
TERRIBLE	a	SET, length = 0
NEUTRAL		otherwise

Table 1: Classification of NPs based on Reference Context

Keyword	Score
GREAT	1.0
GOOD	0.9
NEUTRAL	0.5
BAD	0.3
TERRIBLE	0.0

Table 2: Classification of NPs based on Reference Context

cation to move to new domains, but the core infrastructure did require some improvements to enable continuous understanding. Figure 2 shows the components most involved in continuous understanding with real-world reference interaction. The dashed lines represent incremental communication between modules, while the solid lines represent the updates that occur after a final sentence interpretation has been agreed upon.

The continuous understanding portion of the communication proceeds roughly as follows. Whenever the Parser builds an NP constituent, it forward the logical form of the constituent to the Interpretation Manager (IM). The IM performs two essential functions: first, it filters out non-referring NPs, immediately informing the parser of this fact; and second, it uses domain-specific transformation rules to translate the domain-independent logical form understood by the parser into a domain-dependent knowledge representation (KR) appropriate for the Knowledge Base (KB). The KB then sends back the set of objects which match the specification provided by the IM, with an additional annotation if one of those objects is currently selected on the screen. The IM then interprets this list in the context of the NP constituent to arrive at a real-world reference score. This score is passed back to the parser to be incorporated into its chart.

The set of referents returned from the knowledge base is cross-referenced with the article of the original NP, and classified into one of five categories of "goodness", as shown in Table 1. Each of these keywords is then assigned a value between 0 and 1, as shown in Table 2. Note that the values shown were chosen purely by intuition, and were the only ones tried in the experiments; ideally these would be learned from held-out data.

This reference resolution score is then merged with the score on the constituent. Experiments with a discourse context advisor in a different domain suggested that of a dozen candidate feedback regimes, the "Linear Interpolation with Boost" was most effective in conjunction with the standard TRIPS parser. The new score for the constituent, C_{score}, is given by

$$C_{score} = (1 - \frac{\lambda}{2}) \cdot P_{score} + \lambda \cdot R_{score},$$

where P_{score} is the score assigned by the parser, and R_{score} is the score based on the advice from real-world reference. λ takes a value of 0.1, which was established as optimal through the aforementioned experiments in a different domain.

In this scheme NP candidates which score better than NEUTRAL have their score go up, while those which score lower than NEUTRAL have their score go down. Thus, favoured constituents will be preferentially considered for further expansion, while dispreferred constituents will languish in the agenda until called upon as a last resort. Note that while the incorporation of a TERRIBLE score certainly penalizes a constituent, the small value of λ ensures that it is never completely removed from the search; if the more favoured candidates fail to pan out in the parse, the penalized constituents can still be productive components of future search strategies.

Currently, the IM waits until all processing is complete to pass the final interpretation of the utterance on to the KB in the form of a command. Ideally, the IM will issue partial, underspecified commands as soon as the user's intentions are (believed to be) understood.

A Proof in Principle

In this section we provide a brief example of the continuous understanding aspect of the system at work, using the benefits of processing an ambiguous sentence as a proof in principle of how both parsing accuracy and efficiency can be simultaneously improved by the incremental incorporation of feedback from a real-world reference component.

While eventually the fruit carts system will respond to speech input, for simplicity's sake both for this proof in principle and for our upcoming preliminary experiments with the system we use manually transcribed speech from the collected dialogues. Once experiments have shown productive ways of continuously understanding incrementally arriving text we can move forward to tackling true speech input.

We began by parsing the sentence "put the square near the flag in the park" with the standard version of the parser, operating without continuous understanding. As mentioned above, for a non-continuous parser this sentence is inherently ambiguous, so the choice of a most likely parse is somewhat arbitrary; in the event, the parser selected "the square" as the direct object of the verb, and during the course of the parse built 197 total constituents.[4]

Then we created a simple knowledge base, *KB-selected*, which features a selected square, and a flag in a park, but no square near a flag. This set of knowledge clearly favours the

[4]Measurements of parsing efficiency are always tricky, but since both the continuous and non-continuous versions of the parser use identical grammars, the number of constituents built should serve as a reasonable measure for our purposes.

KB	Mode	Direct Object	Constits	selected	squares	near(sq,flag)	in(flag,park)
none	standard	"the square"	197	n/a	n/a	n/a	n/a
KB-selected	standard	"the square"	121	square	2	N	Y
KB-near	standard	"the square near the flag"	131	none	2	Y	N
KB-park	standard	*"the square"	165	none	2	Y	Y
KB-selected-u	standard	"the square"	144	square	1	N	Y
KB-near-u	standard	"the square near the flag"	108	none	1	Y	N
none	permissive	"the square ... the park"	204				
KB-selected	permissive	"the square"	127				
KB-near	permissive	"the square near the flag"	147				
KB-park	permissive	"the square ... the park"	124				

Table 3: Parsing Accuracy and Efficiency Results for Various Knowledge Bases

interpretation selected by the non-continuous parser above. The continuous parser output the desired interpretation as its most likely parse, but only built 121 constituents; an efficiency improvement of almost 40%.

Operating in continuous mode doesn't just improve the efficiency of the parser, but its accuracy as well. A different initial knowledge base, *KB-near*, features a square near a flag, but no flag in a park, and has no square selected. This KB favours an interpretation in which "the square near the flag" is the direct object. The non-continuous parser cannot make this distinction, even in principle, and so to capture the multiple possible interpretations, each preferable in a different context, it is necessary for the parser to feed forward a number of complete parses at the completion of its processing. A continuous understanding parser, however, has at its disposal, incrementally and immediately, the same knowledge that would be used to disambiguate the complete parses in a non-continuous system.

Purely by changing the knowledge base to *KB-near* and allowing the reference feedback to be incorporated into the parse, the continuous system finds the correct parse as its most likely candidate, while building only 131 constituents.

KB-park is a third knowledge base which has neither a selected square nor a square near a flag, but does feature a square that is near a flag which is in a park. With this KB, the favoured NP is "the square near the flag in the park". However, restrictions on the verb "put" require the parse to have both a direct and indirect object, and the parser thus returns to the same interpretation it favoured in the absence of any information from the KB. Interestingly, this entire process requires the construction of only 165 constituents; that is, even when the KB leads the parse somewhat astray, the incorporation of the domain knowledge still improves on the base parser's efficiency of 197 constituents.

The parser also can operate in a permissive mode, allowing required arguments of verbs to be omitted, a phenomenon which occurs with some regularity in spoken language, and quite frequently in the fruit carts dialogues. In this mode the standard parser actually opted for the interpretation "put [the square near the flag in the park]", building 204 constituents along the way. As Table 3 shows, feedback from *KB-selected* and *KB-near* achieved the expected parse with admirable efficiency, as did *KB-park*, since in this instance its desired interpretation is permitted by the relaxed grammar.

Finally, we tested the sentence "put the square near the flag in the park in the forest". The non-continuous parser found "in the forest" as the indirect object, building 396 constituents in the process. Using *KB-park* as a knowledge source, however, the continuous parser arrived at the same interpretation in only 196 constituents.

Obviously, the example sentence chosen is highly ambiguous and not all knowledge bases will match as perfectly with the intended interpretations as those that we have chosen. But, as we argued earlier, we can expect, in principle, that the real-world information will tend to help disambiguate in-context utterances, based on the principle that users will rarely be either deliberately or accidentally ambiguous. Moreover, the example sentence, while complex, is fairly short, and as the discussion in the previous paragraph demonstrates, the longer the sentence, the larger the potential time savings. Efficiency in parsing is paramount in a real-time spoken dialogue system, especially if the system expects to react prior to the completion of the user's utterance. For example, when run on a single processor, text-to-intention interpretation of the the example sentence with continuous understanding takes approximately 3 seconds, so inefficiencies could clearly move this out of real-time range for longer sentences.

A Dialogue Experiment

As well as the proof-in-principle sentences interpreted in context, we have run the system on the transcript of a complete dialogue from the corpus that we collected for this domain. We are in the process of creating hand-corrected gold standard parses for all collected dialogues, which will allow us to evaluate the accuracy of the parser both in non-continuous and continuous understanding modes; we report preliminary results for the first of these dialogues to be marked up. In conjunction with gold standard interpretations of the utterances as commands to the system (yet to be completed), it will also be possible to evaluate the accuracy of the interpretations that the system assigns to each utterance.

The initial set of six dialogue transcripts to undergo mark-up were chosen because they contained fewer exam-

UTT	Utterance
1	okay so
2	we're going to put a large triangle with nothing into morningside
3	we're going to make it blue
4	and rotate it to the left forty five degrees
5	take one tomato and put it in the center of that triangle
6	take two avocados and put it in the bottom of that triangle
7	and move that entire set a little bit to the left and down
8	mmkay
9	now take a small square with a heart on the corner
10	put it onto the flag area in central park
11	rotate it a little more than forty five degrees to the left
12	now make it brown
13	and put a tomato in the center of it
14	yeah that's good
15	and we'll take a square with a diamond on the corner
16	small
17	put it in oceanview terrace
18	rotate it to the right forty five degrees
19	make it orange
20	take two grapefruit and put them inside that square
21	now take a triangle with the star in the center
22	small
23	put it in oceanview just to the left of oceanview terrace
24	and rotate it left ninety degrees
25	okay
26	and put two cucumbers in that triangle
27	and make the color of the triangle purple

Table 4: The dialogue

ples of speech that required continuous interpretation, and thus would be maximally accessible to the existing non-incremental parser. Table 4 shows the test dialogue chosen.

The experiment proceeded in much the same manner as described in the proof-in-principle section, with candidate NPs being sent forward through the Interpretation Manager to the Knowledge Base, which provided feedback on whether the NP was a reasonable candidate, taking into account both domain-specific knowledge and the current state of the world. Because the user's utterances had to be interpreted relative to the state of the world that the user had been aware of during dialogue collection, a series of knowledge base updates were performed between sentences to ensure that the KB was an accurate reflection of what the user had seen.

The results of the experiment are shown in Table 5, which lists both the overall parser efficiency for the complete dialogue and the number of constituents built for specific utterances of interest. Overall, the continuous understanding parser only had to build 75% as many constituents as the standard parser in order to find its first complete parse of each utterance, with utterances 5 and 6 being shining examples of the improvements in efficiency that are possible.

Both sentences have a similar structure, in that they are a conjunction of the selection of a fruit (or fruits) with an instruction for the placement of that fruit. Both "take one tomato" and "put it in the center of that triangle" (from utterance 5) are aided by the real-world knowledge, parsing in 60 and 99 constituents vs. the 212 and 204 of the standard parser, and this pruning of alternatives aids considerably in the handling of the conjunction. In the first sentence, the multiple senses of "take" combine productively with the multiple senses of "one" in the standard parser, but real-world information has the opportunity to restrict the NP senses of "one" as being unlikely in the real-world context. Similarly, the selected tomato and recently accessed triangle in the real world quickly bias the continuous parser towards appropriate interpretations of "it" and "that triangle".

	Parser	Continuous	%Work
Dialogue	7003	5221	75
Utt-02	286	203	71
Utt-05	592	160	27
Utt-06	472	148	31
Utt-09	523	483	92
Utt-15	436	444	102
Utt-21	352	528	150
2,9,15,21	1597	1658	104
Remainder	5406	3563	66

Table 5: Parser efficiency for the complete dialogue and for selected utterances, reported by number of constituents built. %Work shows the amount of work done by the continous parser compared to the base parser.

The efficiency gains of the continuous understanding parser are impressive, but the accuracy of the parser is also of paramount importance. The base parser found the correct parse for 26 of the 27 example sentences in the dialogue, an accuracy rate that represents the effects of selectional restrictions, domain-specific lexical preference rules, and years of hand-tuning rule weights to bias the parser towards those constructions most common in task-oriented dialogues. For the utterances in this dialogue, the parser's default preferences were correct, although note, as we saw in the proof-in-principle section, that the base parser is unable to adapt its parse to reflect the state of the world.

The continuous understanding parser's first parse was correct for 22 of the 27 example sentences, with new errors introduced in utterances 2, 9, 15, and 21. Note that all of these errors involve the same construction, namely selecting a shape with a decoration whose location is specified. All four errors involve the continuous parser attaching the final PP modifier to the verb rather than to the direct object NP.

In the cases of utterances 2 and 9, the user used an indefinite NP to refer to something that was unique in the real world, and since the KB is happy to approve of "a large triangle" and "a small square with a heart", of which there are multiple instances, the PPs "with nothing" and "on the corner" are attached to the verb rather than the direct object NP. If the definite article is used in the utterances, then both sentences are parsed correctly and the number of constituents built decreases to 170 and 451 respectively (from 203 and 483). Clearly, then, for at least these sentences, the bias towards unique definites and non-unique indefinites is too strong, although this may be in part due to users viewing objects which are not on the map as prototypes: that is, more like widgets on a toolbar than unique entities on a map. Further research is certainly necessary both in automatically

learning the weights assigned to the various categories of response and whether these categories are sufficient across the fruitcart dialogues.

Utterance 15 demonstrates that learning of the interpolation weights is probably necessary, quite possibly tying the interpolation weight to the function of the candidate NP in a larger, top-down structure. The bias in the continuous parser against "the corner", since corners are not unique, is enough to ensure that "a square with a diamond on the corner" is never constructed as a candidate NP (at least not before a complete parse is found). Again, it seems that the bias towards unique interpretations for definite NPs and multiple interpretations for indefinite NPs is too strong; here weights could be learned for a top-down parser based on the role that the NP was filling: i.e. as NP modifier or indirect object.

The final error introduced by the continuous parser is quite interesting. The user intended "a triangle with a star in the center" to refer to a triangle with a star centred on one edge, and the standard parser interpreted this with the correct low attachment. However, in the domain, triangles are containers, so there are no triangles with stars in their centers. In this case, the KB biases the search away from star-filled triangles, arguably an appropriate reaction. Examples like this bring up numerous questions surrounding evaluations of correctness for continuous parsers, an area of research we are hoping to explore in this domain.

All four errors that the continuous understanding parser made could have been mediated by another advisor; while the parse achieved struck the best balance for the continuous parser in terms of having each NP interpretable in the real world, there is no advice flowing back into the parser about how interpretable each VP is. A command advisor which requires that VPs make sense in the domain would likely have resulted in the search continuing to find an interpretation that struck a balance between the competing demands of having interpretable objects and interpretable commands.

An overall improvement of 26% in the parser's efficiency over the course of processing the dialogue is a strong result despite the decrease in accuracy for one particular grammatical construction. Moreover, the efficiency improvement came, for the most part, in the utterances that the continuous parser got correct; in the four offending sentences the parser builds 4% more constituents, and in the remaining 23 sentences the performance improvement is 34%.

Feedback and Speech Lattices

In true speech understanding systems, we rarely have the luxury of a perfect stream of unambiguous words flowing from the speech recognition component; much more likely is a lattice, a tightly packed encoding of the ambiguities remaining after the speech recognition process.

While multiple word ambiguities in the input will obviously complicate processing for any system, a continuous understanding system has significant resources to bring to bear to help resolve some of these ambiguities, or at least ensure that the final interpretation is correct in spite of them.

Consider the case of "put the square/pear near the flag in the park", where there is a recognition ambiguity at the site of the third word. In the extreme case, when no pear exists in the real world, feedback from reference will allow the parser to resolve the ambiguity almost immediately. Even if there are both pears and squares in the current instantiation of the world, in many cases, context will help with the disambiguation.

What's even better, these efficiency gains are orthogonal to those demonstrated above involving attachment ambiguities. Regardless of the presence or absence of pears, the system will make the same attachment decisions about squares that it would have anyway. Thus, with a lattice, the reference interaction prunes all search spaces in parallel, making future attachment decisions easier because of the smaller number of candidates.

It is anticipated that many of the speech recognition ambiguities packed in the lattice will be uninterpretable relative to domain knowledge and knowledge about the state of the real world, and that the increase in interpretation accuracy will have the rather nice side effect of more often selecting the correct underlying words from the input speech latice.

Future Work

We plan to have the first version of the full fruit carts system operational by early Fall 2005. At that time, we will be integrating the TRIPS components with an initial incremental ASR output developed by our collaborators at SRI, Andreas Stolcke and Liz Shriberg. With the full system running, we will be able to begin to evaluate the benefits of a continuous understanding system from a human-computer interface angle.

In the interim, we plan to continue research on a continuous understanding parser, working first with the transcripts of the fruitcart dialogues, and eventually with actual speech lattice input.

Tests against a hand-tuned parser for accuracy and efficiency are apt to give a somewhat distorted picture of results, especially where accuracy is concerned, and present problems with comparability of these results with others in the parsing literature. We intend to do further experiments with a version of the parser which learns probabilistic rule weights.

The feedback classes and the weights reported in this paper were chosen by intuition, and the interpolation parameter was selected in the context of a completely different domain; it is expected that learning values for these parameters from the marked-up fruitcart dialogues will improve the performance of the continuous parser.

As mentioned above, another avenue for development is the incorporation of an additional command advisor that will ensure that verb phrase commands can be appropriately interpreted in the domain; such a module would serve as a nice complement to the current advisor, which only provides input on the suitability of individual noun phrases.

Finally, we plan to provide the parser with lattice output from a speech recognizer in order to test our intuitions about continuous understanding providing significant advantages when applied to the word disambiguation and utterance segmentation tasks.

Conclusion

We designed the fruit carts task domain to be especially productive for research in continuous understanding, building on our earlier experiences with moving towards continuous understanding in less suitable domains. The fruit carts dialogues are a rich source of language that cannot accurately be interpreted without continuous understanding, and the proof-in-principle presented here suggests that the continuous understanding aspects of the fruit carts system will achieve major improvements in parsing efficiency and accuracy, and are likely to result in significant gains in semantic accuracy as well.

The dialogue results show significant improvements in efficiency, but highlight the need to develop more principled parameter selection in the continuous understanding domain. However, moving from layered sequential architectures to continuous understanding systems is a major paradigm shift in natural language processing, and we believe that the fruit carts system represents a significant advance in that direction.

Acknowledgements

This work was supported in part by the National Science Foundation, CISE grant number 0328810, and by the Defense Advanced Research Agency award NBCH-D-03-0010.

References

Allen, J.; Miller, B.; Ringger, E.; and Sikorski, T. 1996. Robust understanding in a dialogue system. In *Proc. of ACL-96*, 62–70.

Allopenna, P. D.; Magnuson, J. S.; and Tanenhaus, M. K. 1998. Tracking the time course of spoken word recognition using eye movements: evidence for continuous mapping models. *Journal of Memory and Language* 38:419–439.

Brill, E.; Florian, R.; Henderson, J. C.; and Mangu, L. 1998. Beyond n-grams: Can linguistic sophistication improve language modeling? In *Proc. of COLING-ACL-98*, 186–190.

Byron, D. K. 2000. Semantically enhanced pronouns. In *Proc. of DAARC2000: 3rd International Conference on Discourse Anaphora and Anaphor Resolution*.

Cooper, R. M. 1974. The control of eye fixation by the meaning of spoken language. *Cognitive Psychology* 6:84–107.

Dowding, J.; Gawron, J. M.; Appelt, D.; Bear, J.; Cherny, L.; Moore, R.; and Moran, D. 1993. Gemini: A natural language system for spoken-language understanding. In *Proc. of ACL-93*, 54–61.

Dowding, J.; Moore, R.; Andry, F.; and Moran, D. 1994. Interleaving syntax and semantics in an efficient bottom-up parser. In *Proc. of ACL-94*, 110–116.

Ferguson, G., and Allen, J. 1998. Trips: An integrated intelligent problem-solving assistant. In *Proc. of AAAI-98*, 567–572.

Ferguson, G.; Allen, J.; and Miller, B. 1996. Trains-95: Towards a mixed-initiative planning assistant. In *Proc. of the 3rd International Conference on Artificial Intelligence Planning Systems*, 70–77.

Horvitz, E., and Paek, T. 1999. A computational architecture for conversation. In *Proceedings of the Seventh International Conference on User Modeling*, 201–210. Banff, Canada: Springer Wien.

Jelinek, F., and Chelba, C. 1999. Putting language into language modeling. In *Proc. of Eurospeech-99*.

Kasper, W.; Krieger, H.-U.; Spilker, J.; and Weber, H. 1996. From word hypotheses to logical form: An efficient interleaved approach. In *Natural Language Processing and Speech Technology: Results of the 3rd KONVENS Conference*, 77–88.

Moore, R.; Appelt, D.; Dowding, J.; Gawron, J. M.; and Moran, D. 1995. Combining linguistic and statistical knowledge sources in natural-language processing for atis. In *Proc. ARPA Spoken Language Systems Technology Workshop*.

Noth, E.; Batliner, A.; Kiessling, A.; Kompe, R.; and Niemann, H. 2000. Verbmobil: The use of prosody in the linguistic components of a speech understanding system. *IEEE Transactions on Speech and Audio Processing* 8(5):519–531.

Phillips, C. 1996. *Order and Structure*. Ph.D. Dissertation, MIT.

Pinkal, M.; Rupp, C.; and Worm, K. 2000. Robust semantic processing of spoken language. In *Verbmobil: Foundations of Speech-to-Speech Translation*, 321–335.

Rose, C. P. 2000. A framework for robust semantic interpretation. In *Proc. of the Sixth Conference on Applied Natural Language Processing*.

Schuler, W. 2002. Interleaved semantic interpretation in environment-based parsing. In *Proc. of COLING-02*.

Sedivy, J. C.; Tanenhaus, M. K.; Chambers, C. G.; and Carlson, G. N. 1999. Achieving incremental semantic interpretation through contextual representation. *Cognition* 71:109–147.

Stoness, S. C.; Tetreault, J.; and Allen, J. 2004. Incremental parsing with reference interaction. In Keller, F.; Clark, S.; Crocker, M.; and Steedman, M., eds., *Proceedings of the ACL Workshop on Incremental Parsing*, 18–25.

Tanenhaus, M. K., and Spivey, M. 1996. Eye-tracking. *Language and Cognition Processes* 11(6):583–588.

Wang, W., and Harper, M. P. 2004. A statistical constraint dependency grammar (cdg) parser. In Keller, F.; Clark, S.; Crocker, M.; and Steedman, M., eds., *Proceedings of the ACL Workshop on Incremental Parsing*, 42–49.

Worm, K. 1998. A model for robust processing of spontaneous speech by integrating viable fragments. In *Proc. of COLING-ACL-98*, 1403–1407.

Zechner, K. 1998. Automatic construction of frame representations for spontaneous speech in unrestricted domains. In *Proc. of COLING-ACL-98*, 1448–1452.