Joint Satisfaction of Syntactic and Pragmatic Constraints Improves Incremental Spoken Language Understanding

Andreas Peldszus
University of Potsdam
Department for Linguistics
peldszus@uni-potsdam.de

Okko Buß
University of Potsdam
Department for Linguistics
okko@ling.uni-potsdam.de

Timo Baumann
University of Hamburg
Department for Informatics
baumann@informatik.uni-hamburg.de

David Schlangen
University of Bielefeld
Department for Linguistics
david.schlangen@uni-bielefeld.de

Abstract

We present a model of semantic processing of spoken language that (a) is robust against ill-formed input, such as can be expected from automatic speech recognisers, (b) respects both syntactic and pragmatic constraints in the computation of most likely interpretations, (c) uses a principled, expressive semantic representation formalism (RMRS) with a well-defined model theory, and (d) works continuously (producing meaning representations on a word-by-word basis, rather than only for full utterances) and incrementally (computing only the additional contribution by the new word, rather than re-computing for the whole utterance-so-far).

We show that the joint satisfaction of syntactic and pragmatic constraints improves the performance of the NLU component (around 10 % absolute, over a syntax-only baseline).

1 Introduction

Incremental processing for spoken dialogue systems (i. e., the processing of user input even while it still may be extended) has received renewed attention recently (Aist et al., 2007; Baumann et al., 2009; Buß and Schlangen, 2010; Skantze and Hjalmarsson, 2010; DeVault et al., 2011; Purver et al., 2011). Most of the practical work, however, has so far focussed on realising the potential for generating more responsive system behaviour through making available processing results earlier (e. g. (Skantze and Schlangen, 2009)), but has otherwise followed a typical pipeline architecture where processing results are passed only in one direction towards the next module. In this paper, we investigate whether the other potential advantage of incremental processing—providing "higher-level"-feedback to lower-level modules, in order to improve subsequent processing of the lower-level module—can be realised as well. Specifically, we experimented with giving a syntactic parser feedback about whether semantic readings of nominal phrases it is in the process of constructing have a denotation in the given context or not. Based on the assumption that speakers do plan their referring expressions so that they can successfully refer, we use this information to re-rank derivations; this in turn has an influence on how the derivations are expanded, given continued input. As we show in our experiments, for a corpus of realistic dialogue utterances collected in a Wizard-of-Oz setting, this strategy led to an absolute improvement in computing the intended denotation of around 10 % over a baseline (even more using a more permissive metric), both for manually transcribed test data as well as for the output of automatic speech recognition.

The remainder of this paper is structured as follows: We discuss related work in the next section, and then describe in general terms our model and its components. In Section 4 we then describe the data resources we used for the experiments and the actual implementation of the model, the baselines for comparison, and the results of our experiments. We close with a discussion and an outlook on future work.

2 Related Work

The idea of using real-world reference to inform syntactic structure building has been previously explored by a number of authors. Stoness et al. (2004, 2005) describe a proof-of-concept imple-

Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics, pages 514–523,
Avignon, France, April 23 - 27 2012.

mentation of a "continuous understanding" module that uses reference information in guiding a bottom-up chart-parser, which is evaluated on a single dialogue transcript. In contrast, our model uses a probabilistic top-down parser with beam search (following Roark (2001)) and is evaluated on a large number of real-world utterances as processed by an automatic speech recogniser. Similarly, DeVault and Stone (2003) describe a system that implements interaction between a parser and higher-level modules (in this case, even more principled, trying to prove presuppositions), which however is also only tested on a small, constructed data-set.

Schuler (2003) and Schuler et al. (2009) present a model where information about reference is used directly within the speech recogniser, and hence informs not only syntactic processing but also word recognition. To this end, the processing is folded into the decoding step of the ASR, and is realised as a hierarchical HMM. While technically interesting, this approach is by design non-modular and restricted in its syntactic expressivity.

The work presented here also has connections to work in psycholinguistics. Padó et al. (2009) present a model that combines syntactic and semantic models into one plausibility judgement that is computed incrementally. However, that work is evaluated for its ability to predict reading time data and not for its accuracy in computing meaning.

3 The Model

3.1 Overview

Described abstractly, the model computes the probability of a syntactic derivation (and its accompanying logical form) as a combination of a syntactic probability (as in a typical PCFG) and a semantic or pragmatic plausibility.[1] The pragmatic plausibility here comes from the presupposition that the speaker intended her utterance to successfully refer, i. e. to have a denotation in the current situation (a unique one, in the case of definite reference). Hence, readings that do have a denotation are preferred over those that do not.

[1]Note that, as described below, in the actual implementation the weights given to particular derivations are not real probabilities anymore, as derivations fall out of the beam and normalisation is not performed after re-weighting.

The components of our model are described in the following sections: first the parser which computes the syntactic probability in an incremental, top-down manner; the semantic construction algorithm which associates (underspecified) logical forms to derivations; the reference resolution component that computes the pragmatic plausibility; and the combination that incorporates the feedback from this pragmatic signal.

3.2 Parser

Roark (2001) introduces a strategy for incremental probabilistic top-down parsing and shows that it can compete with high-coverage bottom-up parsers. One of the reasons he gives for choosing a top-down approach is that it enables fully left-connected derivations, where at every processing step new increments directly find their place in the existing structure. This monotonically enriched structure can then serve as a context for incremental language understanding, as the author claims, although this part is not further developed by Roark (2001). He discusses a battery of different techniques for refining his results, mostly based on grammar transformations and on conditioning functions that manipulate a derivation probability on the basis of local linguistic and lexical information.

We implemented a basic version of his parser without considering additional conditioning or lexicalizations. However, we applied left-factorization to parts of the grammar to delay certain structural decisions as long as possible. The search-space is reduced by using beam search. To match the next token, the parser tries to expand the existing derivations. These derivations are stored in a priorized queue, which means that the most probable derivation will always be served first. Derivations resulting from rule expansions are kept in the current queue, derivations resulting from a successful lexical match are pushed in a new queue. The parser proceeds with the next most probable derivation until the current queue is empty or until a threshhold is reached at which remaining analyses are pruned. This threshhold is determined dynamically: If the probability of the current derivation is lower than the product of the best derivation's probability on the new queue, the number of derivations in the new queue, and a base beam factor (an initial parameter for the size of the search beam), then all further old deriva-

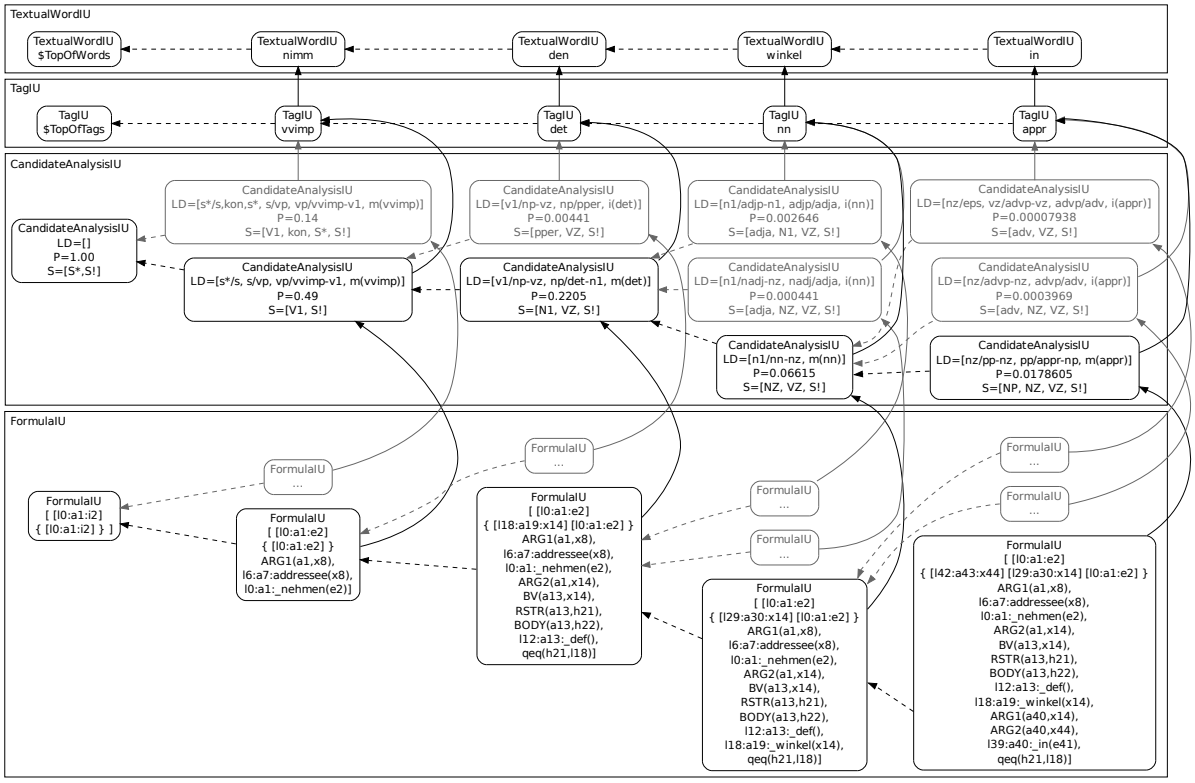

Figure 1: An example network of incremental units, including the levels of words, POS-tags, syntactic derivations and logical forms. See section 3 for a more detailed description.

tions are pruned. Due to probabilistic weighing and the left factorization of the rules, left recursion poses no direct threat in such an approach.

Additionally, we implemented three robust lexical operations: *insertions* consume the current token without matching it to the top stack item; *deletions* can "consume" a requested but actually non-existent token; *repairs* adjust unknown tokens to the requested token. These robust operations have strong penalties on the probability to make sure they will survive in the derivation only in critical situations. Additionally, only a single one of them is allowed to occur between the recognition of two adjacent input tokens.

Figure 1 illustrates this process for the first few words of the example sentence "nimm den winkel in der dritten reihe" (*take the bracket in the third row*), using the *incremental unit* (IU) model to represent increments and how they are linked; see (Schlangen and Skantze, 2009).[2] Here, syntactic derivations ("CandidateAnalysisIUs") are represented by three features: a list of the last parser actions of the derivation (LD), with rule expansions or (robust) lexical matches; the derivation probability (P); and the remaining stack (S), where S* is the grammar's start symbol and S! an explicit end-of-input marker. (To keep the Figure small, we artificially reduced the beam size and cut off alternatives paths, shown in grey.)

3.3 Semantic Construction Using RMRS

As a novel feature, we use for the representation of meaning increments (that is, the contributions of new words and syntactic constructions) as well as for the resulting logical forms the formalism *Robust Minimal Recursion Semantics* (Copestake, 2006). This is a representation formalism that was originally constructed for semantic underspecification (of scope and other phenomena) and then adapted to serve the purposes of semantics repre-

[2]Very briefly: rounded boxes in the Figures represent IUs, and dashed arrows link an IU to its predecessor on the same level, where the levels correspond to processing stages. The Figure shows the levels of input words, POS-tags, syntactic derivations and logical forms. Multiple IUs sharing the same predecessor can be regarded as alternatives. Solid arrows indicate which information from a previous level an IU is grounded in (based on); here, every semantic IU is grounded in a syntactic IU, every syntactic IU in a POS-tag-IU, and so on.

sentations in heterogeneous situations where information from deep and shallow parsers must be combined. In RMRS, meaning representations of a first order logic are underspecified in two ways: First, the scope relationships can be underspecified by splitting the formula into a list of *elementary predications* (EP) which receive a label ℓ and are explicitly related by stating scope constraints to hold between them (e.g. *qeq*-constraints). This way, all scope readings can be compactly represented. Second, RMRS allows underspecification of the predicate-argument-structure of EPs. Arguments are bound to a predicate by anchor variables a, expressed in the form of an *argument relation* ARGREL(a,x). This way, predicates can be introduced without fixed arity and arguments can be introduced without knowing which predicates they are arguments of. We will make use of this second form of underspecification and enrich lexical predicates with arguments incrementally.

Combining two RMRS structures involves at least joining their list of EPs and ARGRELs and of scope constraints. Additionally, equations between the variables can connect two structures, which is an essential requirement for semantic construction. A semantic algebra for the combination of RMRSs in a non-lexicalist setting is defined in (Copestake, 2007). Unsaturated semantic increments have open slots that need to be filled by what is called the *hook* of another structure. Hook and slot are triples $[\ell{:}a{:}x]$ consisting of a label, an anchor and an index variable. Every variable of the hook is equated with the corresponding one in the slot. This way the semantic representation can grow monotonically at each combinatory step by simply adding predicates, constraints and equations.

Our approach differs from (Copestake, 2007) only in the organisation of the slots: In an incremental setting, a proper semantic representation is desired for every single state of growth of the syntactic tree. Typically, RMRS composition assumes that the order of semantic combination is parallel to a bottom-up traversal of the syntactic tree. Yet, this would require *for every incremental step* first to calculate an adequate underspecified semantic representation for the projected nodes on the lower right border of the tree and then to proceed with the combination not only of the new semantic increments but of the complete tree. For our purposes, it is more elegant to proceed with semantic combination in synchronisation with the syntactic expansion of the tree, i.e. in a top-down left-to-right fashion. This way, no underspecification of projected nodes and no re-interpretation of already existing parts of the tree is required. This, however, requires adjustments to the slot structure of RMRS. Left-recursive rules can introduce multiple slots of the same sort before they are filled, which is not allowed in the classic (R)MRS semantic algebra, where only one named slot of each sort can be open at a time. We thus organize the slots as a stack of unnamed slots, where multiple slots of the same sort can be stored, but only the one on top can be accessed. We then define a basic combination operation equivalent to forward function composition (as in standard lambda calculus, or in CCG (Steedman, 2000)) and combine substructures in a principled way across multiple syntactic rules without the need to represent slot names.

Each lexical items receives a generic representation derived from its lemma and the basic semantic type (individual, event, or underspecified denotations), determined by its POS tag. This makes the grammar independent of knowledge about what later (semantic) components will actually be able to process ("understand").[3] Parallel to the production of syntactic derivations, as the tree is expanded top-down left-to-right, semantic macros are activated for each syntactic rule, composing the contribution of the new increment. This allows for a monotonic semantics construction process that proceeds in lockstep with the syntactic analysis.

Figure 1 (in the "FormulaIU" box) illustrates the results of this process for our example derivation. Again, alternatives paths have been cut to keep the size of the illustration small. Notice that, apart from the end-of-input marker, the stack of semantic slots (in curly brackets) is always synchronized with the parser's stack.

3.4 Computing Noun Phrase Denotations

Formally, the task of this module is, given a model $\mathcal{M}$ of the current context, to compute the set of all variable assignments such that $\mathcal{M}$ satisfies ϕ: $\mathcal{G} = \{g \mid \mathcal{M} \models^g \phi\}$. If $|\mathcal{G}| > 1$, we say that ϕ refers ambiguously; if $|\mathcal{G}| = 1$, it refers uniquely;

[3] This feature is not used in the work presented here, but it could be used for enabling the system to learn the meaning of unknown words.

and if $|\mathcal{G}| = 0$, it fails to refer. This process does not work directly on RMRS formulae, but on extracted and unscoped first-order representations of their nominal content.

3.5 Parse Pruning Using Reference Information

After all possible syntactic hypotheses at an increment have been derived by the parser and the corresponding semantic representations have been constructed, reference resolution information can be used to re-rank the derivations. If pragmatic feedback is enabled, the probability of every reprentation that does not resolve in the current context is degraded by a constant factor (we used 0.001 in our experiments described below, determined by experimentation). The degradation thus changes the derivation order in the parsing queue for the next input item and increases the chances of degraded derivations to be pruned in the following parsing step.

4 Experiments and Results

4.1 Data

We use data from the Pentomino puzzle piece domain (which has been used before for example by (Fernández and Schlangen, 2007; Schlangen et al., 2009)), collected in a Wizard-of-Oz study. In this specific setting, users gave instructions to the system (the wizard) in order to manipulate (select, rotate, mirror, delete) puzzle pieces on an upper board and to put them onto a lower board, reaching a pre-specified goal state. Figure 2 shows an example configuration. Each participant took part in several rounds in which the distinguishing characteristics for puzzle pieces (color, shape, proposed name, position on the board) varied widely. In total, 20 participants played 284 games.

We extracted the semantics of an utterance from the wizard's response action. In some cases, such a mapping was not possible to do (e. g. because the wizard did not perform a next action, mimicking a non-understanding by the system), or potentially unreliable (if the wizard performed several actions at or around the end of the utterance). We discarded utterances without a clear semantics alignment, leaving 1687 semantically annotated user utterances. The wizard of course was able to use her model of the previous discourse for resolving references, including anaphoric ones; as

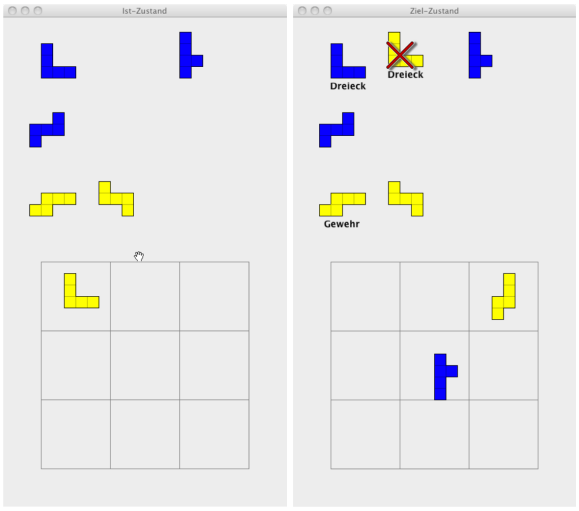

Figure 2: The game board used in the study, as presented to the player: (a) the current state of the game on the left, (b) the goal state to be reached on the right.

our study does not focus on these, we have disregarded another 661 utterances in which pieces are referred to by pronouns, leaving us with 1026 utterances for evaluation. These utterances contained on average 5.2 words (median 5 words; std dev 2 words).

In order to test the robustness of our method, we generated speech recognition output using an acoustic model trained for spontaneous (German) speech. We used leave-one-out language model training, i. e. we trained a language model for every utterance to be recognized which was based on all the other utterances in the corpus. Unfortunately, the audio recordings of the first recording day were too quiet for successful recognition (with a deletion rate of 14 %). We thus decided to limit the analysis for speech recognition output to the remaining 633 utterances from the other recording days. On this part of the corpus word error rate (WER) was at 18 %.

The subset of the full corpus that we used for evaluation, with the utterances selected according to the criteria described above, nevertheless still only consists of natural, spontaneous utterances (with all the syntactic complexity that brings) that are representative for interactions in this type of domain.

4.2 Grammar and Resolution Model

The grammar used in our experiments was hand-constructed, inspired by a cursory inspection of the corpus and aiming to reach good coverage

Words	Predicates	Status
nimm	nimm(e)	-1
nimm den	nimm(e,x) def(x)	0
nimm den Winkel	nimm(e,x) def(x) winkel(x)	0
nimm den Winkel in	nimm(e,x) def(x) winkel(x) in(x,y)	0
nimm den Winkel in der	nimm(e,x) def(x) winkel(x) in(x,y) def(y)	0
nimm den Winkel in der dritten	nimm(e,x) def(x) winkel(x) in(x,y) def(y) third(y)	1
nimm den Winkel in der dritten Reihe	nimm(e,x) def(x) winkel(x) in(x,y) def(y) third(y) row(y)	1

Table 1: Example of logical forms (flattened into first-order base-language formulae) and reference resolution results for incrementally parsing and resolving 'nimm den winkel in der dritten reihe'

for a core fragment. We created 30 rules, whose weights were also set by hand (as discussed below, this is an obvious area for future improvement), sparingly and according to standard intuitions. When parsing, the first step is the assignment of a POS tag to each word. This is done by a simple lookup tagger that stores the most frequent tag for each word (as determined on a small subset of our corpus).[4]

The situation model used in reference resolution is automatically derived from the internal representation of the current game state. (This was recorded in an XML-format for each utterance in our corpus.) Variable assignments were then derived from the relevant *nominal* predicate structures,[5] consisting of extracted simple predications, e. g. *red*(x) and *cross*(x) for the NP in a phrase such as "take the red cross". For each unique predicate argument X in these EP structures (such as as x above), the set of domain objects that satisfied all predicates of which X was an argument were determined. For example for the phrase above, X mapped to all elements that were *red* and *cross*es.

Finally, the size of these sets was determined: no elements, one element, or multiple elements, as described above. Emptiness of at least one set denoted that no resolution was possible (for instance, if no red crosses were available, x's set was empty), uniqueness of all sets denoted that an exact resolution was possible while multiple elements in at least some sets denoted ambiguity. This status was then leveraged for parse pruning, as per Section 3.5.

A more complex example using the scene depicted in Figure 2 and the sentence "nimm den winkel in der dritten reihe" (*take the bracket in the third row*) is shown in Table 1. The first column shows the incremental word hypothesis string, the second the set of predicates derived from the most recent RMRS representation and the third the resolution status (-1 for no resolution, 0 for some resolution and 1 for a unique resolution).

4.3 Baselines and Evaluation Metric

4.3.1 Variants / Baselines

To be able to accurately quantify and assess the effect of our reference-feedback strategy, we implemented different variants / baselines. These all differ in how, at each step, the reading is determined that is evaluated against the gold standard, and are described in the following:

In the **Just Syntax (JS)** variant, we simply take single-best derivation, as determined by syntax alone and evaluate this.

The **External Filtering (EF)** variant adds information from reference resolution, but keeps it separate from the parsing process. Here, we look at the 5 highest ranking derivations (as determined by syntax alone), and go through them beginning at the highest ranked, picking the first derivation where reference resolution can be performed uniquely; this reading is then put up for evaluation. If there is no such reading, the highest ranking one will be put forward for evaluation (as in JS).

Syntax/Pragmatics Interaction (SPI) is the variant described in the previous section. Here, all active derivations are sent to the reference resolution module, and are re-weighted as described above; after this has been done, the highest-ranking reading is evaluated.

Finally, the **Combined Interaction and Filtering (CIF)** variant combines the previous two strategies, by using reference-feedback in computing the ranking for the derivations, and then

[4] A more sophisticated approach has recently been proposed by Beuck et al. (2011); this could be used in our setup.

[5] The domain model did not allow making a plausibility judgement based on verbal resolution.

again using reference-information to identify the most promising reading within the set of 5 highest ranking ones.

4.3.2 Metric

When a reading has been identified according to one of these methods, a score s is computed as follows: $s = 1$, if the correct referent (according to the gold standard) is computed as the denotation for this reading; $s = 0$ if no unique referent can be computed, but the correct one is part of the set of possible referents; $s = -1$ if no referent can be computed at all, or the correct one is not part of the set of those that are computed.

As this is done incrementally for each word (adding the new word to the parser chart), for an utterance of length m we get a sequence of m such numbers. (In our experiments we treat the "end of utterance" signal as a pseudo-word, since knowing that an utterance has concluded allows the parser to close off derivations and remove those that are still requiring elements. Hence, we in fact have sequences of $m+1$ numbers.) A combined score for the whole utterance is computed according to the following formula:

$$su = \sum_{n=1}^{m} (s_n * n/m)$$

(where s_n is the score at position n). The factor n/m causes "later" decisions to count more towards the final score, reflecting the idea that it is more to be expected (and less harmful) to be wrong early on in the utterance, whereas the longer the utterance goes on, the more pressing it becomes to get a correct result (and the more damaging if mistakes are made).[6]

Note that this score is not normalised by utterance length m; the maximally achievable score being $(m + 1)/2$. This has the additional effect of increasing the weight of long utterances when averaging over the score of all utterances; we see this as desirable, as the analysis task becomes harder the longer the utterance is.

We use success in resolving reference to evaluate the performance of our parsing and semantic construction component, where more traditionally, metrics like parse bracketing accuracy might be used. But as we are building this module for an interactive system, ultimately, accuracy in recovering meaning is what we are interested in, and so we see this not just as a proxy, but actually as a more valuable metric. Moreover, this metric can be applied at each incremental step, which is not clear how to do with more traditional metrics.

[6]This metric compresses into a single number some of the concerns of the incremental metrics developed in (Baumann et al., 2011), which can express more fine-grainedly the temporal development of hypotheses.

4.4 Experiments

Our parser, semantic construction and reference resolution modules are implemented within the InproTK toolkit for incremental spoken dialogue systems development (Schlangen et al., 2010). In this toolkit, incremental hypotheses are modified as more information becomes available over time. Our modules support all such modifications (i. e. also allow to revert their states and output if word input is revoked).

As explained in Section 4.1, we used offline recognition results in our evaluation. However, the results would be identical if we were to use the incremental speech recognition output of InproTK directly.

The system performs several times faster than real-time on a standard workstation computer. We thus consider it ready to improve practical end-to-end incremental systems which perform within-turn actions such as those outlined in (Buß and Schlangen, 2010).

The parser was run with a base-beam factor of 0.01; this parameter may need to be adjusted if a larger grammar was used.

4.5 Results

Table 2 shows an overview of the experiment results. The table lists, separately for the manual transcriptions and the ASR transcripts, first the number of times that the final reading did not resolve at all, or to a wrong entitiy; did not uniquely resolve, but included the correct entity in its denotiation; or did uniquely resolve to the correct entity (-1, 0, and 1, respectively). The next lines show "strict accuracy" (proportion of "1" among all results) at the end of utterance, and "relaxed accuracy" (which allows ambiguity, i.e., is the set $\{0, 1\}$). *incr.scr* is the incremental score as described above, which includes in the evaluation the development of references and not just the final state. (And in that sense, is the most appropriate metric here, as it captures the incremental behaviour.) This score is shown both as absolute

		JS	EF	SPI	CIF
transcript	−1	563	518	364	363
	0	197	198	267	268
	1	264	308	392	392
	str.acc.	25.7 %	30.0 %	38.2 %	38.2 %
	rel.acc.	44.9 %	49.3 %	64.2 %	64.3 %
	incr.scr	−1568	−1248	−536	−504
	avg.incr.scr	−1.52	−1.22	−0.52	−0.49
recogntion	−1	362	348	254	255
	0	122	121	173	173
	1	143	158	196	195
	str.acc.	22.6 %	25.0 %	31.0 %	30.8 %
	rel.acc.	41.2 %	44.1 %	58.3 %	58.1 %
	incr.scr	−1906	−1730	−1105	−1076
	avg.incr.scr	−1.86	−1.69	−1.01	−1.05

Table 2: Results of the Experiments. See text for explanation of metrics.

number as well as averaged for each utterance.

As these results show, the strategy of providing the parser with feedback about the real-world utility of constructed phrases (in the form of reference decisions) improves the parser, in the sense that it helps the parser to successfully retrieve the intended meaning more often compared to an approach that only uses syntactic information (**JS**) or that uses pragmatic information only outside of the main programme: 38.2 % strict or 64.2 % relaxed for **SPI** over 25.7 % / 44.9 % for **JS**, an absolute improvement of 12.5 % for strict or even more, 19.3 %, for the relaxed metric; the incremental metric shows that this advantage holds not only at the final word, but also consistently within the utterance, the average incremental score for an utterance being −0.49 for **SPI** and −1.52 for **JS**. The improvement is somewhat smaller against the variant that uses some reference information, but does not integrate this into the parsing process (**EF**), but it is still consistently present. Adding such n-best-list processing to the output of the parser+reference-combination (as variant **CIF** does) finally does not further improve the performance noticeably. When processing partially defective material (the output of the speech recogniser), the difference between the variants is maintained, showing a clear advantage of **SPI**, although performance of all variants is degraded somewhat.

Clearly, accuracy is rather low for the baseline condition (**JS**); this is due to the large number of non-standard constructions in our spontaneous material (e.g., utterances like "löschen, unten" (*delete, bottom*) which we did not try to cover with syntactic rules, and which may not even contain NPs. The **SPI** condition can promote derivations resulting from robust rules (here, *deletion*) which then can refer. In general though state-of-the art grammar engineering may narrow the gap between **JS** and **SPI** – this remains to be tested – but we see as an advantage of our approach that it can improve over the (easy-to-engineer) set of core grammar rules.

5 Conclusions

We have described a model of semantic processing of natural, spontaneous speech that strives to jointly satisfy syntactic and pragmatic constraints (the latter being approximated by the assumption that referring expressions are intended to indeed successfully refer in the given context). The model is robust, accepting also input of the kind that can be expected from automatic speech recognisers, and incremental, that is, can be fed input on a word-by-word basis, computing at each increment only exactly the contribution of the new word. Lastly, as another novel contribution, the model makes use of a principled formalism for semantic representation, RMRS (Copestake, 2006).

While the results show that our approach of combining syntactic and pragmatic information can work in a real-world setting on realistic data—previous work in this direction has so far

only been at the proof-of-concept stage—there is much room for improvement. First, we are now exploring ways of bootstrapping a grammar and derivation weights from hand-corrected parses. Secondly, we are looking at making the variable assignment / model checking function probabilistic, assigning probabilities (degree of strength of belief) to candidate resolutions (as for example the model of Schlangen et al. (2009) does). Another next step—which will be very easy to take, given the modular nature of the implementation framework that we have used—will be to integrate this component into an interactive end-to-end system, and testing other domains in the process.

Acknowledgements We thank the anonymous reviewers for their helpful comments. The work reported here was supported by a DFG grant in the Emmy Noether programme to the last author and a stipend from DFG-CRC (SFB) 632 to the first author.

References

Gregory Aist, James Allen, Ellen Campana, Carlos Gomez Gallo, Scott Stoness, Mary Swift, and Michael K. Tanenhaus. 2007. Incremental understanding in human-computer dialogue and experimental evidence for advantages over nonincremental methods. In *Proceedings of Decalog 2007, the 11th International Workshop on the Semantics and Pragmatics of Dialogue*, Trento, Italy.

Timo Baumann, Michaela Atterer, and David Schlangen. 2009. Assessing and improving the performance of speech recognition for incremental systems. In *Proceedings of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies (NAACL HLT) 2009 Conference*, Boulder, Colorado, USA, May.

Timo Baumann, Okko Buß, and David Schlangen. 2011. Evaluation and optimization of incremental processors. *Dialogue and Discourse*, 2(1):113–141.

Niels Beuck, Arne Köhn, and Wolfgang Menzel. 2011. Decision strategies for incremental pos tagging. In *Proceedings of the 18th Nordic Conference of Computational Linguistics, NODALIDA-2011*, Riga, Latvia.

Okko Buß and David Schlangen. 2010. Modelling sub-utterance phenomena in spoken dialogue systems. In *Proceedings of the 14th International Workshop on the Semantics and Pragmatics of Dialogue (Pozdial 2010)*, pages 33–41, Poznan, Poland, June.

Ann Copestake. 2006. Robust minimal recursion semantics. Technical report, Cambridge Computer Lab. Unpublished draft.

Ann Copestake. 2007. Semantic composition with (robust) minimal recursion semantics. In *Proceedings of the Workshop on Deep Linguistic Processing*, DeepLP '07, pages 73–80, Stroudsburg, PA, USA. Association for Computational Linguistics.

David DeVault and Matthew Stone. 2003. Domain inference in incremental interpretation. In *Proceedings of ICOS 4: Workshop on Inference in Computational Semantics*, Nancy, France, September. INRIA Lorraine.

David DeVault, Kenji Sagae, and David Traum. 2011. Incremental Interpretation and Prediction of Utterance Meaning for Interactive Dialogue. *Dialogue and Discourse*, 2(1):143–170.

Raquel Fernández and David Schlangen. 2007. Referring under restricted interactivity conditions. In Simon Keizer, Harry Bunt, and Tim Paek, editors, *Proceedings of the 8th SIGdial Workshop on Discourse and Dialogue*, pages 136–139, Antwerp, Belgium, September.

Ulrike Padó, Matthew W Crocker, and Frank Keller. 2009. A probabilistic model of semantic plausibility in sentence processing. *Cognitive Science*, 33(5):794–838.

Matthew Purver, Arash Eshghi, and Julian Hough. 2011. Incremental semantic construction in a dialogue system. In J. Bos and S. Pulman, editors, *Proceedings of the 9th International Conference on Computational Semantics (IWCS)*, pages 365–369, Oxford, UK, January.

Brian Roark. 2001. *Robust Probabilistic Predictive Syntactic Processing: Motivations, Models, and Applications*. Ph.D. thesis, Department of Cognitive and Linguistic Sciences, Brown University.

David Schlangen and Gabriel Skantze. 2009. A general, abstract model of incremental dialogue processing. In *EACL '09: Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 710–718. Association for Computational Linguistics, mar.

David Schlangen, Timo Baumann, and Michaela Atterer. 2009. Incremental reference resolution: The task, metrics for evaluation, and a bayesian filtering model that is sensitive to disfluencies. In *Proceedings of SIGdial 2009, the 10th Annual SIGDIAL Meeting on Discourse and Dialogue*, London, UK, September.

David Schlangen, Timo Baumann, Hendrik Buschmeier, Okko Buß, Stefan Kopp, Gabriel Skantze, and Ramin Yaghoubzadeh. 2010. Middleware for Incremental Processing in Conversational Agents. In *Proceedings of SigDial 2010*, Tokyo, Japan, September.

William Schuler, Stephen Wu, and Lane Schwartz. 2009. A framework for fast incremental interpretation during speech decoding. *Computational Linguistics*, 35(3).

William Schuler. 2003. Using model-theoretic semantic interpretation to guide statistical parsing and word recognition in a spoken language interface. In *Proceedings of the 41st Meeting of the Association for Computational Linguistics (ACL 2003)*, Sapporo, Japan. Association for Computational Linguistics.

Gabriel Skantze and Anna Hjalmarsson. 2010. Towards incremental speech generation in dialogue systems. In *Proceedings of the SIGdial 2010 Conference*, pages 1–8, Tokyo, Japan, September.

Gabriel Skantze and David Schlangen. 2009. Incremental dialogue processing in a micro-domain. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2009)*, pages 745–753, Athens, Greece, March.

Mark Steedman. 2000. *The Syntactic Process*. MIT Press, Cambridge, Massachusetts.

Scott C. Stoness, Joel Tetreault, and James Allen. 2004. Incremental parsing with reference interaction. In *Proceedings of the Workshop on Incremental Parsing at the ACL 2004*, pages 18–25, Barcelona, Spain, July.

Scott C. Stoness, James Allen, Greg Aist, and Mary Swift. 2005. Using real-world reference to improve spoken language understanding. In *AAAI Workshop on Spoken Language Understanding*, pages 38–45.