In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (NAACL/HLT-07)*. pp. 172-179, Rochester, NY, April 2007.

# Generation by Inverting a Semantic Parser That Uses Statistical Machine Translation

**Yuk Wah Wong and Raymond J. Mooney**
Department of Computer Sciences
The University of Texas at Austin
1 University Station C0500
Austin, TX 78712-0233, USA
{ywwong,mooney}@cs.utexas.edu

## Abstract

This paper explores the use of statistical machine translation (SMT) methods for tactical natural language generation. We present results on using phrase-based SMT for learning to map meaning representations to natural language. Improved results are obtained by inverting a semantic parser that uses SMT methods to map sentences into meaning representations. Finally, we show that hybridizing these two approaches results in still more accurate generation systems. Automatic and human evaluation of generated sentences are presented across two domains and four languages.

## 1 Introduction

This paper explores the use of statistical machine translation (SMT) methods in natural language generation (NLG), specifically the task of mapping statements in a formal *meaning representation language* (MRL) into a natural language (NL), i.e. tactical generation. Given a corpus of NL sentences each paired with a formal *meaning representation* (MR), it is easy to use SMT to construct a tactical generator, i.e. a statistical model that translates MRL to NL. However, there has been little, if any, research on exploiting recent SMT methods for NLG.

In this paper we present results on using a recent phrase-based SMT system, PHARAOH (Koehn et al., 2003), for NLG.[1] Although moderately effective, the inability of PHARAOH to exploit the formal structure and grammar of the MRL limits its accuracy. Unlike natural languages, MRLs typically have a simple, formal syntax to support effective automated processing and inference. This MRL structure can also be used to improve language generation.

Tactical generation can also be seen as the inverse of *semantic parsing*, the task of mapping NL sentences to MRs. In this paper, we show how to "invert" a recent SMT-based semantic parser, WASP (Wong and Mooney, 2006), in order to produce a more effective generation system. WASP exploits the formal syntax of the MRL by learning a translator (based on a statistical synchronous context-free grammar) that maps an NL sentence to a linearized parse-tree of its MR rather than to a flat MR string. In addition to exploiting the formal MRL grammar, our approach also allows the same learned grammar to be used for both parsing and generation, an elegant property that has been widely advocated (Kay, 1975; Jacobs, 1985; Shieber, 1988). We present experimental results in two domains previously used to test WASP's semantic parsing ability: mapping NL queries to a formal database query language, and mapping NL soccer coaching instructions to a formal robot command language. $\text{WASP}^{-1}$ is shown to produce a more accurate NL generator than PHARAOH.

We also show how the idea of generating from linearized parse-trees rather than flat MRs, used effectively in $\text{WASP}^{-1}$, can also be exploited in PHARAOH. A version of PHARAOH that exploits this approach is experimentally shown to produce more accurate generators that are more competitive with $\text{WASP}^{-1}$'s. Finally, we also show how

[1] We also tried IBM Model 4/REWRITE (Germann, 2003), a word-based SMT system, but it gave much worse results.

```
((bowner our {4})
 (do our {6} (pos (left (half our)))))
```
*If our player 4 has the ball, then our player 6 should stay in the left side of our half.*

(a) CLANG

```
answer(state(traverse_1(riverid('ohio'))))
```
*What states does the Ohio run through?*

(b) GEOQUERY

Figure 1: Sample meaning representations

aspects of PHARAOH's phrase-based model can be used to improve WASP$^{-1}$, resulting in a hybrid system whose overall performance is the best.

## 2 MRLs and Test Domains

In this work, we consider input MRs with a hierarchical structure similar to Moore (2002). The only restriction on the MRL is that it be defined by an available unambiguous context-free grammar (CFG), which is true for almost all computer languages. We also assume that the order in which MR predicates appear is relevant, i.e. the order can affect the meaning of the MR. Note that the order in which predicates appear need not be the same as the word order of the target NL, and therefore, the content planner need not know about the target NL grammar (Shieber, 1993).

To ground our discussion, we consider two application domains which were originally used to demonstrate semantic parsing. The first domain is ROBOCUP. In the ROBOCUP Coach Competition (www.robocup.org), teams of agents compete in a simulated soccer game and receive coach advice written in a formal language called CLANG (Chen et al., 2003). The task is to build a system that translates this formal advice into English. Figure 1(a) shows a piece of sample advice.

The second domain is GEOQUERY, where a functional, variable-free query language is used for querying a small database on U.S. geography (Kate et al., 2005). The task is to translate formal queries into NL. Figure 1(b) shows a sample query.

## 3 Generation using SMT Methods

In this section, we show how SMT methods can be used to construct a tactical generator. This is in contrast to existing work that focuses on the use of NLG in interlingual MT (Whitelock, 1992), in which the roles of NLG and MT are switched. We first consider using a phrase-based SMT system, PHARAOH, for NLG. Then we show how to invert an SMT-based semantic parser, WASP, to produce a more effective generation system.

### 3.1 Generation using PHARAOH

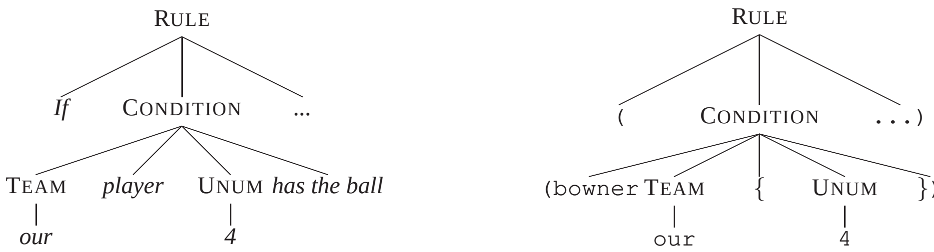
PHARAOH (Koehn et al., 2003) is an SMT system that uses phrases as basic translation units. During decoding, the source sentence is segmented into a sequence of phrases. These phrases are then reordered and translated into phrases in the target language, which are joined together to form the output sentence. Compared to earlier word-based methods such as IBM Models (Brown et al., 1993), phrase-based methods such as PHARAOH are much more effective in producing idiomatic translations, and are currently the best performing methods in SMT (Koehn and Monz, 2006).

To use PHARAOH for NLG, we simply treat the source MRL as an NL, so that phrases in the MRL are sequences of MR tokens. Note that the grammaticality of MRs is not an issue here, as they are given as input.

### 3.2 WASP: The Semantic Parsing Algorithm

Before showing how generation can be performed by inverting a semantic parser, we present a brief overview of WASP (Wong and Mooney, 2006), the SMT-based semantic parser on which this work is based.

To describe WASP, it is best to start with an example. Consider the task of translating the English sentence in Figure 1(a) into CLANG. To do this, we may first generate a parse tree of the input sentence. The meaning of the sentence is then obtained by combining the meanings of the phrases. This process can be formalized using a *synchronous context-free grammar* (SCFG), originally developed as a grammar formalism that combines syntax analysis and code generation in compilers (Aho and Ullman, 1972). It has been used in syntax-based SMT to model the translation of one NL to another (Chiang, 2005). A derivation for a SCFG gives rise to multiple isomorphic parse trees. Figure 2 shows a partial parse of the sample sentence and its corre-




(a) English (b) CLANG

Figure 2: Partial parse trees for the CLANG statement and its English gloss shown in Figure 1(a)

sponding CLANG parse from which an MR is constructed. Note that the two parse trees are isomorphic (ignoring terminals).

Each SCFG rule consists of a non-terminal, $X$, on the left-hand side (LHS), and a pair of strings, $\langle \alpha, \beta \rangle$, on the right-hand side (RHS). The non-terminals in $\beta$ are a permutation of the non-terminals in $\alpha$ (indices are used to show their correspondence). In WASP, $\alpha$ denotes an NL phrase, and $X \rightarrow \beta$ is a production of the MRL grammar. Below are the SCFG rules that generate the parses in Figure 2:

$$\begin{aligned}
\textsc{Rule} &\rightarrow \langle \textit{if}\ \textsc{Condition}_{\boxed{1}}\ ,\ \textsc{Directive}_{\boxed{2}}\ .\ ,\\
&\qquad \texttt{(}\textsc{Condition}_{\boxed{1}}\ \textsc{Directive}_{\boxed{2}}\texttt{)} \rangle\\
\textsc{Condition} &\rightarrow \langle \textsc{Team}_{\boxed{1}}\ \textit{player}\ \textsc{Unum}_{\boxed{2}}\ \textit{has the}\\
&\qquad \textit{ball}\ ,\ \texttt{(bowner}\ \textsc{Team}_{\boxed{1}}\ \{\textsc{Unum}_{\boxed{2}}\}\texttt{)} \rangle\\
\textsc{Team} &\rightarrow \langle \textit{our}\ ,\ \texttt{our} \rangle\\
\textsc{Unum} &\rightarrow \langle \textit{4}\ ,\ \texttt{4} \rangle
\end{aligned}$$

All derivations start with a pair of co-indexed start symbols of the MRL grammar, $\langle S_{\boxed{1}}, S_{\boxed{1}} \rangle$, and each step involves the rewriting of a pair of co-indexed non-terminals (by $\alpha$ and $\beta$, respectively). Given an input sentence, $\mathbf{e}$, the task of semantic parsing is to find a derivation that yields $\langle \mathbf{e}, \mathbf{f} \rangle$, so that $\mathbf{f}$ is an MR translation of $\mathbf{e}$.

Parsing with WASP requires a set of SCFG rules. These rules are learned using a word alignment model, which finds an optimal mapping from words to MR predicates given a set of training sentences and their correct MRs. Word alignment models have been widely used for lexical acquisition in SMT (Brown et al., 1993; Koehn et al., 2003). To use a word alignment model in the semantic parsing scenario, we can treat the MRL simply as an NL, and MR tokens as words, but this often leads to poor results. First, not all MR tokens carry specific meanings. For example, in CLANG, parentheses and braces are delimiters that are semantically vacuous. Such tokens can easily confuse the word alignment model. Second, MR tokens may exhibit polysemy. For example, the CLANG predicate `pt` has three meanings based on the types of arguments it is given (Chen et al., 2003). Judging from the `pt` token alone, the word alignment model would not be able to identify its exact meaning.

A simple, principled way to avoid these difficulties is to represent an MR using a list of productions used to generate it. This list is used in lieu of the MR in a word alignment. Figure 3 shows an example. Here the list of productions corresponds to the top-down, left-most derivation of an MR. For each MR there is a unique linearized parse-tree, since the MRL grammar is unambiguous. Note that the structure of the parse tree is preserved through linearization. This allows us to extract SCFG rules in a bottom-up manner, assuming the alignment is $n$-to-1 (each word is linked to at most one production). Extraction starts with productions whose RHS is all terminals, followed by those with non-terminals. (Details can be found in Wong and Mooney (2006).) The rules extracted from Figure 3 would be almost the same as those used in Figure 2, except the one for `bowner`: $\textsc{Condition} \rightarrow \langle \textsc{Team}_{\boxed{1}}\ \textit{player}\ \textsc{Unum}_{\boxed{2}}\ \textit{has}\ (1)\ \textit{ball},\ \texttt{(bowner}\ \textsc{Team}_{\boxed{1}}\ \{\textsc{Unum}_{\boxed{2}}\}\texttt{)} \rangle$. The token $(1)$ denotes a *word gap* of size 1, due to the unaligned word *the* that comes between *has* and *ball*. It can be seen as a non-terminal that expands to at most one word, allowing for some flexibility in pattern matching.

In WASP, GIZA++ (Och and Ney, 2003) is used to obtain the best alignments from the training examples. Then SCFG rules are extracted from these alignments. The resulting SCFG, however, can be

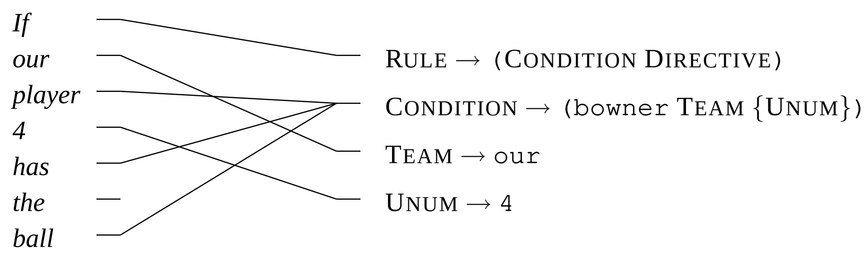



Figure 3: Partial word alignment for the CLANG statement and its English gloss shown in Figure 1(a)

ambiguous. Therefore, a maximum-entropy model that defines the conditional probability of derivations ($\mathbf{d}$) given an input sentence ($\mathbf{e}$) is used for disambiguation:

$$\Pr_\lambda(\mathbf{d}|\mathbf{e}) = \frac{1}{Z_\lambda(\mathbf{e})} \exp \sum_i \lambda_i f_i(\mathbf{d}) \qquad (1)$$

The feature functions, $f_i$, are the number of times each rule is used in a derivation. $Z_\lambda(\mathbf{e})$ is the normalizing factor. The model parameters, $\lambda_i$, are trained using L-BFGS (Nocedal, 1980) to maximize the conditional log-likelihood of the training examples (with a Gaussian prior). The decoding task is thus to find a derivation $\mathbf{d}^\star$ that maximizes $\Pr_\lambda(\mathbf{d}^\star|\mathbf{e})$, and the output MR translation, $\mathbf{f}^\star$, is the yield of $\mathbf{d}^\star$. This can be done in cubic time with respect to the length of $\mathbf{e}$ using an Earley chart parser.

### 3.3 Generation by Inverting WASP

Now we show how to invert WASP to produce WASP$^{-1}$, and use it for NLG. We can use the same grammar for both parsing and generation, a particularly appealing aspect of using WASP. Since an SCFG is fully symmetric with respect to both generated strings, the same chart used for parsing can be easily adapted for efficient generation (Shieber, 1988; Kay, 1996).

Given an input MR, $\mathbf{f}$, WASP$^{-1}$ finds a sentence $\mathbf{e}$ that maximizes $\Pr(\mathbf{e}|\mathbf{f})$. It is difficult to directly model $\Pr(\mathbf{e}|\mathbf{f})$, however, because it has to assign low probabilities to output sentences that are not grammatical. There is no such requirement for parsing, because the use of the MRL grammar ensures the grammaticality of all output MRs. For generation, we need an NL grammar to ensure grammaticality, but this is not available *a priori*.

This motivates the *noisy-channel model* for WASP$^{-1}$, where $\Pr(\mathbf{e}|\mathbf{f})$ is divided into two smaller components:

$$\arg\max_{\mathbf{e}} \Pr(\mathbf{e}|\mathbf{f}) = \arg\max_{\mathbf{e}} \Pr(\mathbf{e}) \Pr(\mathbf{f}|\mathbf{e}) \qquad (2)$$

$\Pr(\mathbf{e})$ is the *language model*, and $\Pr(\mathbf{f}|\mathbf{e})$ is the *parsing model*. The generation task is to find a sentence $\mathbf{e}$ such that (1) $\mathbf{e}$ is a good sentence a priori, and (2) its meaning is the same as the input MR. For the language model, we use an $n$-gram model, which is remarkably useful in ranking candidate generated sentences (Knight and Hatzivassiloglou, 1995; Bangalore et al., 2000; Langkilde-Geary, 2002). For the parsing model, we re-use the one from WASP (Equation 1). Hence computing (2) means maximizing the following:

$$\begin{aligned} & \max_{\mathbf{e}} \Pr(\mathbf{e}) \Pr(\mathbf{f}|\mathbf{e}) \\ \approx\ & \max_{\mathbf{d} \in D(\mathbf{f})} \Pr(\mathbf{e}(\mathbf{d})) \Pr_\lambda(\mathbf{d}|\mathbf{e}(\mathbf{d})) \\ =\ & \max_{\mathbf{d} \in D(\mathbf{f})} \frac{\Pr(\mathbf{e}(\mathbf{d})) \cdot \exp \sum_i \lambda_i f_i(\mathbf{d})}{Z_\lambda(\mathbf{e}(\mathbf{d}))} \qquad (3) \end{aligned}$$

where $D(\mathbf{f})$ is the set of derivations that are consistent with $\mathbf{f}$, and $\mathbf{e}(\mathbf{d})$ is the output sentence that a derivation $\mathbf{d}$ yields. Compared to most existing work on generation, WASP$^{-1}$ has the following characteristics:

1. It does not require any lexical information in the input MR, so lexical selection is an integral part of the decoding algorithm.

2. Each predicate is translated to a phrase. Moreover, it need not be a contiguous phrase (consider the SCFG rule for `bowner` in Section 3.2).

For decoding, we use an Earley chart generator that scans the input MR from left to right. This implies that each chart item covers a certain substring of the input MR, not a subsequence in general. It requires the order in which MR predicates appear to be fixed, i.e. the order determines the meaning of the MR. Since the order need not be identical to the word order of the target NL, there is no need for the content planner to know the target NL grammar, which is learned from the training data.

Overall, the noisy-channel model is a weighted SCFG, obtained by intersecting the NL side of the WASP SCFG with the $n$-gram language model. The chart generator is very similar to the chart parser, except for the following:

1. To facilitate the calculation of $\Pr(\mathbf{e}(\mathbf{d}))$, chart items now include a list of $(n-1)$-grams that encode the context in which output NL phrases appear. The size of the list is $2N+2$, where $N$ is the number of non-terminals to be rewritten in the dotted rule.

2. Words are generated from word gaps through special rules $(g) \rightarrow \langle \alpha, \emptyset \rangle$, where the word gap, $(g)$, is treated as a non-terminal, and $\alpha$ is the NL string that fills the gap ($|\alpha| \leq g$). The empty set symbol indicates that the NL string does not carry any meaning. There are similar constructs in Carroll et al. (1999) that generate function words. Furthermore, to improve efficiency, our generator only considers gap fillers that have been observed during training.

3. The normalizing factor in (3), $Z_\lambda(\mathbf{e}(\mathbf{d}))$, is not a constant and varies across the output string, $\mathbf{e}(\mathbf{d})$. (Note that $Z_\lambda(\mathbf{e})$ is fixed for parsing.) This is unfortunate because the calculation of $Z_\lambda(\mathbf{e}(\mathbf{d}))$ is expensive, and it is not easy to incorporate it into the chart generation algorithm. Normalization is done as follows. First, compute the $k$-best candidate output strings based on the unnormalized version of (3), $\Pr(\mathbf{e}(\mathbf{d})) \cdot \exp \sum_i \lambda_i f_i(\mathbf{d})$. Then re-rank the list by normalizing the scores using $Z_\lambda(\mathbf{e}(\mathbf{d}))$, which is obtained by running the inside-outside algorithm on each output string. This results in a decoding algorithm that is approximate—the best output string might not be in the $k$-best list—and takes cubic time with respect to the length of *each* of the $k$ candidate output strings ($k = 100$ in our experiments).

Learning in WASP$^{-1}$ involves two steps. First, a back-off $n$-gram language model with Good-Turing discounting and no lexical classes[2] is built from all training sentences using the SRILM Toolkit (Stolcke, 2002). We use $n = 2$ since higher values seemed to cause overfitting in our domains. Next, the parsing model is trained as described in Section 3.2.

## 4 Improving the SMT-based Generators

The SMT-based generation algorithms, PHARAOH and WASP$^{-1}$, while reasonably effective, can be substantially improved by borrowing ideas from each other.

### 4.1 Improving the PHARAOH-based Generator

A major weakness of PHARAOH as an NLG system is its inability to exploit the formal structure of the MRL. Like WASP$^{-1}$, the phrase extraction algorithm of PHARAOH is based on the output of a word alignment model such as GIZA++ (Koehn et al., 2003), which performs poorly when applied directly to MRLs (Section 3.2).

We can improve the PHARAOH-based generator by supplying linearized parse-trees as input rather than flat MRs. As a result, the basic translation units are sequences of MRL productions, rather than sequences of MR tokens. This way PHARAOH can exploit the formal grammar of the MRL to produce high-quality phrase pairs. The same idea is used in WASP$^{-1}$ to produce high-quality SCFG rules. We call the resulting hybrid NLG system PHARAOH++.

### 4.2 Improving the WASP-based Generator

There are several aspects of PHARAOH that can be used to improve WASP$^{-1}$. First, the probabilistic model of WASP$^{-1}$ is less than ideal as it requires an extra re-ranking step for normalization, which is expensive and prone to over-pruning. To remedy this situation, we can borrow the probabilistic model of PHARAOH, and define the parsing model as:

$$\Pr(\mathbf{d}|\mathbf{e}(\mathbf{d})) = \prod_{d \in \mathbf{d}} w(r(d)) \qquad (4)$$

which is the product of the weights of the rules used in a derivation $\mathbf{d}$. The rule weight, $w(X \rightarrow \langle \alpha, \beta \rangle)$, is in turn defined as:

$$P(\beta|\alpha)^{\lambda_1} P(\alpha|\beta)^{\lambda_2} P_w(\beta|\alpha)^{\lambda_3} P_w(\alpha|\beta)^{\lambda_4} \exp(-|\alpha|)^{\lambda_5}$$

where $P(\beta|\alpha)$ and $P(\alpha|\beta)$ are the relative frequencies of $\beta$ and $\alpha$, and $P_w(\beta|\alpha)$ and $P_w(\alpha|\beta)$ are

[2] This is to ensure that the same language model is used in all systems that we tested.

the lexical weights (Koehn et al., 2003). The word penalty, $\exp(-|\alpha|)$, allows some control over the output sentence length. Together with the language model, the new formulation of $\Pr(\mathbf{e}|\mathbf{f})$ is a log-linear model with $\lambda_i$ as parameters. The advantage of this model is that maximization requires no normalization and can be done exactly and efficiently. The model parameters are trained using minimum error-rate training (Och, 2003).

Following the phrase extraction phase in PHARAOH, we eliminate word gaps by incorporating unaligned words as part of the extracted NL phrases (Koehn et al., 2003). The reason is that while word gaps are useful in dealing with unknown phrases during semantic parsing, for generation, using *known* phrases generally leads to better fluency. For the same reason, we also allow the extraction of longer phrases that correspond to multiple predicates (but no more than 5).

We call the resulting hybrid system WASP$^{-1}$++. It is similar to the syntax-based SMT system of Chiang (2005), which uses both SCFG and PHARAOH's probabilistic model. The main difference is that we use the MRL grammar to constrain rule extraction, so that significantly fewer rules are extracted, making it possible to do exact inference.

## 5 Experiments

We evaluated all four SMT-based NLG systems introduced in this paper: PHARAOH, WASP$^{-1}$, and the hybrid systems, PHARAOH++ and WASP$^{-1}$++.

We used the ROBOCUP and GEOQUERY corpora in our experiments. The ROBOCUP corpus consists of 300 pieces of coach advice taken from the log files of the 2003 ROBOCUP Coach Competition. The advice was written in CLANG and manually translated to English (Kuhlmann et al., 2004). The average MR length is 29.47 tokens, or 12.82 nodes for linearized parse-trees. The average sentence length is 22.52. The GEOQUERY corpus consists of 880 English questions gathered from various sources. The questions were manually translated to the functional GEOQUERY language (Kate et al., 2005). The average MR length is 17.55 tokens, or 5.55 nodes for linearized parse-trees. The average sentence length is 7.57.

Reference: *If our player 2, 3, 7 or 5 has the ball and the ball is close to our goal line ...*
PHARAOH++: *If player 3 has the ball is in 2 5 the ball is in the area near our goal line ...*
WASP$^{-1}$++: *If players 2, 3, 7 and 5 has the ball and the ball is near our goal line ...*

Figure 4: Sample partial system output in the ROBOCUP domain

| | ROBOCUP | | GEOQUERY | |
|---|---|---|---|---|
| | BLEU | NIST | BLEU | NIST |
| PHARAOH | 0.3247 | 5.0263 | 0.2070 | 3.1478 |
| WASP$^{-1}$ | 0.4357 | 5.4486 | 0.4582 | 5.9900 |
| PHARAOH++ | 0.4336 | 5.9185 | **0.5354** | 6.3637 |
| WASP$^{-1}$++ | **0.6022** | **6.8976** | **0.5370** | **6.4808** |

Table 1: Results of automatic evaluation; **bold** type indicates the best performing system (or systems) for a given domain-metric pair ($p < 0.05$)

### 5.1 Automatic Evaluation

We performed 4 runs of 10-fold cross validation, and measured the performance of the learned generators using the BLEU score (Papineni et al., 2002) and the NIST score (Doddington, 2002). Both MT metrics measure the precision of a translation in terms of the proportion of $n$-grams that it shares with the reference translations, with the NIST score focusing more on $n$-grams that are less frequent and more informative. Both metrics have recently been used to evaluate generators (Langkilde-Geary, 2002; Nakanishi et al., 2005; Belz and Reiter, 2006).

All systems were able to generate sentences for more than 97% of the input. Figure 4 shows some sample output of the systems. Table 1 shows the automatic evaluation results. Paired $t$-tests were used to measure statistical significance. A few observations can be made. First, WASP$^{-1}$ produced a more accurate generator than PHARAOH. Second, PHARAOH++ significantly outperformed PHARAOH, showing the importance of exploiting the formal structure of the MRL. Third, WASP$^{-1}$++ significantly outperformed WASP$^{-1}$. Most of the gain came from PHARAOH's probabilistic model. Decoding was also 4–11 times faster, despite exact inference and a larger grammar due to extraction of longer phrases. Lastly, WASP$^{-1}$++ significantly outperformed PHARAOH++ in the ROBOCUP

| | ROBOCUP | | GEOQUERY | |
|---|---|---|---|---|
| | *Flu.* | *Ade.* | *Flu.* | *Ade.* |
| PHARAOH++ | 2.5 | 2.9 | **4.3** | **4.7** |
| WASP$^{-1}$++ | **3.6** | **4.0** | **4.1** | **4.7** |

Table 2: Results of human evaluation

domain. This is because WASP$^{-1}$++ allows discontiguous NL phrases and PHARAOH++ does not. Such phrases are commonly used in ROBOCUP for constructions like: **players** *2* **,** *3* **,** *7* **and** *5*; 26.96% of the phrases generated during testing were discontiguous. When faced with such predicates, PHARAOH++ would consistently omit some of the words: e.g. **players** *2 3 7 5*, or not learn any phrases for those predicates at all. On the other hand, only 4.47% of the phrases generated during testing for GEOQUERY were discontiguous, so the advantage of WASP$^{-1}$++ over PHARAOH++ was not as obvious.

Our BLEU scores are not as high as those reported in Langkilde-Geary (2002) and Nakanishi et al. (2005), which are around 0.7–0.9. However, their work involves the regeneration of automatically parsed text, and the MRs that they use, which are essentially dependency parses, contain extensive lexical information of the target NL.

### 5.2 Human Evaluation

Automatic evaluation is only an imperfect substitute for human assessment. While it is found that BLEU and NIST correlate quite well with human judgments in evaluating NLG systems (Belz and Reiter, 2006), it is best to support these figures with human evaluation, which we did on a small scale. We recruited 4 native speakers of English with no previous experience with the ROBOCUP and GEOQUERY domains. Each subject was given the same 20 sentences for each domain, randomly chosen from the test sets. For each sentence, the subjects were asked to judge the output of PHARAOH++ and WASP$^{-1}$++ in terms of fluency and adequacy. They were presented with the following definition, adapted from Koehn and Monz (2006):

| *Score* | *Fluency* | *Adequacy* |
|---|---|---|
| 5 | Flawless English | All meaning |
| 4 | Good English | Most meaning |
| 3 | Non-native English | Some meaning |
| 2 | Disfluent English | Little meaning |
| 1 | Incomprehensible | No meaning |

For each generated sentence, we computed the average of the 4 human judges' scores. No score normalization was performed. Then we compared the two systems using a paired $t$-test. Table 2 shows that WASP$^{-1}$++ produced better generators than PHARAOH++ in the ROBOCUP domain, consistent with the results of automatic evaluation.

| | PHARAOH++ | | WASP$^{-1}$++ | |
|---|---|---|---|---|
| | BLEU | NIST | BLEU | NIST |
| *English* | 0.5344 | 5.3289 | **0.6035** | **5.7133** |
| *Spanish* | **0.6042** | **5.6321** | **0.6175** | **5.7293** |
| *Japanese* | 0.6171 | **4.5357** | **0.6585** | **4.6648** |
| *Turkish* | 0.4562 | **4.2220** | **0.4824** | **4.3283** |

Table 3: Results of automatic evaluation on the multilingual GEOQUERY data set

### 5.3 Multilingual Experiments

Lastly, we describe our experiments on the multilingual GEOQUERY data set. The 250-example data set is a subset of the larger GEOQUERY corpus. All English questions in this data set were manually translated into Spanish, Japanese and Turkish, while the corresponding MRs remain unchanged. Table 3 shows the results, which are similar to previous results on the larger GEOQUERY corpus. WASP$^{-1}$++ outperformed PHARAOH++ for some language-metric pairs, but otherwise performed comparably.

## 6 Related Work

Numerous efforts have been made to unify the tasks of semantic parsing and tactical generation. One of the earliest espousals of the notion of grammar reversability can be found in Kay (1975). Shieber (1988) further noted that not only a single grammar can be used for parsing and generation, but the same language-processing architecture can be used for both tasks. Kay (1996) identified parsing charts as such an architecture, which led to the development of various chart generation algorithms: Carroll et al. (1999) for HPSG, Bangalore et al. (2000) for LTAG, Moore (2002) for unification grammars, White and Baldridge (2003) for CCG. More recently, statistical chart generators have emerged, including White (2004) for CCG, Carroll and Oepen (2005) and Nakanishi et al. (2005) for HPSG. Many of these systems, however, focus on the task of *surface realization*—inflecting and ordering words—which ignores the problem of lexical selection. In contrast, our SMT-based methods integrate lexical selection and realization in an elegant framework and automatically learn all of their linguistic knowledge from an annotated corpus.

## 7 Conclusion

We have presented four tactical generation systems based on various SMT-based methods. In particular, the hybrid system produced by inverting the WASP semantic parser shows the best overall results across different application domains.

## Acknowledgments

We would like to thank Kevin Knight, Jason Baldridge, Razvan Bunescu, and the anonymous reviewers for their valuable comments. We also sincerely thank the four annotators who helped us evaluate the SMT-based generators. This research was supported by DARPA under grant HR0011-04-1-0007 and a gift from Google Inc.

## References

A. V. Aho and J. D. Ullman. 1972. *The Theory of Parsing, Translation, and Compiling*. Prentice Hall, Englewood Cliffs, NJ.

S. Bangalore, O. Rambow, and S. Whittaker. 2000. Evaluation metrics for generation. In *Proc. INLG-00*, pages 1–8, Mitzpe Ramon, Israel, July.

A. Belz and E. Reiter. 2006. Comparing automatic and human evaluation of NLG systems. In *Proc. EACL-06*, pages 313–320, Trento, Italy, April.

P. F. Brown, V. J. Della Pietra, S. A. Della Pietra, and R. L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–312, June.

J. Carroll and S. Oepen. 2005. High efficiency realization for a wide-coverage unification grammar. In *Proc. IJCNLP-05*, pages 165–176, Jeju Island, Korea, October.

J. Carroll, A. Copestake, D. Flickinger, and V. Poznański. 1999. An efficient chart generator for (semi-)lexicalist grammars. In *Proc. EWNLG-99*, pages 86–95, Toulouse, France.

M. Chen et al. 2003. Users manual: RoboCup soccer server manual for soccer server version 7.07 and later. Available at `http://sourceforge.net/projects/sserver/`.

D. Chiang. 2005. A hierarchical phrase-based model for statistical machine translation. In *Proc. ACL-05*, pages 263–270, Ann Arbor, MI, June.

G. Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proc. ARPA Workshop on Human Language Technology*, pages 128–132, San Diego, CA.

U. Germann. 2003. Greedy decoding for statistical machine translation in almost linear time. In *Proc. HLT/NAACL-03*, Edmonton, Canada.

P. S. Jacobs. 1985. PHRED: A generator for natural language interfaces. *Computational Linguistics*, 11(4):219–242.

R. J. Kate, Y. W. Wong, and R. J. Mooney. 2005. Learning to transform natural to formal languages. In *Proc. AAAI-05*, pages 1062–1068, Pittsburgh, PA, July.

M. Kay. 1975. Syntactic processing and functional sentence perspective. In *Theoretical Issues in Natural Language Processing—Supplement to the Proceedings*, pages 12–15, Cambridge, MA, June.

M. Kay. 1996. Chart generation. In *Proc. ACL-96*, pages 200–204, San Francisco, CA.

K. Knight and V. Hatzivassiloglou. 1995. Two-level, many-paths generation. In *Proc. ACL-95*, pages 252–260, Cambridge, MA.

P. Koehn and C. Monz. 2006. Manual and automatic evaluation of machine translation between European languages. In *Proc. SMT-06 Workshop*, pages 102–121, New York City, NY, June.

P. Koehn, F. J. Och, and D. Marcu. 2003. Statistical phrase-based translation. In *Proc. HLT/NAACL-03*, Edmonton, Canada.

G. Kuhlmann, P. Stone, R. J. Mooney, and J. W. Shavlik. 2004. Guiding a reinforcement learner with natural language advice: Initial results in RoboCup soccer. In *Proc. of the AAAI-04 Workshop on Supervisory Control of Learning and Adaptive Systems*, San Jose, CA, July.

I. Langkilde-Geary. 2002. An empirical verification of coverage and correctness for a general-purpose sentence generator. In *Proc. INLG-02*, pages 17–24, Harriman, NY, July.

R. C. Moore. 2002. A complete, efficient sentence-realization algorithm for unification grammar. In *Proc. INLG-02*, pages 41–48, Harriman, NY, July.

H. Nakanishi, Y. Miyao, and J. Tsujii. 2005. Probabilistic models for disambiguation of an HPSG-based chart generator. In *Proc. IWPT-05*, pages 93–102, Vancouver, Canada, October.

J. Nocedal. 1980. Updating quasi-Newton matrices with limited storage. *Mathematics of Computation*, 35(151):773–782, July.

F. J. Och and H. Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

F. J. Och. 2003. Minimum error rate training in statistical machine translation. In *Proc. ACL-03*, pages 160–167, Sapporo, Japan, July.

K. Papineni, S. Roukos, T. Ward, and W. J. Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proc. ACL-02*, pages 311–318, Philadelphia, PA, July.

S. M. Shieber. 1988. A uniform architecture for parsing and generation. In *Proc. COLING-88*, pages 614–619, Budapest, Hungary.

S. M. Shieber. 1993. The problem of logical-form equivalence. *Computational Linguistics*, 19(1):179–190.

A. Stolcke. 2002. SRILM—an extensible language modeling toolkit. In *Proc. ICSLP-02*, pages 901–904, Denver, CO.

M. White and J. Baldridge. 2003. Adapting chart realization to CCG. In *Proc. EWNLG-03*, Budapest, Hungary, April.

M. White. 2004. Reining in CCG chart realization. In *Proc. INLG-04*, New Forest, UK, July.

P. Whitelock. 1992. Shake-and-bake translation. In *Proc. COLING-92*, pages 784–791, Nantes, France.

Y. W. Wong and R. J. Mooney. 2006. Learning for semantic parsing with statistical machine translation. In *Proc. HLT/NAACL-06*, pages 439–446, New York City, NY, June.