In *Proceedings of the 39th Meeting of the Association for Computational Linguistics (ACL'01)*, Toulouse, France.

Evaluating a Trainable Sentence Planner for a Spoken Dialogue System

Owen Rambow
AT&T Labs – Research
Florham Park, NJ, USA
rambow@research.att.com

Monica Rogati
Carnegie Mellon University
Pittsburgh, PA, USA
mrogati+@cs.cmu.edu

Marilyn A. Walker
AT&T Labs – Research
Florham Park, NJ, USA
walker@research.att.com

Abstract

Techniques for automatically training modules of a natural language generator have recently been proposed, but a fundamental concern is whether the quality of utterances produced with trainable components can compete with hand-crafted template-based or rule-based approaches. In this paper We experimentally evaluate a trainable sentence planner for a spoken dialogue system by eliciting subjective human judgments. In order to perform an exhaustive comparison, we also evaluate a hand-crafted template-based generation component, two rule-based sentence planners, and two baseline sentence planners. We show that the trainable sentence planner performs better than the rule-based systems and the baselines, and as well as the hand-crafted system.

1 Introduction

The past several years have seen a large increase in commercial dialog systems. These systems typically use system-initiative dialog strategies, with system utterances highly scripted for style and register and recorded by voice talent. However several factors argue against the continued use of these simple techniques for producing the system side of the conversation. First, text-to-speech has improved to the point of being a viable alternative to pre-recorded prompts. Second, there is a perceived need for spoken dialog systems to be more flexible and support user initiative, but this requires greater flexibility in utterance generation. Finally, systems to support complex planning are being developed, which will require more sophisticated output.

As we move away from systems with pre-recorded prompts, there are two possible approaches to producing system utterances. The first is **template-based** generation, where utterances are produced from hand-crafted string templates. Most current research systems use template-based generation because it is conceptually straightforward. However, while little or no linguistic training is needed to write templates, it is a tedious and time-consuming task: one or more templates must be written for each combination of goals and discourse contexts, and linguistic issues such as subject-verb agreement and determiner-noun agreement must be repeatedly encoded for each template. Furthermore, maintenance of the collection of templates becomes a software engineering problem as the complexity of the dialog system increases.[1]

The second approach is **natural language generation** (NLG), which customarily divides the generation process into three modules (Rambow and Korelsky, 1992): (1) Text Planning, (2) Sentence Planning, and (3) Surface Realization. In this paper, we discuss only sentence planning; the role of the sentence planner is to choose abstract lexico-structural resources for a text plan, where a text plan encodes the communicative goals for an utterance (and, sometimes, their rhetorical structure). In general, NLG promises portability across application domains and dialog situations by focusing on the development of rules for each generation module that are general and domain-

[1] Although we are not aware of any software engineering studies of template development and maintenance, this claim is supported by abundant anecdotal evidence.

independent. However, the quality of the output for a particular domain, or a particular situation in a dialog, may be inferior to that of a template-based system without considerable investment in domain-specific rules or domain-tuning of general rules. Furthermore, since rule-based systems use sophisticated linguistic representations, this handcrafting requires linguistic knowledge.

Recently, several approaches for *automatically training* modules of an NLG system have been proposed (Langkilde and Knight, 1998; Mellish et al., 1998; Walker, 2000). These hold the promise that the complex step of customizing NLG systems by hand can be automated, while avoiding the need for tedious hand-crafting of templates. While the engineering benefits of trainable approaches appear obvious, it is unclear whether the utterance quality is high enough.

In (Walker et al., 2001) we propose a new model of sentence planning called SPOT. In SPOT, the sentence planner is automatically trained, using feedback from two human judges, to choose the best from among different options for realizing a set of communicative goals. In (Walker et al., 2001), we evaluate the performance of the learning component of SPOT, and show that SPOT learns to select sentence plans that are highly rated by the two human judges. While this evaluation shows that SPOT has indeed learned from the human judges, it does not show that using only two human judgments is sufficient to produce more broadly acceptable results, nor does it show that SPOT performs as well as optimized hand-crafted template or rule-based systems. In this paper we address these questions. Because SPOT is trained on data from a working system, we can directly compare SPOT to the hand-crafted, template-based generation component of the current system. In order to perform an exhaustive comparison, we also implemented two rule-based and two baseline sentence-planners. One baseline simply produces a single sentence for each communicative goal. Another baseline randomly makes decisions about how to combine communicative goals into sentences. We directly compare these different approaches in an evaluation experiment in which 60 human subjects rate each system's output on a scale of 1 to 5.

The experimental design is described in section 2. The sentence planners used in the evaluation are described in section 3. In section 4, we present our results. We show that the trainable sentence planner performs better than both rule-based systems and as well as the hand-crafted template-based system. These four systems outperform the baseline sentence planners. Section 5 summarizes our results and discusses related and future work.

System1:	Welcome.... What airport would you like to fly out of?
User2:	I need to go to Dallas.
System3:	Flying to Dallas. What departure airport was that?
User4:	from Newark on September the 1st.
System5:	Flying from Newark to Dallas, Leaving on the 1st of September, And what time did you want to leave?

Figure 1: A dialog with AMELIA

2 Experimental Context and Design

Our research concerns developing and evaluating a portable generation component for a mixed-initiative travel planning system, AMELIA, developed at AT&T Labs as part of DARPA Communicator. Consider the required generation capabilities of AMELIA, as illustrated in Figure 1.

Utterance System1 requests information about the caller's departure airport, but in User2, the caller takes the initiative to provide information about her destination. In System3, the system's goal is to implicitly confirm the destination (because of the possibility of error in the speech recognition component), and request information (for the second time) of the caller's departure airport. This combination of communicative goals arises dynamically in the dialog because the system supports user initiative, and requires different capabilities for generation than if the system could only understand the direct answer to the question that it asked in System1.

In User4, the caller provides this information but takes the initiative to provide the month and day of travel. Given the system's dialog strategy, the communicative goals for its next turn are to implicitly confirm all the information that the user has provided so far, i.e. the departure and destination cities and the month and day information, as well as to request information about the time of travel. The system's representation of its communicative goals for System5 is in Figure 2. As before, this combination of communicative goals arises in response to the user's initiative.

implicit-confirm(orig-city:NEWARK)
implicit-confirm(dest-city:DALLAS)
implicit-confirm(month:9)
implicit-confirm(day-number:1)
request(depart-time:whatever)

Figure 2: The text plan (communicative goals) for System5 in Figure 1

Like most working research spoken dialog systems, AMELIA uses hand-crafted, template-based generation. Its output is created by choosing string templates for each elementary speech act, using a large choice function which depends on the type of speech act and various context conditions. Values of template variables (such as origin and destination cities) are instantiated by the dialog manager. The string templates for all the speech acts of a turn are heuristically ordered and then appended to produce the output. In order to produce output that is not highly redundant, string templates must be written for every possible combination of speech acts in a text plan. We refer to the output generated by AMELIA using this approach as the TEMPLATE output.

System	Realization
TEMPLATE	Flying from Newark to Dallas, Leaving on the 1st of September, And what time did you want to leave?
SPoT	What time would you like to travel on September the 1st to Dallas from Newark?
RBS (Rule-Based)	What time would you like to travel on September the 1st to Dallas from Newark?
ICF (Rule-Based)	What time would you like to fly on September the 1st to Dallas from Newark?
RANDOM	Leaving in September. Leaving on the 1st. What time would you, traveling from Newark to Dallas, like to leave?
NOAGG	Leaving on the 1. Leaving in September. Going to Dallas. Leaving from Newark. What time would you like to leave?

Figure 3: Sample outputs for System5 of Figure 1 for each type of generation system used in the evaluation experiment.

We perform an evaluation using human subjects who judged the TEMPLATE output of AMELIA against five NLG-based approaches: SPOT, two rule-based approaches, and two baselines. We describe them in Section 3. An example output for the text plan in Figure 2 for each system is in Figure 3. The experiment required human subjects to read 5 dialogs of real interactions with AMELIA. At 20 points over the 5 dialogs, AMELIA's actual utterance (TEMPLATE) is augmented with a set of variants; each set of variants included a representative generated by SPOT, and representatives of the four comparison sentence planners. At times two or more of these variants coincided, in which case sentences were not repeated and fewer than six sentences were presented to the subjects. The subjects rated each variation on a 5-point Likert scale, by stating the degree to which they agreed with the statement *The system's utterance is easy to understand, well-formed, and appropriate to the dialog context*. Sixty colleagues not involved in this research completed the experiment.

3 Sentence Planning Systems

This section describes the five sentence planners that we compare. SPOT, the two rule-based systems, and the two baseline sentence planners are all NLG based sentence planners. In Section 3.1, we describe the shared representations of the NLG based sentence planners. Section 3.2 describes the baselines, RANDOM and NOAGG. Section 3.3 describes SPOT. Section 3.4 describes the rule-based sentence planners, RBS and ICF.

3.1 Aggregation in Sentence Planning

In all of the NLG sentence planners, each speech act is assigned a canonical lexico-structural representation (called a **DSyntS** – Deep Syntactic Structure (Melčuk, 1988)). We exclude issues of lexical choice from this study, and restrict our attention to the question of how elementary structures for separate elementary speech acts are assembled into extended discourse. The basis of all the NLG systems is a set of clause-combining operations that incrementally transform a list of elementary predicate-argument representations (the DSyntSs corresponding to the elementary speech acts of a single text plan) into a list of lexico-structural representations of one or more sentences, that are sent to a surface realizer. We utilize the RealPro Surface realizer with all of the

Rule	Arg 1	Arg 2	Result
MERGE	You are leaving from Newark.	You are leaving at 5	You are leaving at 5 from Newark
SOFT-MERGE	You are leaving from Newark	You are going to Dallas	You are traveling from Newark to Dallas
CONJUNCTION	You are leaving from Newark.	You are going to Dallas.	You are leaving from Newark and you are going to Dallas.
RELATIVE-CLAUSE	Your flight leaves at 5.	Your flight arrives at 9.	Your flight, which leaves at 5, arrives at 9.
ADJECTIVE	Your flight leaves at 5.	Your flight is nonstop.	Your nonstop flight leaves at 5.
PERIOD	You are leaving from Newark.	You are going to Dallas.	You are leaving from Newark. You are going to Dallas
RANDOM CUE-WORD	What time would yo like to leave?	n/a	Now, what time would you like to leave?

Figure 4: List of clause combining operations with examples

sentence planners (Lavoie and Rambow, 1997).

DSyntSs are combined using the operations exemplified in Figure 4. The result of applying the operations is a **sentence plan tree** (or **sp-tree** for short), which is a binary tree with leaves labeled by all the elementary speech acts from the input text plan, and with its interior nodes labeled with clause-combining operations. As an example, Figure 5 shows the sp-tree for utterance System5 in Figure 1. Node **soft-merge-general**$_3$ merges an implicit-confirmation of the destination city and the origin city. The row labelled SOFT-MERGE in Figure 4 shows the result when Args 1 and 2 are implicit confirmations of the origin and destination. See (Walker et al., 2001) for more detail on the sp-tree. The experimental sentence planners described below vary how the sp-tree is constructed.

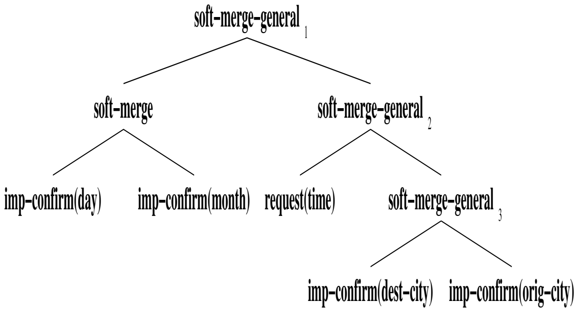

Figure 5: A Sentence Plan Tree for Utterance System 5 in Dialog D1

3.2 Baseline Sentence Planners

In one obvious baseline system the sp-tree is constructed by applying only the PERIOD operation: each elementary speech act is realized as its own sentence. This baseline, NOAGG, was suggested by Hovy and Wanner (1996). For NOAGG, we order the communicative acts from the text plan as follows: implicit confirms precede explicit confirms precede requests. Figure 3 includes a NOAGG output for the text plan in Figure 2.

A second possible baseline sentence planner simply applies combination rules randomly according to a hand-crafted probability distribution based on preferences for operations such as the MERGE family over CONJUNCTION and PERIOD. In order to be able to generate the resulting sentence plan tree, we exclude certain combinations, such as generating anything other than a PERIOD above a node labeled PERIOD in a sentence plan. The resulting sentence planner we refer to as RANDOM. Figure 3 includes a RANDOM output for the text plan in Figure 2.

In order to construct a more complex, and hopefully better, sentence planner, we need to encode constraints on the application of, and ordering of, the operations. It is here that the remaining approaches differ. In the first approach, SPOT, we learn constraints from training material; in the second approach, **rule-based**, we construct constraints by hand.

3.3 SPoT: A Trainable Sentence Planner

For the sentence planner SPOT, we reconceptualize sentence planning as consisting of two distinct phases as in Figure 6. In the first phase, the sentence-plan-generator (**SPG**) randomly generates up to twenty possible sentence plans for a given text-plan input. For this phase we use the RANDOM sentence-planner. In the second phase, the sentence-plan-ranker (SPR) ranks the sample

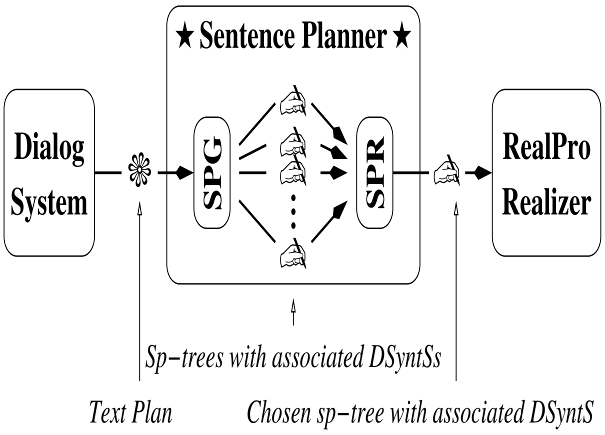

Figure 6: Architecture of SPoT

sentence plans, and then selects the top-ranked output to input to the surface realizer. The SPR is automatically trained by applying RankBoost (Freund et al., 1998) to learn ranking rules from training data. The training data was assembled by using RANDOM to randomly generate up to 20 realizations for 100 turns; two human judges then ranked each of these realizations (using the setup described in Section 2). Over 3,000 features were discovered from the generated trees by routines that encode structural and lexical aspects of the sp-trees and the DSyntS. RankBoost identified the features that contribute most to a realization's ranking. The SPR uses these rules to rank alternative sp-trees, and then selects the top-ranked output as input to the surface realizer. Walker et al. (2001) describe SPoT in detail.

3.4 Two Rule-Based Sentence Planners

It has not been the object of our research to construct a rule-based sentence planner by hand, be it domain-independent or optimized for our domain. Our goal was to compare the SPoT sentence planner with a representative rule-based system. We decided against using an existing off-the-shelf rule-based system, since it would be too complex a task to port it to our application. Instead, we constructed two reasonably representative rule-based sentence planners. This task was made easier by the fact that we could reuse much of the work done for SPoT, in particular the data structure of the sp-tree and the implementation of the clause-combining operations. We developed the two systems by applying heuristics for producing good output, such as preferences for aggregation. They differ only in the initial ordering of the communicative acts in the input text plan.

In the first rule-based system, RBS (for "Rule-Based System"), we order the speech acts with explicit confirms first, then requests, then implicit confirms. Note that explicit confirms and requests do not co-occur in our data set. The second rule-based system is identical, except that implicit confirms come first rather than last. This system we call ICF (for "Rule-based System with Implicit Confirms First").

In the initial step of both RBS and ICF, we take the two leftmost members of the text plan and try to combine them using the following preference ranking of the combination operations: ADJECTIVE, the MERGEs, CONJUNCTION, RELATIVE-CLAUSE, PERIOD. The first operation to succeed is chosen. This yields a binary sp-tree with three nodes, which becomes the **current sp-tree**. As long as the root node of the current sp-tree is not a PERIOD, we iterate through the list of remaining speech acts on the ordered text plan, combining each one with the current sp-tree using the preference-ranked operations as just described. The result of each iteration step is a binary, left-branching sp-tree. However, if the root node of the current sp-tree is a PERIOD, we start a new current sp-tree, as in the initial step described above. When the text plan has been exhausted, all partial sp-trees (all of which except for the last one are rooted in PERIOD) are combined in a left-branching tree using PERIOD. Cue words are added as follows: (1) The cue word *now* is attached to utterances beginning a new subtask; (2) The cue word *and* is attached to utterances continuing a subtask; (3) The cue words *alright* or *okay* are attached to utterances containing implicit confirmations. Figure 3 includes an RBS and an ICF output for the text plan in Figure 2. In this case ICF and RBS differ only in the verb chosen as a more general verb during the SOFT-MERGE operation.

We illustrate the RBS procedure with an example for which ICF works similarly. For RBS, the text plan in Figure 2 is ordered so that the request is first. For the request, a DSyntS is chosen that can be paraphrased as *What time would you like to leave?*. Then, the first implicit-confirm is translated by lookup into a DSyntS which on its own could generate *Leaving in September*. We first try the ADJECTIVE aggregation operation, but since neither tree is a predicative adjective, this fails. We then try the MERGE family. MERGE-GENERAL succeeds, since the tree for the request has an embedded node labeled *leave*. The resulting DSyntS can be paraphrased as *What time would you like to leave in September?*, and is attached to the new root node of the resulting sp-tree. The root node is labeled MERGE-GENERAL, and its two daughters are the two speech acts. The implicit-confirm of the day is added in a similar manner (adding another left-branching node to the sp-tree), yielding a DSyntS that can be paraphrased as *What time would you like to leave on September the 1st?* (using some special-case attachment for dates within MERGE). We now try and add the DSyntS for the implicit-confirm, whose DSyntS might generate *Going to Dallas*. Here, we again cannot use ADJECTIVE, nor can we use MERGE or MERGE-GENERAL, since the verbs are not identical. Instead, we use SOFT-MERGE-GENERAL, which identifies the *leave* node with the *go* root node of the DSyntS of the implicit-confirm. When soft-merging *leave* with *go*, *fly* is chosen as a generalization, resulting in a DSyntS that can be generated as *What time would you like to fly on September the 1st to Dallas?*. The sp-tree has added a layer but is still left-branching. Finally, the last implicit-confirm is added to yield a DSyntS that is realized as *What time would you like to fly on September the 1st to Dallas from Newark?*.

4 Experimental Results

All 60 subjects completed the experiment in a half hour or less. The experiment resulted in a total of 1200 judgements for each of the systems being compared, since each subject judged 20 utterances by each system. We first discuss overall differences among the different systems and then make comparisons among the four different types of systems: (1) TEMPLATE, (2) SPOT, (3) two rule-based systems, and (4) two baseline systems. All statistically significant results discussed here had p values of less than .01.

We first examined whether differences in human ratings (score) were predictable from the type of system that produced the utterance being rated. A one-way ANOVA with system as the independent variable and score as the dependent variable showed that there were significant differences in score as a function of system. The overall differences are summarized in Figure 7.

System	Min	Max	Mean	S.D.
TEMPLATE	1	5	3.9	1.1
SPoT	1	5	3.9	1.3
RBS	1	5	3.4	1.4
ICF	1	5	3.5	1.4
No Aggregation	1	5	3.0	1.2
Random	1	5	2.7	1.4

Figure 7: Summary of Overall Results for all Systems Evaluated

As Figure 7 indicates, some system outputs received more consistent scores than others, e.g. the standard deviation for TEMPLATE was much smaller than RANDOM. The ranking of the systems by average score is TEMPLATE, SPOT, ICF, RBS, NOAGG, and RANDOM. Posthoc comparisons of the scores of individual pairs of systems using the adjusted Bonferroni statistic revealed several different groupings.[2]

The highest ranking systems were TEMPLATE and SPOT, whose ratings were not statistically significantly different from one another. This shows that it is possible to match the quality of a hand-crafted system with a trainable one, which should be more portable, more general and require less overall engineering effort.

The next group of systems were the two rule-based systems, ICF and RBS, which were not statistically different from one another. However SPOT was statistically better than both of these systems ($p < .01$). Figure 8 shows that SPOT got more high rankings than either of the rule-based systems. In a sense this may not be that surprising, because as Hovy and Wanner (1996) point out, it is difficult to construct a rule-based sentence planner that handles all the rule interactions in a reasonable way. Features that SPoT's SPR uses allow SPOT to be sensitive to particular discourse configurations or lexical collocations. In order to encode these in a rule-based sentence

[2]The adjusted Bonferroni statistic guards against accidentally finding differences between systems when making multiple comparisons among systems.

planner, one would first have to discover these constraints and then determine a way of enforcing them. However the SPR simply learns that a particular configuration is less preferred, resulting in a small decrement in ranking for the corresponding sp-tree. This flexibility of incrementing or decrementing a particular sp-tree by a small amount may in the end allow it to be more sensitive to small distinctions than a rule-based system.

Along with the TEMPLATE and RULE-BASED systems, SPoT also scored better than the baseline systems NoAGG and RANDOM. This is also somewhat to be expected, since the baseline systems were intended to be the simplest systems constructable. However it would have been a possible outcome for SPoT to not be different than either system, e.g. if the sp-trees produced by RANDOM were all equally good, or if the aggregation rules that SPoT learned produced output less readable than NoAGG. Figure 8 shows that the distributions of scores for SPoT vs. the baseline systems are very different, with SPoT skewed towards higher scores.

Interestingly NoAGG also scored better than RANDOM ($p < .01$), and the standard deviation of its scores was smaller (see Figure 7). Remember that RANDOM's sp-trees often resulted in arbitrarily ordering the speech acts in the output. While NoAGG produced redundant utterances, it placed the initiative taking speech act at the end of the utterance in its most natural position, possibly resulting in a preference for NoAGG over RANDOM. Another reason to prefer NoAGG could be its predictability.

5 Discussion and Future Work

Other work has also explored automatically training modules of a generator (Langkilde and Knight, 1998; Mellish et al., 1998; Walker, 2000). However, to our knowledge, this is the first reported experimental comparison of a trainable technique that shows that the quality of system utterances produced with trainable components can compete with hand-crafted or rule-based techniques. The results validate our methodology; SPoT outperforms two representative rule-based sentence planners, and performs as well as the hand-crafted TEMPLATE system, but is more easily and quickly tuned to a new domain: the training materials for the SPoT sentence planner can be collected from subjective judgements from a small number of judges with little or no linguistic knowledge.

Previous work on evaluation of natural language generation has utilized three different approaches to evaluation (Mellish and Dale, 1998). The first approach is a subjective evaluation methodology such as we use here, where human subjects rate NLG outputs produced by different sources (Lester and Porter, 1997). Other work has evaluated template-based spoken dialog generation with a task-based approach, i.e. the generator is evaluated with a metric such as task completion or user satisfaction after dialog completion (Walker, 2000). This approach can work well when the task only involves one or two exchanges, when the choices have large effects over the whole dialog, or the choices vary the content of the utterance. Because sentence planning choices realize the same content and only affect the current utterance, we believed it important to get local feedback. A final approach focuses on subproblems of natural language generation such as the generation of referring expressions. For this type of problem it is possible to evaluate the generator by the degree to which it matches human performance (Yeh and Mellish, 1997). When evaluating sentence planning, this approach doesn't make sense because many different realizations may be equally good.

However, this experiment did not show that trainable sentence planners produce, *in general*, better-quality output than template-based or rule-based sentence planners. That would be impossible: given the nature of template and rule-based systems, any quality standard for the output can be met given sufficient person-hours, elapsed time, and software engineering acumen. Our principal goal, rather, is to show that the quality of the TEMPLATE output, for a currently operational dialog system whose template-based output component was developed, expanded, and refined over about 18 months, can be achieved using a trainable system, for which the necessary training data was collected in three person-days. Furthermore, we wished to show that a representative rule-based system based on current literature, without massive domain-tuning, cannot achieve the same

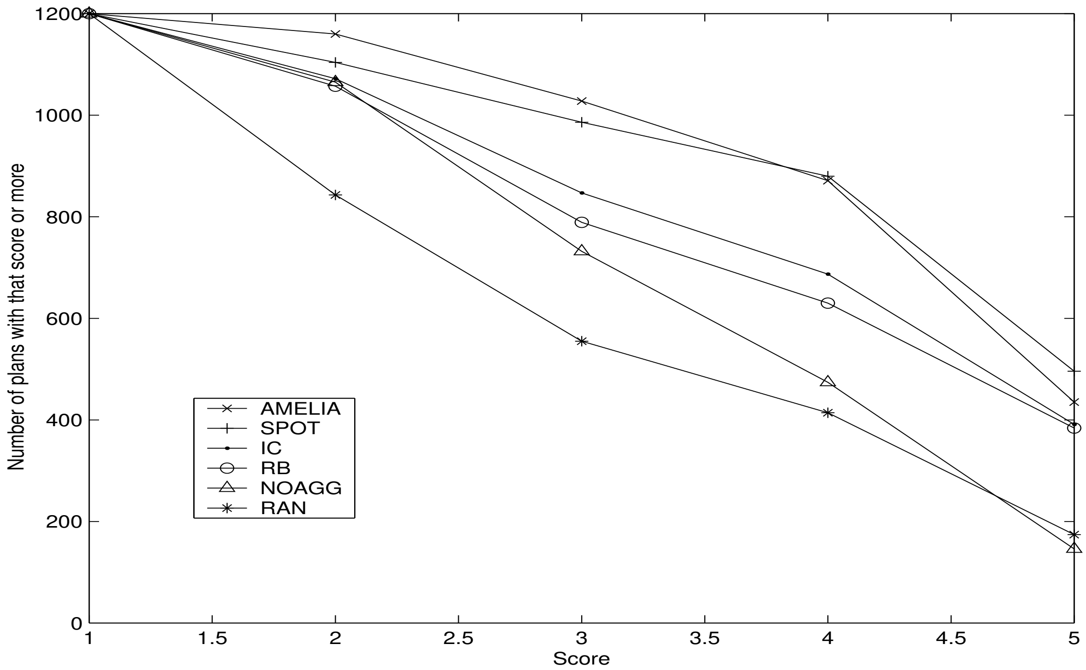

Figure 8: Chart comparing distribution of human ratings for SPoT, RBS, ICF, NoAgg and Random.

level of quality. In future work, we hope to extend SPoT and integrate it into AMELIA.

6 Acknowledments

This work was partially funded by DARPA under contract MDA972-99-3-0003.

References

Y. Freund, R. Iyer, R. E. Schapire, and Y.Singer. 1998. An efficient boosting algorithm for combining preferences. In *Machine Learning: Proc. of the Fifteenth International Conference*.

E.H. Hovy and L. Wanner. 1996. Managing sentence planning requirements. In *Proc. of the ECAI'96 Workshop Gaps and Bridges: New Directions in Planning and Natural Language Generation*.

I. Langkilde and K. Knight. 1998. Generation that exploits corpus-based statistical knowledge. In *Proc. of COLING-ACL*.

Benoit Lavoie and Owen Rambow. 1997. A fast and portable realizer for text generation systems. In *Proc. of the Third Conference on Applied Natural Language Processing, ANLP97*, pages 265–268.

J. Lester and B. Porter. 1997. Developing and empirically evaluating robust explanation generators: The knight experiments. *Computational Linguistics*, 23-1:65–103.

C. Mellish and R. Dale. 1998. Evaluation in the context of natural language generation. *Computer Speech and Language*, 12(3).

C. Mellish, A. Knott, J. Oberlander, and M. O'Donnell. 1998. Experiments using stochastic search for text planning. In *Proc. of International Conference on Natural Language Generation*, pages 97–108.

I. A. Melčuk. 1988. *Dependency Syntax: Theory and Practice*. SUNY, Albany, New York.

O. Rambow and T. Korelsky. 1992. Applied text generation. In *Proc. of the Third Conference on Applied Natural Language Processing, ANLP92*, pages 40–47.

M. Walker, O. Rambow, and M. Rogati. 2001. Spot: A trainable sentence planner. In *Proc. of the North American Meeting of the Association for Computational Linguistics*.

M. A. Walker. 2000. An application of reinforcement learning to dialogue strategy selection in a spoken dialogue system for email. *Journal of Artificial Intelligence Research*, 12:387–416.

C.L. Yeh and C. Mellish. 1997. An empirical study on the generation of anaphora in chinese. *Computational Linguistics*, 23-1:169–190.