Supporting Linguistic Consistency and Idiosyncracy with an Adaptive Interface Design

Jill Fain Lehman
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
jef@f.cs.cmu.edu

Abstract

Despite the goal to permit freedom of expression, natural language interfaces remain unable to recognize the full range of language that occurs in spontaneously generated user input. Simply increasing the linguistic coverage of a large, static interface is a poor solution; as coverage increases, response time decreases, regardless of whether the extensions benefit any particular user. Instead, we propose that an *adaptive interface* be dedicated to each user. By automatically acquiring the idiosyncratic language of each individual, an adaptive interface permits greater freedom of expression while slowing system response only insofar as there is ambiguity in the individual's language. The usefulness of adaptation relies on the presence of three regularities in users' linguistic behaviors: within-user consistency, across-user variability, and limited user adaptability. We show that these behaviors are characteristic of users under conditions of frequent use.

1. Introduction

Research in designing natural language interfaces attempts to facilitate human-computer interaction by allowing a user the power and ease of her usual forms of expression to accomplish a task. Unfortunately, current technology falls short of this ideal. No existing interface permits all of the complexities of natural language: including the full range of potentially relevant vocabulary, idiomatic phrases, and extra-grammatical constructions. In addition, a tension exists between providing full freedom of expression and providing reasonable response time. Due to language's local ambiguity, significant increases in the coverage of a grammar are usually accompanied by intolerable decreases in system performance.

The standard resolution to this tension is the "monolithic" interface. For a given domain and task, the designer must choose a restricted sublanguage that has the property of being both computationally tractable and "user-friendly" for all users. Three assumptions underlie such a design:

- It is desirable to build a single interface for all users.
- There is a tractable subset of language for the domain and task that is both common to and natural for all users.
- If a user's preferred form of expression differs from the common subset, the user can easily adapt her style to conform to the limitations imposed by the interface.

How well does an interface based on these assumptions serve the individual? To the extent that a user's preferred forms of expression lie outside the restricted subset, the system relies on her adaptability and is, therefore, no more facilitating of the interaction than a system-supplied command language. In other words, this design sacrifices expressibility for responsiveness. On the other hand, to the extent a user's preferred forms of expression lie inside the restricted subset but correspond to only a small fraction of the sublanguage, the user pays a price in response time for the system's ability to understand utterances she will never use. Thus the design also sacrifices responsiveness for expressibility.

As an alternative, we propose the *adaptive interface* design. Under this paradigm each user has a dedicated interface capable of understanding her idiosyncratic language and limited in its responsiveness only by the ambiguity inherent in that language. This single-user approach is possible through a technique called *adaptive parsing*, a learning method in which the system automatically acquires a user's preferred forms of expression by growing its kernel grammar dynamically in response to user interactions. In essence, the learning mechanism explains perceived errors in the input utterance as one or more deviations with respect to the system's current grammar (a deviation corresponds to an insertion, deletion, substitution, or transposition of text with respect to a known grammatical form). The explanation is then transformed into new grammatical components that are capable of recognizing the general structure of the deviation in future interactions.

The practicality of this approach has been demonstrated by the implementation of an adaptive interface called CHAMP which acquired eight very different idiosyncratic grammars from users' spontaneously generated input (see [4, 5]). The assumptions upon which CHAMP is based differ significantly from those of a monolithic design. In the remainder of this paper we examine the assumptions underlying adaptive parsing and demonstrate their validity.

2. Behavioral Hypotheses

There are three conditions necessary for an adaptive interface to benefit user performance. Each condition corresponds, in turn, to an hypothesis about users' linguistic behavior:

- **Condition 1: within-user consistency.** We hypothesize that in a natural language interface used frequently by the same individual over time, the user tends to rely on those forms of expression that she remembers as having worked in the past. In other words, frequent use leads to a *stylized* and *self-bounded* grammar. If the hypothesis is false and a user does not rely primarily on what worked in the past, then a fixed, restricted sublanguage is necessary—a system based on automatic adaptation would have no protection against the intractability associated with steadily increasing coverage.
- **Condition 2: across-user variability.** We hypothesize that given an environment that permits a reasonable subset of English, users will demonstrate significant idiosyncracy in their preferred forms of expression. If this hypothesis is false and all users employ the same restricted sublanguage then a methodology such as that of Kelley [3] or Good *et al.* [1] is preferable; one builds a monolithic system incorporating the complete common subset through a generate-and-test cycle.
- **Condition 3: limited user adaptability.** A monolithic approach to interface design assumes that the cognitive burden of learning the interface's sublanguage is a minor one. We hypothesize that a system that permits the natural expression of idiosyncratic language patterns results in better task performance than a system that forces the user to adapt to it. If our hypothesis is wrong and the user is able to adapt quickly and with little effort to a restricted sublanguage, then adaptation on the part of the system is no longer a clear advantage.

The experiments discussed in the next section validate our three behavioral hypotheses and prove that with frequent use the required conditions are met.

3. Description of the Experiments

Data was collected at two distinct points in our research on adaptive interfaces. To demonstrate the feasibility of adaptive parsing and to establish the validity of the behavioral hypotheses, we first simulated an adaptive interface using a hidden-operator paradigm.[1] The algorithm used by the hidden operator formed the basis for the implementation of the working interface, CHAMP. Next, to verify that the implementation captured the crucial aspects of the simulated environment (and to assure ourselves that our initial results did not depend on having a human in the loop), we used CHAMP to run the initial experiments "on-line" with two new users.

Materials. A calendar scheduling task was chosen because it has fairly well-defined semantics and requires frequent interactions over time. The stimuli consisted of calendar pages with pictorial representations of changes to be made to the schedule. The stimuli was pictorial in order to minimize its influence on the user's forms of expression.

Users. All of the users were female adults between twenty and sixty-five years of age. None had prior experience with natural language interfaces. Each was employed as a secretary or executive assistant in either a business or university environment. Although not every user maintained a calendar for her employer, each had kept a personal calendar for at least one year.

Procedure. The *Adapt* condition was designed to test our hypotheses about the development of self-bounded, idiosyncratic grammars. Users were told that they would be typing the input to a *natural language learning*

[1] A hidden-operator experiment is one in which the user believes she is interacting with a computer system when, in reality, the feedback is produced by an experimenter simulating the system's behavior. For examples of other studies conducted under the hidden-operator paradigm, see [1], [3], [9], and [10]. The hidden-operator experiments described here were first reported in [6].

interface that would increase its knowledge of English while helping them keep an on-line calendar for a busy professor/entrepreneur. In each session, the user was asked to effect the changes in ten to twelve task pictures by typing her commands as if she were "speaking or writing to another person." Although no time limit was enforced, the instructions told users to proceed to the next task after three unsuccessful attempts.

In responding to a user's utterances, the hidden operator or CHAMP had the user's current grammar and lexicon available. With respect to that information, an utterance was judged either *parsable*, *learnable* (containing at most two deviations), or *uninterpretable* (more than two deviations). A parsable utterance was accepted without further interaction and the desired action was performed. The interpretation for a learnable utterance had to be verified by the user. If verified, the current grammar was augmented to capture the new forms, and then the action was performed. If the utterance was uninterpretable, a user in the hidden-operator experiments was told which segments, if any, appeared understandable and was asked to try again. In the on-line experiments, users were simply asked to try again. Each of the six users in the Adapt condition participated in nine sessions; Users 1, 2, 3, and 4 took part in the original hidden-operator experiments, Users 9 and 10 participated in the later on-line experiments.

To evaluate the counterargument to adaptation that maintains that the user will naturally adapt to the system's limitations faster than the system can usefully adapt to the user's idiosyncrasies, two users participated in the *No-Adapt* control condition. In this condition, the experiment was conducted as outlined above except that users were told the system was a natural language interface (not a learning interface), and the kernel grammar was never changed. Although the system remained more permissive of extragrammaticality than the average natural language interface (by allowing up to two ungrammatical constructions with respect to the kernel grammar), the user was nevertheless restricted to a fixed sublanguage. Any improvement in task performance would therefore be attributable to the user's adaptability. The number of sessions varied in the control condition according to the availability of the users. User 6 participated in three sessions, User 8 in five sessions.[2]

4. Results and Discussion

With the exception noted below, all users were able to complete most tasks; they rarely required more than three attempts per task in the initial sessions and averaged only slightly more than one attempt per task in later sessions. The results for users in the Adapt condition indicated a high degree of self-limiting behavior, converging towards a user-specific subset of English. Users in the No-Adapt condition showed very limited adaptability. In the remainder of this section we review the results pertaining to each behavioral hypothesis in turn.

Within-user consistency. To measure this aspect of user behavior, we examine the number of constituents added to an individual's grammar over time. If a user is self-bounding in her language, then this number should decrease asymptotically. Figure 4-1 shows the measure for each of the six users in the Adapt condition. The figure indicates, by user and session, the number of new constructions per *interpretable* sentence which is calculated as the total number of changes to the grammar divided by the total number of *parsable* or *learnable* sentences. The number of *uninterpretable* sentences in a session is given in parentheses. Shaded areas of the figures indicate sessions following the introduction of the supplementary instructions to work quickly.[3]

---

[2]Users 5 and 7 participated in a different control condition (*Adapt/Echo*) which was run in response to arguments by Slator *et al.* [9] that users want to and will learn a mnemonic command syntax and domain-specific vocabulary. Users in the Adapt/Echo condition were given the same instructions as those in the Adapt condition except that they were told the system would display a paraphrase of their utterance in an "internal form" that they were free to incorporate into their commands or to ignore, as they saw fit. No experimental effect was observed in this condition. Further discussion can be found in [4].

[3]The supplementary instructions were included in response to the observation that users try to employ terser, more economical language as time goes on [7, 2, 3, 8]. It was unclear whether nine sessions would be adequate time for the tendency to be manifested. Thus, the additional instructions were included to subject the system to more extreme linguistic behavior within the time available. Note, however, that the supplementary instructions were not given until the user's grammar had essentially stabilized. Figure 4-2 shows examples of movement toward terser language by our users.

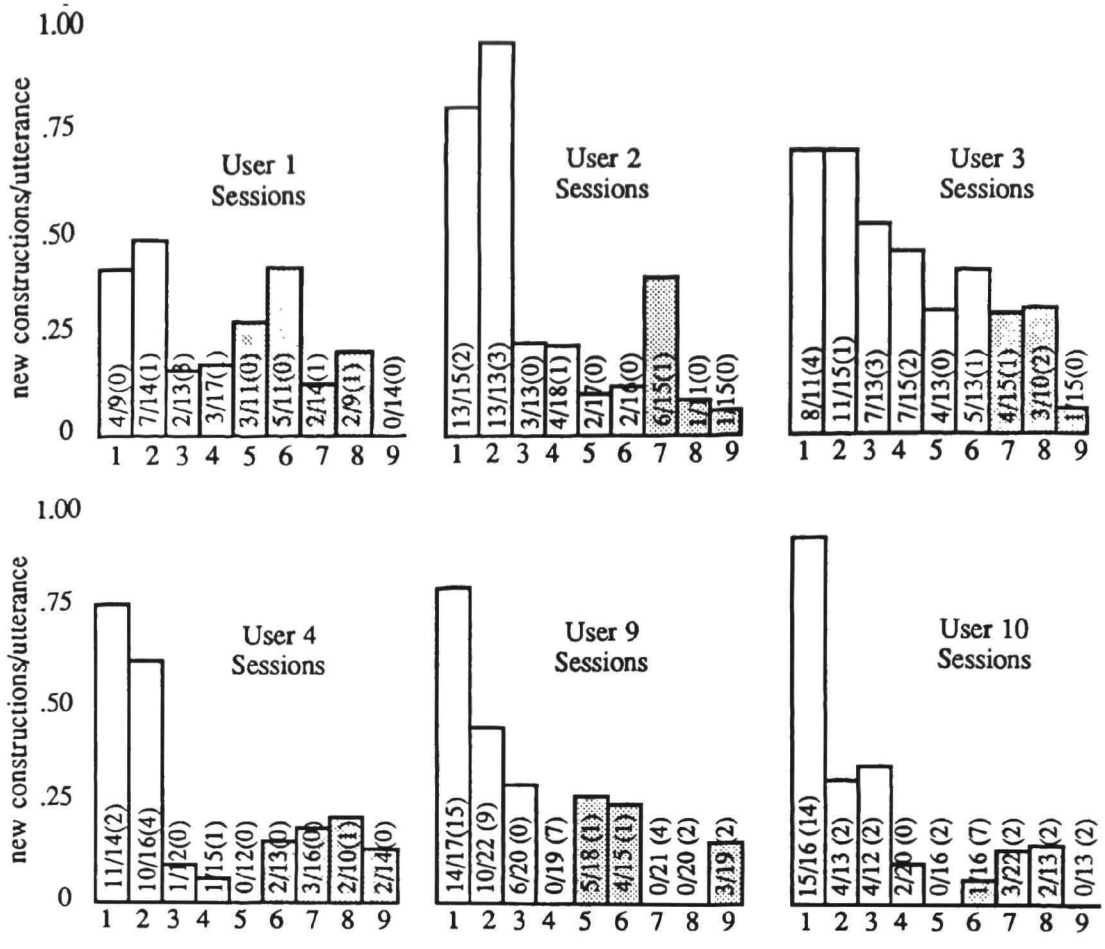

Figure 4-1: Self-limiting behavior in the Adapt condition. Shaded sessions followed the supplementary instructions to work quickly. The number of unparsable utterances for each session is given in parentheses.

Figure 4-1 shows both the effect of the choice of kernel forms and the difference in the rates at which language stabilization occurs. Observe that User 1's grammar required the least learning and stabilized quickly. She began with terse utterances and a vocabulary that was quite close to the kernel chosen for the experiment. User 2's grammar showed less initial resemblance, but she employed it consistently and, thus, stabilized quickly as well. User 3 had a tendency to be both polite and verbose, so her utterances were more likely to contain new forms and stabilization occurred more slowly. User 4 showed rapid stabilization because the new forms she introduced in sessions one and two, while not terse, were quite "natural" (that is, they were used consistently and without modification in subsequent sessions).

Users 9 and 10 showed the same overall decrease in the number of new constructions per interpretable sentence as the users in the hidden-operator experiments. They also showed the same local increase and return to stabilization after the supplementary instructions were given. User 9's behavior was particularly interesting. Despite the fact that the instructions clearly stated that the system could not understand its own output, she insisted on using the system's messages as a template for her own commands (User 7 showed a less extreme form of the same behavior in the hidden-operator experiments). By the end of her third session, User 9 had essentially used the adaptive qualities of the interface to bootstrap the system to understand a terser form of its own output.

Across-user variability. The high degree of idiosyncracy in users' preferred forms of expression can be seen in Figure 4-2. Note that much of that idiosyncracy is maintained despite the movement over time to terser utterances. For users in the Adapt condition of the hidden-operator experiments, the grammars resulting from

interactions with the simulated interface show little overlap. One could argue, however, that the idiosyncracy found is an artifact of the hidden operator's freedom when adding a new form to a user's grammar. CHAMP's uniform adaptation and generalization mechanism provide the opportunity to examine across-user variability in a more meaningful way.

u1.1.9	move Anderson seminar on June 10 to room 7220
u1.9.3	move 1:30 PM class on June 23 to room 5409
u2.1.14	change the location of the meeting on June 10, 1986 at 12:00 noon ending at 1:30 from room 5409 to 7220
u2.9.3	change Class 15-731 location to Room 5409 on June 23
u3.1.13	CogSci Seminar will be held on Tuesday, June 10, 1986 from 12:30 to 1:30 in room 7220; Speaker will be Anderson.
u3.9.3	Class 15-731 will be held in Room 5409 instead of 7220 on June 23rd.
u4.1.13	change room number of seminar from 5409 to 7220 on Tuesday June 10 from 12:30 to 1:30
u4.9.3	class 15-731 on Monday, June 23 will be in room 5409
u9.1.27	change location of COGSCI Seminar to Room 7220 June 6
u9.9.5	change class 15-731 June 19 from 7220 to 5409 at 1:30pm
u10.1.26	Change June 6 Cogsci seminar to WeH 7220
u10.9.3	Change June 19 class to room 5409

Figure 4-2: Sample sentences from sessions one and nine for users in the Adapt condition (u*i.j.k* refers to the *k*th utterance in session *j* for user *i*).

To attain a quantitative measure for idiosyncracy, we examine the acceptance rate for each user's total set of utterances across the nine sessions using her own final grammar (her grammar at the end of session nine) and using each of the other users' final grammars. If each final grammar recognizes approximately the same language, then there should be no significant difference in the acceptance rates. In contrast, Figure 4-3 clearly demonstrates a wide range of variability. For example, User 2's final grammar is able to parse 88% of her own sentences, but can parse only 11% of the utterances from User 3.[4] Note that in no instance does the grammar for another user approach the acceptance rate of the user's own final grammar. The high values on the diagonal, which denote the acceptance rate of each user's final grammar for her own utterances, result both from within-user consistency and from the efficacy of the learning mechanism.

CHAMP also allows us to quantify the degree to which idiosyncracy contributes to poor response in a monolithic interface design. Recall that in motivating the idea of adaptive parsing, we argued from a theoretical point of view that the monolithic approach must engender some degree of mismatch between the language accepted by the system and the language employed by any particular user. The mismatch is of two types: language the user prefers that cannot be understood by the interface and language the interface understands that is never employed by the user. To compare designs, we must control for the first kind of mismatch by guaranteeing that the monolithic grammar can parse at least as many sentences in the test set as can CHAMP. To accomplish this, we construct a monolithic grammar by adding to the kernel the union of the users' final grammars minus any redundant components.

Figure 4-4 demonstrates the monolithic grammar's performance on each user's utterances. The figure provides values for three measures: the average number of search states, the average number of roots (each root

[4]We do not compare final grammars across experiments because the on-line users started with a slightly different kernel grammar than the hidden-operator users; a cross-experiment comparison would show inflated variability due to those differences.

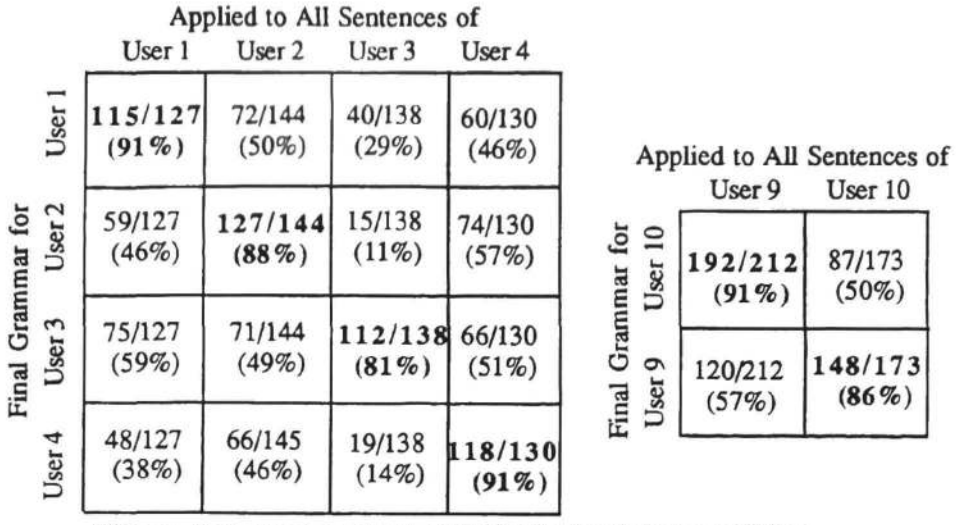

Figure 4-3: Across-user variability in the Adapt condition.

represents a unique parse tree; multiple roots correspond to multiple interpretations), and the number of utterances parsable by the monolithic grammar that were not also parsable by the user's final grammar. Due to the implementation, increased values for the first two measures (states and roots) can correspond to relatively independent sources of ambiguity in the grammar. From the user's point of view, an increase in the number of states corresponds to decreased response time, while a higher number of roots corresponds to an increase in the number of interactions required to arrive at a unique interpretation for the utterance.

User	Avg. states$_{user}$	Avg. states$_{mono}$	Avg. roots$_{user}$	Avg. roots$_{mono}$	Additional$_{mono}$
1	37.6	109.6	1.5	7.7	4
2	59.8	155.5	1.6	8.5	4
3	43.3	156.5	2.0	9.3	1
4	40.7	148.3	1.5	7.1	1
9	80.6	96.7	2.6	2.6	0
10	45.8	68.2	1.7	3.4	0

Figure 4-4: A comparison of the relative costs of parsing with a monolithic grammar versus parsing with the idiosyncratic grammars provided by adaptation.

Let us first consider the results for the users in the hidden-operator experiments. Figure 4-4 shows that in every instance the average number of search states examined by the monolithic grammar is significantly greater than the number required by the user's idiosyncratic grammar. Although we do gain the ability to understand ten additional sentences in the test set (this occurs when adaptations learned for User *i* are able to parse a sentence that was rejected for User *j*), the trade-off hardly seems favorable. User 1 would find a 3% increase in the number of her sentences understood by the interface (4/127) at a cost of almost three times the search. User 2 also receives about a 3% increase (4/144) at a cost of about two and a half times the search. User 3 and User 4 suffer the most; each receives an increase of less than 1% in accepted sentences but must wait through 3.6 times the search. The trade-off seems even less favorable when we examine the difference in the number of roots produced by each system. The monolithic static grammar produces about five times as many roots on the average as a user's adapted grammar. Thus, in addition to slower response, the monolithic design requires more user interactions to arrive at a unique interpretation.

Turning to the results for users in the on-line experiments, we note that User 9 would notice little difference between interactions with a monolithic interface and those with CHAMP. User 10, on other hand, would notice

some difference, primarily in the number of interactions required to resolve the extra interpretations produced by the monolithic grammar. Most of the extra work required to accept User 10's sentences is caused by User 9's grammar. In other words, User 10 must pay for User 9's idiosyncracy in the monolithic design.

In essence, Figure 4-4 shows the price paid by each user for search over those portions of the monolithic system's language she will never use. We know that we cannot control for this kind of mismatch without sacrificing freedom of expression because of the real variability in preferred forms across users. Although it may be possible to achieve the same acceptance rate with somewhat less ambiguity in a much more carefully crafted monolithic grammar, the across-user variability is not going to disappear. Adaptation resolves this dilemma without requiring inordinate skill as a grammar designer. By learning the user's preferred forms of expression and engendering only those mismatches required by its small kernel, an adaptive interface keeps the cost of understanding each user in proportion to the inherent ambiguity in her idiosyncratic language.

Limited user adaptability. Figure 4-5 displays the results for the No-Adapt condition. Recall that the purpose of this variation is to study the claim that people adapt well enough to obviate the need for system adaptation. We measure user adaptation categorizing each utterance according to the minimum number of deviations required to interpret it. If the user adapted to the limitations of the system, we should see a general increase in *interpretable* sentences (those containing zero, one or two deviations).

The graph for User 6 does show an increase in interpretable sentences over her three sessions. Her behavior is interesting in two respects. First, it is reminiscent of User 1's behavior; User 6's grammar was fairly close to the kernel, especially with respect to vocabulary, and she tended to rely on terse forms. As a result, her utterances presented fewer loci for deviation than might occur in a more verbose style. Second, User 6, like the users in the Adapt condition, relied on those forms that had worked in the past. When an utterance met with failure, her next try was usually a minor variation that almost always succeeded. In short, User 6 performed successfully in the No-Adapt condition, but relatively little adaptation was required of her.

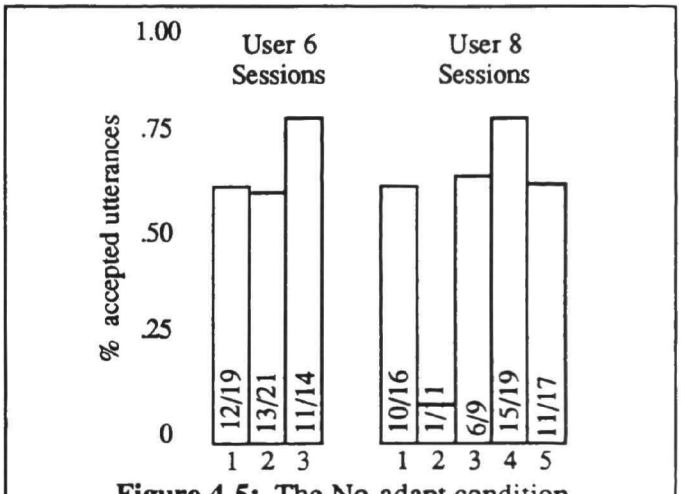

Figure 4-5: The No-adapt condition.

Figure 4-5 also displays User 8's performance in the No-Adapt condition. There is no pattern of improvement in User 8's data; her success at adapting to the interface's subset of English was extremely limited. Both her work experience and the content of her utterances lead to the conclusion that little of her behavior was attributable to task misunderstanding. Put simply, she was unable or refused to adapt. Her linguistic style, like that of User 3, is best characterized as verbose. In response to a typical utterance, the hidden operator told her which segments could be parsed and asked her to try again. She would then type exactly those segments just echoed, with no connecting text. When the terse form met with failure she would gravitate back to overly explanatory sentences.

User 8's performance in session two was so poor (in 31 minutes she produced one learnable sentence with two deviations) and her frustration so great, that she was given hints about how to use the system more effectively prior to beginning session three.[5] Although the help appeared to improve her performance in sessions three and four, in session five she still generated a high percentage of unparsable forms. This is in sharp contrast to the experiences of users in the Adapt condition, most of whom had no unparsable utterances by session five.

[5]Specifically, she was told to try to type simple but fully grammatical sentences and to think of the system as someone to whom she was writing instructions. She was the only user who was given this aid.

The contrast is sharpened by examining task completion. As mentioned above, all other users completed almost all tasks (98% completion on the average); User 8 completed only 44% of the tasks in her five sessions. This was due largely to the fact that she never managed to find an acceptable form of expression for an entire class of tasks (those involving the **change** action). Thus, although User 8 relied on the few forms that had worked in the past, her performance must be seen as a strong counterargument to the notion that everyone finds it natural or easy to adapt to a system's linguistic limitations.

5. Summary and Future Work

The purpose of these experiments was to establish certain behavioral characteristics of frequent users which, in turn, guarantee the conditions necessary for an adaptive interface to benefit user performance. The behavior of the users in the Adapt condition demonstrates the self-limiting, idiosyncratic language use predicted. As a result, we may assume that the conditions of within-user consistency and across-user variability will be met, making single-user, adaptive interfaces a practical and desirable alternative to a monolithic interface design.

The experimental results validate the idea of limited user adaptability as well. Although we cannot guarantee that a particular user will find adaptation to an interface's limitations difficult, User 8 serves as a dramatic example of the kind of limited user adaptability we must be prepared to encounter. Further, a comparison of Users 6 and 8 supports Watt's argument [11] that the ease with which a user adapts to a rigid interface depends significantly on a fortuitous correspondence between the user's natural language and the sublanguage provided by the interface designer. More importantly, the performance of the users in the adaptive condition demonstrates that an initial lack of correspondence *can* be overcome by system adaptation.

Despite these results, it remains to be seen whether the behavioral assumptions upon which adaptation relies hold over more realistic periods of time (nine months rather than nine sessions) and over very different tasks. Assuming such generality can be demonstrated, we believe adaptive parsing represents a significant step forward in the effort to provide users with full freedom of expression in computer interactions.

Acknowledgements: This research was partially supported by the Defense Advanced Research Projects Agency (DOD), ARPA Order No. 4976, and monitored by the Air Force Avionics Laboratory under Contract F33615-87-C-1499. The author was partially supported during this research by a Boeing Company fellowship. The content and clarity of this paper were improved by the comments of Angela Kennedy Hickman, Rick Lewis, and Thad Polk.

References

1. Good, M. D., Whiteside, J. A., Wixon, D.R., and Jones, J.J., "Building a User-Derived Interface," *Communications of the ACM*, Vol. 27, No. 10, 1984.
2. Harris, L. R., "Experience with ROBOT in 12 Commercial Natural Language Database Query Applications," *Proceedings of the Sixth International Joint Conference on Artificial Intelligence*, 1979.
3. Kelley, J. F., "An Iterative Design Methodology for User-Friendly Natural Language Office Information Applications," *ACM Transactions on Office Information Systems*, Vol. 2, No. 1, 1984.
4. Lehman, J. Fain, *Adaptive Parsing: Self-extending Natural Language Interfaces*, PhD dissertation, Carnegie Mellon University, 1989.
5. Lehman, J. Fain, "Adaptive Parsing: A General Method for Learning Idiosyncratic Grammars," *Proceedings of the Sixth International Conference on Machine Learning*, 1990.
6. Lehman, J. Fain, and Carbonell, J. G., "Learning the User's Language, A Step Towards Automated Creation of User Models," in *User Modelling in Dialog Systems*, Wahlster, W., and Kobsa, A., eds., Springer-Verlag, 1989.
7. Malhotra, A., "Knowledge-Based English Language Systems for Management Support: An Analysis of Requirements," *Proceedings of the Fourth International Joint Conference on Artificial Intelligence*, 1975.
8. Rich, E., "Natural Language Interfaces," *IEEE Computer*, Vol. 17, No. 9, 1984.
9. Slator, B. M., Anderson, M. P., and Conley, W., "Pygmalion at the Interface," *Communications of the ACM*, Vol. 29, No. 7, 1986.
10. Tennant, H., *Evaluation of Natural Language Processors*, PhD dissertation, University of Illinois, 1980.
11. Watt, W. C., "Habitability," *American Documentation*, July 1968.