Evaluating Spoken Dialogue Agents with PARADISE: Two Case Studies

Marilyn A. Walker, Diane J. Litman, Candace A. Kamm and Alicia Abella
AT&T Labs—Research
180 Park Avenue
Florham Park, NJ 07932-0971 USA
walker,diane,cak,abella@research.att.com

May 6, 1998

Abstract

This paper presents PARADISE (PARAdigm for DIalogue System Evaluation), a general framework for evaluating and comparing the performance of spoken dialogue agents. The framework decouples task requirements from an agent's dialogue behaviors, supports comparisons among dialogue strategies, enables the calculation of performance over subdialogues and whole dialogues, specifies the relative contribution of various factors to performance, and makes it possible to compare agents performing different tasks by normalizing for task complexity. After presenting PARADISE, we illustrate its application to two different spoken dialogue agents. We show how to derive a performance function for each agent and how to generalize results across agents. We then show that once such a performance function has been derived, that it can be used both for making predictions about future versions of an agent, and as feedback to the agent so that the agent can learn to optimize its behavior based on its experiences with users over time.

1 Introduction

Interactive spoken dialogue systems are based on many component technologies: speech recognition, text-to-speech, natural language understanding, natural language generation, and database query languages. While there has been a great deal of progress in developing well-understood evaluation metrics for many of these components [Sparck-Jones and Galliers, 1996; Walker, 1989; Hirschman *et al.*, 1990; Lee, 1988; Ralston *et al.*, 1995; Pallett, 1985], there has been less progress in developing metrics and frameworks for evaluating dialogue systems that integrate all of these components.

One problem is that dialogue evaluation is not reducible to transcript evaluation, or to comparison with a wizard's reference answers [Bates and Ayuso, 1993; Polifroni *et al.*, 1992; Price *et al.*, 1992], because the set of potentially acceptable dialogues can be very large. Another problem is that there are many potential metrics that can be used to evaluate a dialogue system. For example, a dialogue system can be evaluated by measuring the system's ability to help users achieve their goals, the system's robustness in detecting and recovering from errors of speech recognition or of understanding, and the overall quality of the system's interactions with users [Danieli and Gerbino, 1995; Hirschman and Pao, 1993; Polifroni *et al.*, 1992; Price *et al.*, 1992; Sparck-Jones and Galliers, 1996; Simpson and Fraser, 1993]. It is not clear how different metrics overlap with one another, or what the tradeoffs between metrics might be.

Previous proposals for dialogue evaluation have focused on the development of dialogue metrics that can be classified as *objective* or *subjective*. *Objective metrics* can be calculated without recourse to human judgement, and in many cases, can be logged by the spoken dialogue system so that they can be calculated automatically. Objective metrics that have been used to evaluate a dialogue as a whole include [Abella *et al.*, 1996; Ciaremella, 1993; Danieli and Gerbino, 1995; Hirschman *et al.*, 1990; Hirschman *et al.*, 1993; Polifroni *et al.*, 1992; Price *et al.*, 1992; Sparck-Jones and Galliers, 1996; Smith and Hipp, 1994; Smith and Gordon, 1997; Walker, 1996]:

- percentage of correct answers with respect to a set of reference answers
- percentage of successful transactions or completed tasks[1]

[1]Task-based success measures are only objective when the user's task is well-defined,

- number of turns or utterances
- dialogue time or task completion time
- mean user response time
- mean system response time
- percentage of diagnostic error messages
- percentage of "non-trivial" (more than one word) utterances
- mean length of "non-trivial" utterances

Subjective metrics require subjects using the system and/or human evaluators to categorize the dialogue or utterances within the dialogue along various qualitative dimensions. Because these metrics are based on human judgements, such judgements need to be reliable across judges in order to compete with the reproducibility of metrics based on objective criteria. Subjective metrics can still be quantitative, as when a ratio between two subjective categories is computed. Subjective metrics that have been used to evaluate dialogue systems include [Danieli and Gerbino, 1995; Boyce and Gorin, 1996; Hirschman and Pao, 1993; Simpson and Fraser, 1993; Danieli *et al.*, 1992; Bernsen *et al.*, 1996]:

- percentage of implicit recovery utterances (where the system uses dialogue context to recover from errors of partial recognition or understanding)
- percentage of explicit recovery utterances
- percentage of contextually appropriate system utterances
- cooperativity (the adherence of the system's behavior to Grice's conversational maxims [Grice, 1967])
- percentage of correct and partially correct answers

as in a controlled experiment. In a field study, determining whether a user accomplished his or her task typically requires a subjective judgment.

- percentage of appropriate and inappropriate system directive and diagnostic utterances
- user satisfaction (user's perceptions about the usability of a system, usually assessed with multiple choice questionnaires that ask users to rank the system's performance on a range of usability features according to a scale of potential assessments)

Both the objective and subjective metrics have been very useful to the spoken dialogue community. For example, the metrics have enabled comparisons between different systems carrying out the same task. However, both the metrics and the current methodologies for using them also have important limitations.

One widely acknowledged limitation is that the use of reference answers makes it impossible to compare systems that use different dialogue strategies for carrying out the same task. This is because the reference answer approach requires canonical responses (i.e., a single "correct" answer) to be defined for every user utterance, even though there are potentially many correct answers. For example, it is not possible to use the reference answer approach to compare a system that gives a list of database values as a response to a database query with a system that gives an abstract summary as a response.

A second limitation is that interdependencies between metrics are not yet well understood. For example, various metrics may be highly correlated with one another, and thus provide redundant information on performance. Determining correlations requires a suite of metrics that are widely used, and testing whether correlations hold across multiple dialogue applications.

A third limitation arises from the inability to tradeoff or combine various metrics and to make generalizations [Fraser, 1995; Sparck-Jones and Galliers, 1996]. Consider a comparison of two train timetable information agents [Danieli and Gerbino, 1995], where Agent A in Dialogue 1 uses an explicit confirmation strategy, while Agent B in Dialogue 2 uses an implicit confirmation strategy:

(1) User: I want to go from Torino to Milano.
Agent A: Do you want to go from Trento to Milano? Yes or No?
User: No.

(2) User: I want to travel from Torino to Milano.
Agent B: At which time do you want to leave from Merano to Milano?
User: No, I want to leave from Torino in the evening.

In their evaluation, Danieli and Gerbino found that Agent A had a higher transaction success rate and produced fewer inappropriate and repair utterances than Agent B, and that Agent B produced dialogues that were approximately half as long as Agent A's. However, due to an inability to combine metrics, they could not report whether Agent A's higher transaction success or Agent B's efficiency was more critical to performance.

The ability to identify how multiple factors impact performance is also critical for making generalizations across systems performing different tasks [Cohen, 1995; Sparck-Jones and Galliers, 1996]. It would be useful to know how users' perceptions of performance depend on the strategy used, and on tradeoffs among factors like efficiency, usability, and accuracy. In addition to agent factors such as the differences in dialogue strategy seen in Dialogues 1 and 2, task factors such as database size and environmental factors such as background noise may also be relevant predictors of performance.

Walker proposed that a combined performance metric for dialogue systems could be derived as a weighted linear combination of a task-based success measure and dialogue costs. She showed that, given particular assumptions about the weights for dialogue costs, effective dialogue strategies depend on an interaction between task complexity and the hearer's working memory [Walker, 1993; Walker, 1996]. However, she did not specify how the weights in the performance function could be computed.

This paper presents PARADISE, a general framework for evaluating and comparing the performance of spoken dialogue agents that addresses these limitations [Walker *et al.*, 1997]. PARADISE supports comparisons among dialogue strategies by providing a task representation that decouples *what* an agent needs to achieve in terms of the task requirements from *how* the agent carries out the task via dialogue. PARADISE uses a decision-theoretic framework to specify the relative contribution of various factors to an agent's overall *performance*. Following [Walker, 1993], performance is modeled as a weighted function of a task-based success measure and dialogue-based cost measures. However, here we show how to solve for the weights by correlating user satisfaction with performance. We will also show that, once a performance function has been derived, it can be used both to make predictions

about future versions of the agent and as the basis of feedback to the agent so that the agent can learn to optimize its behavior based on its experiences with users over time.

The goal of this paper is both to explain PARADISE and to illustrate its application. In Section 2 we introduce PARADISE, and for expository purposes, illustrate its use in a simplified train schedule information domain. In Section 3 we illustrate the use of PARADISE in the evaluations of two spoken dialogue agents: ELVIS [Walker *et al.*, 1997], an agent for accessing email over the phone; and TOOT [Litman *et al.*, 1998], an agent for accessing online Amtrak train schedule information. We show how to derive a performance function for each of these agents, and how to generalize results across agents. In Section 4 we discuss how we can use the derived performance function to make predictions and to learn how to optimize agent dialogue behavior. In Section 5 we conclude and suggest future work.

2 PARADISE: A Framework for Deriving Performance Models for Dialogue

PARADISE uses methods from decision theory [Keeney and Raiffa, 1976; Doyle, 1992] to combine a disparate set of performance measures (i.e., user satisfaction, task success, and dialogue cost, all of which have been previously noted in the literature) into a single performance evaluation function. The use of decision theory requires a specification of both the objectives of the decision problem and a set of measures (known as attributes) for operationalizing the objectives. The PARADISE model is based on the structure of objectives (rectangles) shown in Figure 1.

The PARADISE model posits that performance can be correlated with a meaningful external criterion such as usability, and thus that the overall goal of a spoken dialogue agent is to maximize an objective related to usability. User satisfaction ratings [Kamm, 1995; Shriberg *et al.*, 1992; Polifroni *et al.*, 1992] have been frequently used in the literature as an external indicator of the usability of a dialogue agent. Thus, as shown in Figure 1, user satisfaction is the top level objective to be maximized. The model further posits that two types of factors are potential relevant contributors to user satisfaction (namely task success and dialogue costs), and that both

efficiency and dialogue quality are potential relevant contributors to costs. This reflects the proposal that performance is a weighted linear combination of task success and dialogue costs [Walker, 1996]. A novel aspect of PARADISE is the use of multivariate linear regression to estimate a performance equation. In particular, regression is used to solve for the weights in the performance equation, and thereby to quantify the relative contribution of the success and cost factors to user satisfaction.

The remainder of this section explains the measures (ovals in Figure 1) used to operationalize the set of objectives and the methodology for estimating a quantitative performance function that reflects the objective structure. Section 2.1 describes PARADISE's task representation, which is needed to calculate the task-based success measure described in Section 2.2. Section 2.3 describes some of the cost measures considered in PARADISE, which reflect both the efficiency and the naturalness of an agent's dialogue behaviors. Section 2.4 describes the use of linear regression and user satisfaction to estimate the relative contribution of the success and cost measures in a single performance function.

2.1 Tasks as Attribute Value Matrices

A general evaluation framework requires a task representation that decouples *what* an agent and user accomplish from *how* the task is accomplished using dialogue strategies. We propose that an *attribute value matrix (AVM)* can represent many dialogue tasks. This consists of the information that must be exchanged between the agent and the user during the dialogue, represented as a set of ordered pairs of attributes and their possible values. [2]

As a first illustrative example, consider a simplification of the train timetable domain of Dialogues 1 and 2, where the timetable only contains information about rush-hour trains between four cities, as shown in Table 1.[3] This AVM consists of four attributes (abbreviations for each attribute name are also shown).[4] In Table 1, these attribute-value pairs are annotated with the

[2]For infinite sets of values, actual values found in the experimental data constitute the required finite set.

[3]In addition to the AVMs presented in this paper, a more general set of examples can be found in [Walker *et al.*, 1997].

[4]The AVM serves as an evaluation mechanism only. We are not claiming that AVMs determine an agent's behavior or serve as an utterance's semantic representation.

direction of information flow to represent who acquires the information, although this information is not used for evaluation. During the dialogue the agent must acquire from the user the values of depart-city, arrival-city, and depart-range, while the user must acquire depart-time.

Performance evaluation for an agent requires a corpus of dialogues between users and the agent, in which users execute a set of scenarios. Each scenario execution has a corresponding AVM instantiation indicating the task information that was actually obtained via the dialogue.

For example, assume that a scenario requires the user to find a train from Torino to Milano that leaves in the evening, as in the longer versions of Dialogues 1 and 2 in Figures 2 and 3.[5]

Table 2 contains an AVM corresponding to a "key" for this scenario.

All dialogues resulting from execution of this scenario in which the agent and the user **correctly** convey all attribute values (as in Figures 2 and 3) would have the same AVM as the scenario key in Table 2. The AVMs of the remaining dialogues would differ from the key by at least one value. The AVM represents the attribute value pairs as they exist at the end of the dialog (AVMs are not created during the course of a dialog). Note that even though the dialogue strategies in Figures 2 and 3 are radically different, the AVM task representation for these dialogues is identical and the performance of the system for the same task can thus be assessed on the basis of the AVM representation.

This example used an AVM that contained a single value for each attribute. However this representation can be extended to handle multiple right answers (values). One way to do this is to represent the value as a disjunction of possible values as illustrated in table 7. Another solution is to use the entity relationship model [Korth, 1991] to design the AVM (see appendix). Note that the AVM representation is confined to a discrete (albeit large) collection of responses.

2.2 Measuring Task Success

Success at the task for a whole dialogue (or subdialogue) is measured by how well the agent and user achieve the information requirements of the

[5]These dialogues have been slightly modified from [Danieli and Gerbino, 1995]. The attribute names at the end of each utterance will be explained below.

task by the end of the dialogue (or subdialogue). This section explains how PARADISE uses the Kappa coefficient [Carletta, 1996; Siegel and Castellan, 1988] to operationalize the task-based success measure in Figure 1.

The Kappa coefficient, κ, can be calculated from a confusion matrix that summarizes how well an agent achieves the information requirements of a particular task for a set of dialogues instantiating a set of scenarios.[6] For example, Tables 3 and 4 show two hypothetical confusion matrices that could have been generated in an evaluation of 100 complete dialogues with each of two train timetable agents A and B (perhaps using the confirmation strategies illustrated in Figures 2 and 3, respectively).[7] The values in the matrix cells are based on comparisons between the dialogue and scenario key AVMs. Whenever an attribute value in a dialogue (i.e., data) AVM *matches* the value in its scenario key, the number in the appropriate diagonal cell of the matrix (boldface for clarity) is incremented by 1. The off-diagonal cells represent *misunderstandings* that are not corrected in the dialogue. Note that depending on the strategy that a spoken dialogue agent uses, confusions across attributes are possible, e.g., "Milano " could be confused with "morning."

The effect of misunderstandings that *are* corrected during the course of the dialogue are reflected in the *costs* associated with the dialogue, as will be discussed below. The individual AVMs for the dialogue and the confusion matrices derived from them reflect only the state of the information exchange at the **end** of the dialogue. The time course of the dialogue and error handling for any misunderstandings are assumed to be reflected in the costs associated with the dialogue.

The matrix in Table 3 summarizes how the 100 AVMs representing each dialogue with Agent A compare with the AVMs representing the relevant scenario keys, while the matrix in Table 4 summarizes the information exchange with Agent B. Labels v1 to v4 in each matrix represent the possible values of depart-city shown in Table 1; v5 to v8 are for arrival-city, etc. Columns represent the key, specifying which information values the agent and user were supposed to communicate to one another given a particular scenario. (The equivalent column sums in both tables reflects that users of both agents were assumed to have performed the same scenarios). Rows represent the

[6]Confusion matrices can be constructed to summarize the result of dialogues for any subset of the scenarios, attributes, users or dialogues.

[7]The distributions in the tables were roughly based on performance results in [Danieli and Gerbino, 1995].

data collected from the dialogue corpus, reflecting what attribute values were actually communicated between the agent and the user. When the data contains attribute values that are not present in the key (as in both of the case studies described below), an extra row representing "all other values" can be added for each relevant attribute.

Given a confusion matrix M, success at achieving the information requirements of the task is measured with the Kappa coefficient [Carletta, 1996; Siegel and Castellan, 1988]:

$$\kappa = \frac{P(A) - P(E)}{1 - P(E)}$$

P(A) is the proportion of times that the AVMs for the actual set of dialogues agree with the AVMs for the scenario keys, and P(E) is the proportion of times that the AVMs for the dialogues and the keys are expected to agree by chance.[8] When there is no agreement other than that which would be expected by chance, $\kappa = 0$. When there is total agreement, $\kappa = 1$. κ is superior to other measures of success such as transaction success [Danieli and Gerbino, 1995], concept accuracy [Simpson and Fraser, 1993], and percent agreement [Gale *et al.*, 1992] because κ takes into account the inherent complexity of the task by correcting for chance expected agreement. Thus κ provides a basis for comparisons across agents that are performing *different* tasks.

When the prior distribution of the categories is unknown, P(E), the expected chance agreement between the data and the key, can be estimated from the distribution of the values in the keys. This can be calculated from confusion matrix M, since the columns represent the values in the keys. In particular:

$$P(E) = \sum_{i=1}^{n} (\frac{t_i}{T})^2$$

where t_i is the sum of the frequencies in column i of M, and T is the sum of the frequencies in M ($t_1 + \ldots + t_n$).

[8] κ has been used to measure pairwise agreement among coders making category judgments [Carletta, 1996; Krippendorf, 1980; Siegel and Castellan, 1988]. Thus, the observed user/agent interactions are modeled as a coder, and the ideal interactions as an expert coder.

P(A), the actual agreement between the data and the key, can be computed over all the scenarios from the confusion matrix M:

$$P(A) = \frac{\sum_{i=1}^{n} M(i,i)}{T}$$

When it is desirable to compute a per dialogue κ, as we will do below, P(A) can be computed on a per dialogue basis by simply calculating the percent of the attribute values in the AVM instantiation for the dialogue that agree with the attribute values in the AVM key for the dialogue scenario.

Given the confusion matrices in Tables 3 and 4, P(E) = 0.079 for both agents.[9] For Agent A, P(A) = 0.795 over all scenarios and κ = 0.777, while for Agent B, P(A) = 0.59 over all scenarios and κ = 0.555, suggesting that Agent A is more successful than B in achieving the task goals.

2.3 Measuring Dialogue Costs

As shown in Figure 1, performance is also a function of a combination of cost measures. Intuitively, cost measures should be calculated on the basis of any user or agent dialogue behaviors that should be minimized. As discussed in Section 1, a wide range of both efficiency and qualitative cost measures have been used in previous work. Since it is not clear in advance of empirical work which cost factors may be the strongest contributors to user satisfaction, it is important that a wide range of these measures be used in empirical studies. Furthermore, in order to make cross-task generalizations about important factors, the same measures must be examined across different tasks. Section 3 discusses the particular set of cost measures used in our two case studies.

PARADISE represents each cost measure as a function c_i that can be applied to any (sub)dialogue. First, consider the simplest case of calculating efficiency measures over a whole dialogue. For example, let c_1 be the total number of utterances. For the whole dialogue D1 in Figure 2, c_1(D1) is 23 utterances. For the whole dialogue D2 in Figure 3, c_1(D2) is 10 utterances.

To calculate costs over subdialogues and for some of the qualitative measures, it is necessary to be able to specify which information goals each ut-

[9]Using a single confusion matrix for all attributes as in Tables 3 and 4 inflates κ when there are few cross-attribute confusions by making P(E) smaller. In some cases it might be desirable to calculate κ first for identification of attributes and then for values within attributes, or to average κ for each attribute to produce an overall κ for the task.

terance contributes to. PARADISE uses its AVM representation to link the information goals of the task to any arbitrary dialogue behavior, by tagging the dialogue with the attributes for the task.[10] This makes it possible to evaluate any potential dialogue strategies for achieving the task, as well as to evaluate dialogue strategies that operate at the level of dialogue subtasks (subdialogues).[11]

Consider the longer versions of Dialogues 1 and 2 in Figures 2 and 3. Each utterance in Figures 2 and 3 has been tagged using one or more of the attribute abbreviations in Table 1, according to the subtask(s) the utterance contributes to. As a convention of this type of tagging, utterances that contribute to the success of the whole dialogue, such as greetings, are tagged with all the attributes. Since the structure of a dialogue reflects the structure of the task [Carberry, 1989; Grosz and Sidner, 1986; Litman and Allen, 1990], the tagging of a dialogue by the AVM attributes can be used to generate a hierarchical discourse structure such as that shown in Figure 4 for Dialogue 1 (Figure 2). For example, segment (subdialogue) S2 in Figure 4 is about both depart-city (DC) and arrival-city (AC). It contains segments S3 and S4 within it, and consists of utterances U1 ... U6.

Tagging by AVM attributes is required to calculate costs over subdialogues, since for any subdialogue, task attributes define the subdialogue. For subdialogue S4 in Figure 4, which is about the attribute arrival-city and consists of utterances A6 and U6, c_1(S4) is 2.

Tagging by AVM attributes is also required to calculate the cost of some of the qualitative measures, such as number of repair utterances. (Note that to calculate such costs, each utterance in the corpus of dialogues must also be tagged with respect to the qualitative phenomenon in question, e.g. whether the utterance is a repair.[12]) For example, let c_2 be the number of repair utterances. The repair utterances for the whole dialogue D1 in Figure 2 are A3 through U6, thus c_2(D1) is 10 utterances and c_2(S4) is 2 utterances.

[10] This tagging can be hand generated, or system generated and hand corrected. Preliminary studies indicate that reliability for human tagging is higher for AVM attribute tagging than for other types of discourse segment tagging [Passonneau and Litman, 1997; Hirschberg and Nakatani, 1996].

[11] Walker et al.[1997] contains a complete example of the use of PARADISE at the subdialogue level.

[12] Previous work has shown that this can be done with high reliability [Hirschman and Pao, 1993].

The repair utterance for the whole dialogue D2 in Figure 3 is U2, but note that according to the AVM task tagging, U2 simultaneously addresses the information goals for depart-range. In general, if an utterance U contributes to the information goals of N different attributes, each attribute accounts for 1/N of any costs derivable from U. Thus, c_2(D2) is .5.

Given a set of c_i, it is necessary to combine the different cost measures in order to determine their relative contribution to performance. The next section explains how to combine κ with a set of c_i to yield an overall performance measure.

2.4 Estimating a Performance Function

Given the definition of success and costs above and the model in Figure 1, performance for any (sub)dialogue D is defined as follows:

$$\text{Performance} = (\alpha * \mathcal{N}(\kappa)) - \sum_{i=1}^{n} w_i * \mathcal{N}(c_i)$$

Here α is a weight on κ, the cost functions c_i are weighted by w_i, and $\mathcal{N}$ is a Z score normalization function [Cohen, 1995]. We assume an additive performance (utility) function because our analysis indicates that κ and the various cost factors c_i are utility independent and additive independent [Keeney and Raiffa, 1976].

The normalization function is used to overcome the problem that the values of c_i are not on the same scale as κ, and that the cost measures c_i may also be calculated over widely varying scales (e.g. response delay could be measured using seconds while, in the example, costs were calculated in terms of number of utterances). If the values of κ and c_i are not normalized, then the magnitude of the coefficients α and w_i will not reflect the relative contribution of each factor to performance. This problem is easily solved by normalizing each factor x to its Z score:

$$\mathcal{N}(x) = \frac{x - \overline{x}}{\sigma_x}$$

where σ_x is the standard deviation for x.

To illustrate the method for estimating a performance function, we will use a subset of the data from Tables 3 and 4, shown in Table 5. Table 5

represents the results from a hypothetical experiment in which eight users were randomly assigned to communicate with Agent A and eight users were randomly assigned to communicate with Agent B. Table 5 shows user satisfaction (US) ratings (discussed below), κ, number of utterances (#utt) and number of repair utterances (#rep) for each of these users. Users 5 and 11 correspond to the dialogues in Figures 2 and 3 respectively. To normalize c_1 for user 5, we determine that $\overline{c_1}$ is 38.6 and σ_{c_1} is 18.9. Thus, $\mathcal{N}(c_1)$ is -0.83. A similar calculation for user 11 gives $\mathcal{N}(c_1)$ = -1.51.

To estimate the performance function, the weights α and w_i must be solved for. Recall that the claim implicit in Figure 1 was that the relative contribution of task success and dialogue costs to performance should be calculated by considering their contribution to user satisfaction. User satisfaction is typically calculated with surveys that ask users to specify the degree to which they agree with one or more statements about the behavior or the performance of the system. A single user satisfaction measure can be calculated from a single question, as the mean of a set of ratings, or as a linear combination of a set of ratings. In Section 3, we will discuss the surveys we used for user satisfaction and the way we calculated user satisfaction ratings in our case studies. The hypothetical user satisfaction ratings shown in Table 5 range from a high of 6 to a low of 1.

Given a set of dialogues for which user satisfaction (US), κ and the set of c_i have been collected experimentally, the weights α and w_i can be solved for using multivariate linear regression. Multivariate linear regression produces a set of coefficients (weights) describing the relative contribution of each predictor factor in accounting for the variance in a predicted factor. In this case, on the basis of the model in Figure 1, US is treated as the predicted factor. Normalization of the predictor factors (κ and c_i) to their Z scores guarantees that the relative magnitude of the coefficients directly indicates the relative contribution of each factor. Regression on the Table 5 data for both sets of users tests which factors κ, #utt, #rep most strongly predicts US.

In this illustrative example, the results of the regression with all factors included shows that only κ and #rep are significant ($p < .02$). In order to develop a performance function estimate that includes only significant factors and eliminates redundancies, a second regression including only significant factors must then be done. In this case, a second regression yields the predictive equation:

$$\text{Performance} = .40\mathcal{N}(\kappa) - .78\mathcal{N}(c_2)$$

i.e., α is .40 and w_2 is .78. The results also show κ is significant at $p < .0003$, #rep significant at $p < .0001$, and the combination of κ and #rep account for 92% of the variance in US, the external validation criterion. The factor #utt was not a significant predictor of performance, in part because #utt and #rep are highly redundant. (The correlation between #utt and #rep is 0.91).

Given these predictions about the relative contribution of different factors to performance, it is then possible to return to the problem first introduced in Section 1: given potentially conflicting performance criteria such as robustness and efficiency, how can the performance of Agent A and Agent B be compared? Given values for α and w_i, performance can be calculated for both agents using the equation above. The mean performance of A is -.44 and the mean performance of B is .44, suggesting that Agent B may perform better than Agent A overall.

The evaluator must then however test these performance differences for statistical significance. In this case, a t test shows that differences are only significant at the $p < .07$ level, indicating a trend only.

Finally, note that once the weights in the performance function have been solved for, user satisfaction ratings no longer need to be collected. Instead, predictions about user satisfaction can be made on the basis of the predictor variables.

3 Two Case Studies

Above we both presented the PARADISE framework and used PARADISE to evaluate two hypothetical dialogue agents in a simplified train timetable task domain. In particular, we used PARADISE to derive a performance function for our hypothetical task by estimating the relative contribution of a set of potential predictors to user satisfaction. The PARADISE methodology consisted of the following steps:

- definition of a task and a set of scenarios;
- specification of the AVM task representation;

- experiments with alternate dialogue agents for the task;
- calculation of user satisfaction using surveys;
- calculation of task success;
- calculation of dialogue cost using efficiency and qualitative measures;
- estimation of a performance function using linear regression and values for user satisfaction, task success and dialogue costs;
- application of the performance function to experimental performance data to compare performance differences among agent strategies, tasks, or other experimental variables

In this section we describe the application of PARADISE to experimental studies of two actual spoken dialogue systems: (1) ELVIS (EmaiL Voice Interactive System) [Walker *et al.*, 1997]; and (2) TOOT [Litman *et al.*, 1998], an agent for accessing online Amtrak schedule information over the phone. Our experimental studies consist of controlled experiments with clearly defined user tasks. Section 3.1 describes the spoken dialogue platform used to implement both applications. Section 3.2 describes the experimental setup for each dialogue agent. Section 3.3 discusses the performance equation derived for each application, and generalizes results across applications.

3.1 Spoken Dialogue Platform

Both ELVIS and TOOT are implemented using a general-purpose platform for voice dialogue agents, which combines automatic speech recognition (ASR), an audio server for both voice recordings and text-to-speech (TTS), a phone interface, a module for application specific functions, and modules for specifying the dialogue manager and the application grammars [Kamm *et al.*, 1997].

The speech recognizer is a speaker-independent hidden Markov model (HMM) system with context-dependent phone models for telephone speech and constrained grammars defining the vocabulary that is permitted at any point in a dialogue [Rabiner *et al.*, 1996]. The platform supports barge-in, so that the user can interrupt the agent when it is speaking, e.g., when reading a long email message. The grammar module supports dynamic grammar

generation and loading because the ASR vocabulary must change during the interaction, e.g., to support selection of email messages by content fields such as sender and subject.

The text-to-speech technology is concatenative diphone synthesis [Sproat and Olive, 1995]. The audio server can switch between voice recordings and TTS and integrate voice recordings with TTS. However, the ELVIS and TOOT agents only use TTS due to the dynamic and unpredictable nature of both web-based tables and email messages.

The application module for each application provides application specific functions. For example, ELVIS provides functions for accessing message attributes such as subject and author, and using these attributes in folder summaries and for selecting messages. TOOT's application module provides functions for accessing tables stored on AMTRAK's web site and for selecting subsets of these tables that satisfy constraints in the user's query.

The dialogue manager is based on a state machine. Each state specifies transitions to other states and the conditions that license these transitions, as well as a grammar for what the user can say. State definitions also include the specification of agent *prompts* in terms of templates, with variables that are instantiated each time the state is entered. Prompts include: (1) an initial prompt, which the agent says upon entering the state (this may include a response to the user's current request); (2) a help prompt which the agent says if the user says *help*; (3) multiple rejection prompts which the agent says if the speech recognizer confidence is too low to continue without more user input; (4) multiple timeout prompts which the agent produces if the user doesn't say anything.

Each of these specifications is affected by the agent's dialogue strategy. An agent's dialogue strategy is implemented as a combination of the prompts that are played to the user and the state transitions that the agent makes in response to the user's utterance. In particular, alternative prompts can be specified for all types of prompts (initial, help, rejection and timeout) to provide alternate dialogue strategies in each state.

3.2 Experimental Design

Both the ELVIS and TOOT experiments were designed to test different dialogue strategies for the particular agent involved in the experiment. Both experiments required every user to complete a set of application tasks in con-

versations with a particular version of the agent. Instructions to the users were given on a set of web pages; there was one page for each experimental task. Each web page consisted of a brief general description of the functionality of the agent, a list of hints for talking to the agent, a task description, and information on how to call the agent. Each page also contained a form for specifying information acquired from the agent during the dialogue, and a survey, to be filled out after task completion, designed to probe the user's satisfaction with the system. Users read the instructions in their offices before calling the agent from their office phone. In this section, we describe the dialogue strategies tested in each case study (Section 3.2.1), the tasks that each case study involved (Section 3.2.2), and how the values for task success, the dialogue costs, and user satisfaction were logged and extracted from the experimental data (Section 3.2.3).

3.2.1 Dialogue Strategies Tested

The types of dialogue strategies that were tested were different for each agent. The ELVIS experiments tested different initiative management strategies, while the TOOT experiments tested different information presentation strategies.[13]

In ELVIS, we tested a system-initiative (SI) versus a mixed-initiative (MI) dialogue strategy [Walker *et al.*, 1997; Walker and Whittaker, 1990]. The SI strategy uses *directive prompts* [Kamm, 1995], and controls the dialogue by directing the user to select among a limited number of choices, as in Dialogue 3:

(3) E: Hi, I've got your mail. Say Read, or Summarize, or say Help for more options.
U: Read.
E: Select messages by subject, by sender, or in order of arrival?
U: Sender.
E. Which sender?
U: Kim. ....

The MI strategy never volunteers information. It assumes that users

[13]The ELVIS experiments also tested other types of dialogue strategies, e.g. information presentation strategies, although we will not present those results here.

know what to do and therefore it lets the user control the dialogue as in Dialogue 4:

(4) E: Hi, I've got your mail.
U: Read me my messages from Kim...

The hypotheses we investigated were: (1) the SI strategy would be easier for novice users and would result in better ASR performance; (2) the MI strategy would be easier to use and would be preferred by users once they had some experience with the system; and (3) the ability to summarize and select messages by content attributes such as sender and subject would lead to greater dialogue efficiency. Our results are reported in detail in Walker et al.[1997; 1998a].

In TOOT, we tested a literal versus a cooperative dialogue strategy for presenting information in a displayless environment [Abella *et al.*, 1996; Goddeau *et al.*, 1996; Seneff *et al.*, 1995; Meng *et al.*, 1996; Pieraccini *et al.*, 1997; Allen and Perrault, 1980]. The difference in the presentation strategies was linked to two conditions: (A) when there was no information that matched a user's query; and (B) when there was too much matching information to present in a single utterance. For condition A, the first strategy reports only the literal answer to the query, as in the fragment of the TOOT dialogue shown in (5):

(5) T: Do you want me to find the trains going to Chicago from Philadelphia on Sunday around 10:30 am now?

U: Yes.

T:There are no trains going to Chicago from Philadelphia on Sunday around 10:30 am. Do you want to continue and find a new set of trains?

The second strategy automatically relaxes the user's time constraint, and also allows the user to perform additional relaxations, as in (6):

(6) ...
T: There are no trains going to Chicago from Philadelphia on Sunday around 10:30 am. The closest earlier train leaves at 9:28 am every day, and it takes 1 day 3 hours 36 minutes. The closest later train leaves

at 11:45 am on Saturday and Sunday, and it takes 22 hours 5 minutes. Please say "relax" to change your departure time or travel day, or say "continue" if my answer was sufficient, or say "repeat" to hear this message again.

For condition B, the first strategy groups the information to be presented into manageable units, then lists each unit, as in (7):

(7) T: Do you want me to find the trains going to Philadelphia from New York City on the weekend around 4 o'clock pm now?

U: Yes.

T: There are 7 trains available around 4 o'clock pm. Here are the first 3 trains. The 1st train leaves at 3:00 pm on Saturday and Sunday, and it takes 1 hour 12 minutes. The 2nd train leaves at 3:20 pm every day, and it takes 1 hour 22 minutes. The 3rd train leaves at 4:00 pm on Sunday, and it takes 1 hour 12 minutes. Do you want to hear the rest of the trains?

The second strategy instead summarizes the range of trains available, then allows the user to group and list the trains, or to further constrain the query, as in (8):

(8) ...
T: There are 7 trains available around 4 o'clock pm. Here is the earliest train we have. The first train leaves at 3:00 pm on Saturday and Sunday, and it takes 1 hour 12 minutes. Here is the latest train we have. The seventh train leaves at 5:00 pm on Saturday, and it takes 1 hour 12 minutes. Please say "list" to hear trains 3 at a time, or say "add constraint" to constrain your departure time or travel day, or say "continue" if my answer was sufficient, or say "repeat" to hear this message again.

The hypothesis we investigated was that strategy performance was dependent on task. Our results are reported in detail in [Litman *et al.*, 1998].

3.2.2 Task Descriptions

For ELVIS, each user performed three tasks in sequence, and each task consisted of two subtasks, totaling six subtasks in all. Each subtask was represented by a scenario where the agent and the user had to exchange information about criteria for selecting messages and information within the message body. Two sample task scenarios are shown below, where scenarios 1.1 and 1.2 were done in the same conversation with ELVIS:

- TASK 1.1: You are working at home in the morning and plan to go directly to a meeting when you go into work. Kim said she would send you a message telling you where and when the meeting is. Find out **the Meeting Time** and **the Meeting Place**.

- TASK 1.2: The second task involves finding information in a different message. Yesterday evening, you had told Lee you might want to call him this morning. Lee said he would send you a message telling you where to reach him. Find out **Lee's Phone Number**.

Scenario 1.1 is represented in terms of the attribute value matrix (AVM) in Table 6. The AVM representation for all six subtasks is similar to Table 6.

For TOOT, each user performed 4 tasks in sequence. Each task was represented by a scenario where the user had to find a train satisfying certain constraints, by using the agent to retrieve and process online train schedules. A sample task scenario is as follows:

- TASK 1: Try to find a train going to **Boston** from **New York City** on **Saturday** at **6:00 pm**. If you cannot find an exact match, find the one with the **closest** departure time. Please write down the **exact departure time** of the train you found as well as the **total travel time**.

The scenarios in TOOT are represented by an attribute value matrix which includes the attributes in Table 1 (with different possible values) as well as the attributes "depart-day" and "travel-time."

3.2.3 Collection and Extraction of Experimental Measures

In each case study, experimental results were collected by three means, and a set of cost metrics were extracted. Cost metrics are given in **boldface** below. First, all of the dialogues were recorded. This allows us to transcribe utterances and to check aspects of the timing of the interaction, such as whether there were long delays for agent responses, and how many times users barged in on agent utterances (the variable named **Barge Ins**). In addition, the recording was used to calculate the total time of the interaction (the variable named **Elapsed Time**).

Second, the agent's dialogue behavior was logged in terms of entering and exiting each state in the state transition table for the dialogue. For each state, the system logged the dialogue strategy that the agent executed, as well as the number of timeout prompts that were played (**Timeout Prompts**), the number of times that ASR rejected the user's utterance (**ASR Rejections**), and the number of times that the user accessed the state-specific help messages by saying *Help* (**Help Requests**). The number of **System Turns** and the number of **User Turns** were also calculated on the basis of this data. In addition, the ASR result for the user's utterance was logged, including the log-likelihood score that ASR assigned to the utterance. The recordings were used in combination with the logged ASR result to calculate a concept accuracy measure for each utterance by hand (i.e., whether the recognizer's output correctly captured the task-related information in the utterance). Mean concept accuracy was then calculated over the whole dialogue and used as a **Mean Recognition Score** for the dialogue.

Third, users were required to fill out a web page form after each task. Users first specified whether they thought they had completed the task or not (**Completed**). Users next specified the information that they had acquired from the agent (e.g., in ELVIS, the values for Email.att1 and Email.att2 in Table 6). This information was used in conjunction with the (logged) information that the agent had acquired from the user (e.g., in ELVIS, the value for Selection Criteria in Table 6) to compute **Kappa**. Finally, users responded to a survey on their subjective evaluation of their satisfaction with the agent's performance. The survey as it was instantiated for ELVIS follows:

- Was ELVIS easy to understand in this conversation? (**TTS Performance**)

- In this conversation, did ELVIS understand what you said? (**ASR Performance**)
- In this conversation, was it easy to find the message you wanted? (**Task Ease**)
- Was the pace of interaction with ELVIS appropriate in this conversation? (**Interaction Pace**)
- In this conversation, did you know what you could say at each point of the dialogue? (**User Expertise**)
- How often was ELVIS sluggish and slow to reply to you in this conversation? (**System Response**)
- Did ELVIS work the way you expected him to in this conversation? (**Expected Behavior**)
- In this conversation, how did ELVIS's voice interface compare to the touch-tone interface to voice mail?[14] (**Comparable Interface**)
- From your current experience with using ELVIS to get your email, do you think you'd use ELVIS regularly to access your mail when you are away from your desk? (**Future Use**)

Most question responses ranged over values such as (*almost never, rarely, sometimes, often, almost always*). Each of these responses was mapped to an integer in 1 ... 5. Some questions had (*yes, no, maybe*) responses. Each question emphasized the user's experience with the system in the current conversation, with the hope that satisfaction measures would indicate perceptions specific to each conversation, rather than reflecting an overall evaluation of the system over the three tasks. We calculated a **User Satisfaction** score (cumulative satisfaction) for each dialogue by summing the scores of the multiple choice questions in the survey.

In terms of the model shown in Figure 1, the **User Satisfaction** score is used as a measure of User Satisfaction. **Kappa** measures actual task success. The measures of **System Turns**, **User Turns** and **Elapsed Time** are

[14]The instantiation of the survey for TOOT did not contain a "comparable interface" question.

efficiency cost measures. The qualitative measures are **Completed**, **Barge Ins**, **Timeout Prompts**, **ASR Rejections**, **Help Requests**, and **Mean Recognition Score**. Because we were concerned about the reliability of the qualitative measures, we mainly used measures that the system could log automatically, which did not require hand labeling. The only measures in the experiments reported here that require hand labeling are **Mean Recognition Score** and **Barge Ins**. As we mentioned above, we believe that at this stage, it is impossible to prejudge which measures contribute to user satisfaction. Instead, we use PARADISE and empirical data to tell us which measures have merit, and to quantify their relative importance.

3.3 Deriving a Performance Equation from Experimental Data

This section describes the results of two different experiments, one with ELVIS and one with TOOT. Table 7 illustrates the type of information that was accumulated at the end of each experiment, as a result of the logging and hand-labeling of data described above.

Each row in the table represents a dialogue in the collected dialogue corpus. For each dialogue, the logged information consists of the user, the task the user performed, the actions that the agent performed in each dialogue state which depends on the agent's dialogue strategy parameters, and values for all of the other variables discussed in Section 3.2.3. These include values for **User Satisfaction** (US), for the task success measure **Kappa** (κ), for cost efficiency measures such as **Elapsed Time** (ET) and **System Turns** (STs), and for qualitative costs measures such as **Timeout Prompts** (TOs) and **Mean Recognition Score** (MRS). We show below how experimental data such as these can be used to derive a performance function within applications, and how results can be generalized across applications. For more detailed information about these experiments, see [Walker *et al.*, 1997; Walker *et al.*, 1998a; Litman *et al.*, 1998].

The ELVIS experimental data consists of three dialogues for each of three tasks for 48 users, for a total of 144 dialogues and 6481 turns. Most users successfully completed each task in about 5 minutes; average κ over all subjects and tasks was .82 and mean elapsed time was 315 seconds. P(E), estimated by using the data from all subjects and tasks, was .5, while P(A) was .91.

When we applied PARADISE to ELVIS experimental data to derive a performance equation, we found that user perception of task completion (COMP, p = .004), elapsed time (ET, p = .001) and mean recognition score (MRS, p < .0001) were the factors that significantly contributed to user satisfaction:

$$\text{Performance} = .20\mathcal{N}(COMP) + .45\mathcal{N}(MRS) - .23\mathcal{N}(ET)$$

COMP was a significant factor in predicting user satisfaction (rather than κ) because user's perceptions of task completion sometimes varied from κ. It is interesting to note that although task completion and efficiency have typically been assumed to be the most important factors in a system's success, the relative magnitude in the performance equation of the coefficient for MRS, a qualitative measure, is greater than the measures related to task success (COMP) and the efficiency measure (ET).

Plugging our experimental data back into this performance function shows that the mean performance of the system-initiative (SI) strategy is greater than the mean performance of the mixed-initiative (MI) strategy. There were two main reasons for this difference: the MI strategy was harder to learn and MRS for MI was much lower on average. However, as we had hypothesized, performance for MI increased over successive tasks as users' expertise increased. We are continuing to explore how users' expertise affects performance of the MI strategy.

The TOOT experimental data consists of 4 tasks for each of 12 users, for a total of 48 dialogues and 1344 turns. Average κ over all subjects and tasks was .81 and mean elapsed time was 267 seconds. P(E) (chance expected agreement, estimated by using the data from all subjects and tasks) was .05, indicating that the TOOT task was inherently more complex than the ELVIS task. That is, in TOOT, it would have been harder to *guess* the correct attribute values in the AVM scenario keys. Although P(A) averaged over all subjects and tasks was .82 (in contrast to .91 for ELVIS), the κ scores indicate that TOOT and ELVIS in fact perform comparably with respect to task success. This is because κ adjusts P(A) to account for P(E).

The performance equation derived from the application of PARADISE to the experimental data for TOOT shows that user's perception of task completion (COMP, p < .0002), mean recognition score (MRS, p < .003), and the number of times that the user interrupted the system (BargeIns, p < .0004) were significant factors:

$$\text{Performance} = .45\mathcal{N}(COMP) + .35\mathcal{N}(MRS) - .42\mathcal{N}(BargeIns)$$

Our performance function demonstrates that TOOT performance involves a combination of task success and dialogue quality factors. It is interesting to note that in the case of TOOT, none of the efficiency measures are significant predictors of performance. Like the results from ELVIS, the results from TOOT again draw into question the frequently made assumption that efficiency is one of the most important measures of system performance.

Plugging the experimental TOOT data back into the derived performance function shows that the mean performance scores of the literal and cooperative dialogue strategies are not significantly different. As discussed in [Litman *et al.*, 1998], this reflects the fact that of the three (fairly equally weighted) factors in the performance function, only COMP significantly differs across the literal and cooperative presentation strategies.

Finally, remember that one of the major goals of PARADISE is the ability to make generalizations about factors that contribute to performance in dialogue agents by combining data across applications. Performance function estimation should be done iteratively over many different tasks and dialogue strategies to see which factors generalize. In this way, the field can make progress on identifying the relationship between various factors and can move towards more predictive models of spoken dialogue agent performance.

Because the same measures were collected in ELVIS and TOOT, we can make a first step towards generalization by combining the data from the two experiments. When we combine the data, we find that COMP ($p = .0001$), MRS ($p < .0001$), and ET ($p = .0004$) are significant:

$$\text{Performance} = .23\mathcal{N}(COMP) + .43\mathcal{N}(MRS) - .21\mathcal{N}(ET)$$

Thus the factors that are important over both applications generalize the factors that were shown to be important in the ELVIS data set, but the relative magnitude of the coefficients change when modeling the combined data set, reflecting the strong contribution that COMP plays in the performance model for TOOT.[15] Interestingly, this performance equation demonstrates

[15]The fact that ET but not BargeIns generalizes across applications might also reflect the fact that the combined dataset contains 144 ELVIS dialogues and only 48 TOOT dialogues. In future work we will explore the use of sampling techniques to balance combined data.

a trade-off between recognizer performance (MRS) and efficiency (ET) that was noted in other work [Danieli and Gerbino, 1995]. Note again, that in the generalized performance equation as in ELVIS, recognizer performance contributes more to user satisfaction than efficiency, even though it has often been assumed that users care more about efficiency. Our hypothesis is that people are not very accurate perceivers for efficiency measures such as elapsed time. This would be consistent with other findings in the literature on spoken dialogue interfaces and user interfaces in general [Geelhoed *et al.*, 1995]. Instead, we hypothesize that users are more attuned to qualitative aspects of the dialogue that are highly correlated with recognizer performance. In particular, users may be particularly attuned to the confusion that results from misunderstandings and to requests to repeat what they have said. Future work utilizing PARADISE in different tasks will attempt to verify this hypothesis as well as develop other generalizations about performance in spoken dialogue agents.

4 Using the Performance Equation

In Section 3 we showed how PARADISE could be used to evaluate different versions of an agent and to make generalizations across different agents about the factors that are critical predictors of performance. We did this by (1) deriving a performance function for each agent, and then (2) by combining experimental data across multiple agents and then deriving a performance function for the combined data. Here we will show that once we have derived a performance equation, that there are many ways in which we can use it.

4.1 Learning to Optimize Dialogue Strategy Choices

One potentially broad use of the PARADISE performance function is as feedback to the agent. There are many stochastic optimization algorithms that require an objective performance function to provide feedback to the agent. These algorithms make it possible for the agent to learn how to optimize its behavior automatically. Some possible algorithms are Genetic Algorithms [Goldberg, 1989; Russell and Norvig, 1995], Q-learning [Watkins, 1989], TD-Learning [Tesauro, 1992], and Adaptive Dynamic Programming [Bellman, 1957; Sutton, 1991; Russell and Norvig, 1995; Barto *et al.*, 1995].

The basic idea is to apply the performance function to any dialogues D_i in which the agent conversed with a user. Then each dialogue has an associated real numbered performance value P_i, which represents the performance of the agent for that dialogue. If the agent can make different choices in the dialogue about what to do in various situations, this performance feedback can be used to help the agent learn automatically, over time, which choices are optimal. Learning could be either *on-line* so that the agent tracks its behavior on a dialogue by dialogue basis, or *off-line* where the agent collects a lot of experience and then tries to learn from it.

In previous work, Walker proposed that the performance value P_i can be used as the agent fitness function in a genetic algorithm (a type of reinforcement learning algorithm) [Walker, 1993]. Biermann and Long applied a type of reinforcement learning algorithm to the selection of agent messages in a speech-graphics Pascal tutor [1996]. Levin and Pieraccini[1997] pointed out that dialogue must be characterized as a Markov Decision Process in order to apply reinforcement learning.

In each case, the performance value P_i is used as a 'fitness function' or a 'reward' in algorithms that assign a utility to the dialogue strategy choices that the agent made in that dialogue. The basis of such algorithms is the Maximum Expected Utility Principle:

> **Maximum Expected Utility Principle**: An optimal action is one that maximizes the expected utility of outcome states.

In principle, this approach would allow the agent to learn how to optimize over a large set of actions (dialogue strategy choices). Here, we illustrate the approach by applying Adaptive Dynamic Programming to the trivial problem of deciding whether ELVIS's system-initiative strategy or mixed-initiative strategy is more optimal. In this case, we already know that the system-initiative strategy is more optimal: the point here is simply to illustrate the method. See [Walker *et al.*, 1998b] for experiments on more complex action choices.

Calculating the optimal action via Adaptive Dynamic Programming is based on the following recursive equation, which is the basis for dynamic programming [Bellman, 1957]:

$$U(S_i) = R(S_i) + \max_a \sum_j M_{ij}^a U(S_j)$$

where S_i, S_j are states that the agent can be in, S_j is a successor state to state S_i, $U(S_i)$ is the utility of state S_i, $R(S_i)$ is a reward associated with being in state S_i, and M_{ij}^a is the probability of reaching state S_j if action a is taken in state S_i. The state transition model M_{ij}^a is learned from direct observations of transitions in the state/action history sequences that are logged as a set of dialogues are collected.

Value Iteration is the process of updating the estimate of $U(S_i)$, based on updated utility estimates for neighboring states, i.e. the equation above becomes:

$$U_{t+1}(S_i) = R(S_i) + \max_a \sum_j M_{ij}^a U_t(S_j)$$

where $U_t(S_i)$ is the utility estimate for state S_i after t iterations. The performance value P_i, calculated by applying the PARADISE performance equation, is used as the utility for the final states of the dialogue. Value iteration stops when the difference between $U_t(S_i)$ and $U_{t+1}(S_i)$ is below a threshold, and utility values have been associated with the earlier states in the dialogue, in which various action (strategy) selections were made. The result of value iteration is a table, specifying for each state S_i, the action A_i that will maximize the expected utility of the agent.

The result of applying the ADP algorithm to the choice that ELVIS makes between system and mixed initiative is illustrated in Figure 5. At the end of 108 training sessions (dialogues), U(system-initiative) is estimated at .39 and U(mixed-initiative) is estimated at -.43.

We are currently applying these techniques to optimize choice of presentation strategies for summarizing email folders in ELVIS and schedule tables in TOOT and for reading email messages in ELVIS [Walker *et al.*, 1998b].

4.2 Making Predictions

Another potentially broad use of the PARADISE performance function is to use it as a predictive model that can be both validated and refined by further experiments. For example, recall the performance equation derived from the combined ELVIS and TOOT data:

$$\text{Performance} = .23\mathcal{N}(COMP) + .43\mathcal{N}(MRS) - .21\mathcal{N}(ET)$$

This equation, which states that mean recognition score (MRS) is the most important contributor to performance, yielded the best fit to our data. Thus,

one prediction we can make from this equation is that if we can improve mean recognition score (especially if we can do so without decreasing the value of COMP or increasing ET), then we can increase performance by some amount.

One way that this prediction could be validated would be to replace the recognizer used in our implementation platform with an improved version (i.e., one that yielded a higher mean recognition score), rerun our experiments with new users, and confirm that performance does indeed increase by the predicted amount. Unfortunately we do not currently have access to a better speech recognizer, so cannot perform these experiments at this time.

However, even without changing our speech recognizer, we believe that we can still improve mean recognition score, by changing various dialogue behaviors of our agents. For example, the mean recognition score in the system-initiative version of ELVIS is higher than in the mixed-initiative version [Walker *et al.*, 1997]. It has also been suggested that by changing the wording of reprompts, user inputs can be elicited that are less likely to be misrecognized [Boyce and Gorin, 1996].

These observations have motivated our current work - the development of *adaptive* versions of ELVIS and TOOT, that can recognize poor speech recognition performance and change future dialogue behavior accordingly. In particular, our research involves the development of a classifier for "problematic" dialogues (with respect to speech recognition), and the use of this classifier to determine when to adapt an agent's dialogue behavior (e.g., to change from a more natural mixed-initiative dialogue strategy to a system-initiative strategy). We predict that this ability to recognize and respond to poor speech recognition will yield higher mean recognition scores, and based on the performance equation shown above, higher agent performance. We will test this prediction by using PARADISE to evaluate adaptive and non-adaptive versions of ELVIS and TOOT.

5 Discussion

This paper presented the PARADISE framework for evaluating spoken dialogue agents, and illustrated its application in two case studies. The PARADISE performance measure is a function of both task success and dialogue costs. The methodology for deriving this performance measure from data consists of the following steps:

- definition of a task and a set of scenarios;
- specification of the AVM task representation;
- experiments with alternate dialogue agents for the task;
- calculation of user satisfaction using surveys;
- calculation of task success using κ;
- calculation of dialogue cost using efficiency and qualitative measures;
- estimation of a performance function using linear regression and values for user satisfaction, κ and dialogue costs;
- application of the performance function to experimental performance data to compare performance differences among agent strategies, tasks, or other experimental variables;
- comparison with other agents/tasks to determine which factors generalize;
- refinement of the performance model.

The PARADISE framework has a number of advantages compared to previous work. First, PARADISE supports comparisons among dialogue strategies, with a task representation that decouples *what* an agent needs to achieve in terms of the task requirements from *how* the agent carries out the task via dialogue. Second, because PARADISE's task success measure κ normalizes for task complexity, PARADISE provides a basis for comparing agents performing *different* tasks. Because κ depends on P(A) and P(A) measures the percentage of attributes that were acquired correctly, our success measure is not binary (i.e., all or none), but rather can reflect partial success at achieving a task. Third, PARADISE supports performance evaluation at any level of a dialogue, because task success and dialogue costs can be calculated for any dialogue subtask. Because performance can be measured over any subtask and dialogue strategies can range over subdialogues or the whole dialogue, performance can be associated with individual dialogue strategies. Fourth, the PARADISE performance function combines success measures

with both objective and subjective cost measures, and specifies how to quantify the relative contributions of these measures to overall performance. To our knowledge, we are the first to propose using user satisfaction to determine weights on factors related to performance.

In addition, the PARADISE framework is broadly integrative, incorporating aspects of transaction success, concept accuracy, multiple cost measures, and user satisfaction. In PARADISE, transaction success is reflected in κ, corresponding to dialogues with a P(A) of 1. The PARADISE performance measure also captures information similar to concept accuracy, where low concept accuracy scores translate into either higher costs for acquiring information from the user, or lower κ scores.

One limitation of the PARADISE approach is that the current task-based success measure does not handle the possibility that some solutions might be better than others. For example, in the train timetable domain, we might like our task-based success measure to give higher ratings to agents that suggest express over local trains, or that provide helpful information that was not explicitly requested, especially since the better solutions might occur in dialogues with higher costs. It might be possible to address this limitation by using the interval scaled data version of κ [Krippendorf, 1980]. Another possibility is to simply substitute a domain-specific task-based success measure in the performance model for κ.

Our approach also does not make a distinction between providing a wrong solution and a failure to provide a solution. Subsequently we did not address what the impact would be of providing a wrong solution (or likewise a failure to provide a solution) to user satisfaction.

Another possible limitation is that PARADISE currently models performance as a linear combination of task success and dialogue costs. This choice was made because κ and costs appear to be both additive independent and utility independent [Keeney and Raiffa, 1976]. However, it is possible that user satisfaction data collected in future experiments (or other data such as willingness to pay or use) would indicate otherwise. If so, continuing use of an additive function might require a transformation of the data, a reworking of the model shown in Figure 1, or the inclusion of interaction terms in the model [Cohen, 1995].

We believe that the evaluation model presented here has many applications in spoken dialogue processing, and that the framework is also applicable to other dialogue modalities, and to human-human task-oriented

dialogues. In addition, while there are many proposals in the literature for algorithms for dialogue strategies that are cooperative, collaborative or helpful to the user [Webber and Joshi, 1982; Pollack *et al.*, 1982; Joshi *et al.*, 1984; Chu-Carroll and Carberry, 1995], very few of these strategies have been evaluated as to whether they improve any measurable aspect of a dialogue interaction. As we have demonstrated here, any dialogue strategy can be evaluated, so it should be possible to show that a cooperative response, or other cooperative strategy, actually improves task performance by reducing costs or increasing task success. We hope that this framework will be broadly applied in future dialogue research.

6 Acknowledgments

We would like to thank James Allen, Jennifer Chu-Carroll, Morena Danieli, Wieland Eckert, Giuseppe Di Fabbrizio, Jeanne Fromer, Don Hindle, Julia Hirschberg, Shri Narayanan, Shimei Pan, Jay Wilpon, Steve Whittaker and three anonymous reviewers for helpful discussion and comments on earlier versions of this paper.

7 Appendix

The AVM representation introduced in section 2.1 can be extended using the entity relationship model to handle multiple right answers. As an illustration we will use the task scenario depicted in Table 6. Figure 6 depicts the scheme for this task.

Figure 7 illustrates the instantiation of the task scheme shown in figure 6.

This representation is a formal extension of the AVM and can be used to handle multiple answers. The added advantage of this representation is that the information can be organized very systematically into a relational database (Sybase, Oracle, etc.), analyzed and easily accessed for evaluation purposes.

References

[Abella *et al.*, 1996] Alicia Abella, Michael K Brown, and Bruce Buntschuh.

Development principles for dialog-based interfaces. In *ECAI-96 Spoken Dialog Processing Workshop*, Budapest, Hungary, 1996.

[Allen and Perrault, 1980] James F. Allen and C. Raymond Perrault. Analyzing intention in utterances. *Artificial Intelligence*, 15:143–178, 1980.

[Barto *et al.*, 1995] A.G. Barto, S. J. Bradtke, and S. P. Singh. Learning to act using real-time dynamic programming. *Artificial Intelligence Journal*, 72(1-2):81–138, 1995.

[Bates and Ayuso, 1993] Madeleine Bates and Damaris Ayuso. A proposal for incremental dialogue evaluation. In *Proceedings of the DARPA Speech and NL Workshop*, pages 319–322, 1993.

[Bellman, 1957] Richard E. Bellman. *Dynamic Programming*. Princeton University Press, Princeton, N.J., 1957.

[Bernsen *et al.*, 1996] Niels Ole Bernsen, Hans Dybkjaer, and Laila Dybkjaer. Principles for the design of cooperative spoken human-machine dialogue. In *International Conference on Spoken Language Processing, ICSLP 96*, pages 729–732, 1996.

[Biermann and Long, 1996] A. W. Biermann and Philip M. Long. The composition of messages in speech-graphics interactive systems. In *Proceedings of the 1996 International Symposium on Spoken Dialogue*, pages 97–100, 1996.

[Boyce and Gorin, 1996] S. Boyce and A. L. Gorin. User interface issues for natural spoken dialogue systems. In *Proceedings of ISSD*, pages 65–68, 1996.

[Carberry, 1989] S. Carberry. Plan recognition and its use in understanding dialogue. In A. Kobsa and W. Wahlster, editors, *User Models in Dialogue Systems*, pages 133–162. Springer Verlag, Berlin, 1989.

[Carletta, 1996] Jean C. Carletta. Assessing the reliability of subjective codings. *Computational Linguistics*, 22(2):249–254, 1996.

[Chu-Carroll and Carberry, 1995] Jennifer Chu-Carroll and Sandra Carberry. Response generation in collaborative negotiation. In *Proceedings*

of the Conference of the 33rd Annual Meeting of the Association for Computational Linguistics, pages 136–143, 1995.

[Ciaremella, 1993] A. Ciaremella. A prototype performance evaluation report. Technical Report Project Esprit 2218 SUNDIAL, WP8000-D3, 1993.

[Cohen, 1995] Paul R. Cohen. *Empirical Methods for Artificial Intelligence.* MIT Press, Boston, 1995.

[Danieli *et al.*, 1992] M. Danieli, W. Eckert, N. Fraser, N. Gilbert, M. Guyomard, P. Heisterkamp, M. Kharoune, J. Magadur, S. McGlashan, D. Sadek, J. Siroux, and N. Youd. Dialogue manager design evaluation. Technical Report Project Esprit 2218 SUNDIAL, WP6000-D3, 1992.

[Danieli and Gerbino, 1995] M. Danieli and E. Gerbino. Metrics for evaluating dialogue strategies in a spoken language system. In *Proceedings of the 1995 AAAI Spring Symposium on Empirical Methods in Discourse Interpretation and Generation*, pages 34–39, 1995.

[Doyle, 1992] Jon Doyle. Rationality and its roles in reasoning. *Computational Intelligence*, 8(2):376–409, 1992.

[Fraser, 1995] Norman M. Fraser. Quality standards for spoken dialogue systems: a report on progress in EAGLES. In *ESCA Workshop on Spoken Dialogue Systems Vigso, Denmark*, pages 157–160, 1995.

[Gale *et al.*, 1992] William Gale, Ken W. Church, and David Yarowsky. Estimating upper and lower bounds on the performance of word-sense disambiguation programs. In *Proc. of 30th ACL*, pages 249–256, Newark, Delaware, 1992.

[Geelhoed *et al.*, 1995] Erik Geelhoed, Peter Toft, Suzanne Roberts, and Patrick Hyland. To influence time perception. In *Proceedings of the Conference on Computer Human Interaction*, 1995.

[Goddeau *et al.*, 1996] David Goddeau, Helen Meng, Joe Polifroni, Stephanie Seneff, and Senis Busayapongchai. A form-based dialogue manager for spoken language applications. In *Fourth International Conference on Spoken Language Processing*, 1996.

[Goldberg, 1989] David Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison–Wesley, 1989.

[Grice, 1967] H. P. Grice. Logic and conversation. 1967.

[Grosz and Sidner, 1986] Barbara J. Grosz and Candace L. Sidner. Attentions, intentions and the structure of discourse. *Computational Linguistics*, 12:175–204, 1986.

[Hirschberg and Nakatani, 1996] Julia Hirschberg and Christine Nakatani. A prosodic analysis of discourse segments in direction-giving monologues. In *34th Annual Meeting of the Association for Computational Linguistics*, pages 286–293, 1996.

[Hirschman *et al.*, 1993] L. Hirschman, M. Bates, D. Dahl, W. Fisher, J. Garofolo, D. Pallett, K. Hunicke-Smith, P. Price, A. Rudnicky, and E. Tzoukermann. Multi-site data collection and evaluation in spoken language understanding. In *Proceedings of the Human Language Technology Workshop*, pages 19–24, 1993.

[Hirschman *et al.*, 1990] Lynette Hirschman, Deborah A. Dahl, Donald P. McKay, Lewis M. Norton, and Marcia C. Linebarger. Beyond class A: A proposal for automatic evaluation of discourse. In *Proceedings of the Speech and Natural Language Workshop*, pages 109–113, 1990.

[Hirschman and Pao, 1993] Lynette Hirschman and Christine Pao. The cost of errors in a spoken language system. In *Proceedings of the Third European Conference on Speech Communication and Technology*, pages 1419–1422, 1993.

[Joshi *et al.*, 1984] Aravind K. Joshi, Bonnie L. Webber, and Ralph M. Weischedel. Preventing false inferences. In *COLING84: Proc. 10th International Conference on Computational Linguistics*, pages 134–138, 1984.

[Kamm, 1995] Candace Kamm. User interfaces for voice applications. In David Roe and Jay Wilpon, editors, *Voice Communication between Humans and Machines*, pages 422–442. National Academy Press, 1995.

[Kamm *et al.*, 1997] Candace Kamm, Shrikanth Narayanan, Dawn Dutton, and Russell Ritenour. Evaluating spoken dialog systems for telecommunication services. In *5th European Conference on Speech Technology and Communication, EUROSPEECH 97*, 1997.

[Keeney and Raiffa, 1976] Ralph Keeney and Howard Raiffa. *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*. John Wiley and Sons, 1976.

[Korth, 1991] Henry F. Korth. *Database System Concepts*. McGraw-Hill, 1991.

[Krippendorf, 1980] Klaus Krippendorf. *Content Analysis: An Introduction to its Methodology*. Sage Publications, Beverly Hills, Ca, 1980.

[Lee, 1988] Kai-Fu Lee. *Large Vocabulary Speaker-Independent Continous Speech Recognizer: the Sphinx System*. PhD thesis, 1988.

[Levin and Pieraccini, 1997] E. Levin and R. Pieraccini. A stochastic model of computer-human interaction for learning dialogue strategies. In *EUROSPEECH 97*, 1997.

[Litman and Allen, 1990] Diane Litman and James Allen. Recognizing and relating discourse intentions and task-oriented plans. In Philip Cohen, Jerry Morgan, and Martha Pollack, editors, *Intentions in Communication*. MIT Press, 1990.

[Litman *et al.*, 1998] Diane Litman, Shimei Pan, and Marilyn Walker. Evaluating Response Strategies in a Web-Based Spoken Dialogue Agent. In *Proceedings of ACL/COLING 98: 36th Annual Meeting of the Association of Computational Linguistics*, 1998.

[Meng *et al.*, 1996] Helen Meng, Senis Busayapongchai, James Glass, Dave Goddeau, Lee Hetherington, Ed Hurley, Christine Pao, Joe Polifroni, Stephanie Seneff, and Victor Zue. Wheels: A conversational system in the automobile classifieds domain. In *Proceedings of the Fourth International Conference on Spoken Languaggge Processing*, 1996.

[Pallett, 1985] David S. Pallett. Performance assessment of automatic speech recognizers. *J. Res. Natl. Bureau of Standards*, 90:371–387, 1985.

[Passonneau and Litman, 1997] Rebecca J. Passonneau and Diane Litman. Discourse segmentation by human and automated means. *Computational Linguistics*, 23(1), 1997.

[Pieraccini *et al.*, 1997] Roberto Pieraccini, Esther Levin, and Wieland Eckert. AMICA: The AT&T mixed initiative conversational architecture. In *Eurospeech*, 1997.

[Polifroni *et al.*, 1992] Joseph Polifroni, Lynette Hirschman, Stephanie Seneff, and Victor Zue. Experiments in evaluating interactive spoken language systems. In *Proceedings of the DARPA Speech and NL Workshop*, pages 28–33, 1992.

[Pollack *et al.*, 1982] Martha Pollack, Julia Hirschberg, and Bonnie Webber. User participation in the reasoning process of expert systems. In *Proceedings First National Conference on Artificial Intelligence*, pages pp. 358–361, 1982.

[Price *et al.*, 1992] Patti Price, Lynette Hirschman, Elizabeth Shriberg, and Elizabeth Wade. Subject-based evaluation measures for interactive spoken language systems. In *Proceedings of the DARPA Speech and NL Workshop*, pages 34–39, 1992.

[Rabiner *et al.*, 1996] L. R. Rabiner, B. H. Juang, and C. H. Lee. An overview of automatic speech recognition. In C. H. Lee, F. K. Soong, and K. K. Paliwal, editors, *Automatic Speech and Speaker Recognition, Advanced Topics*, pages 1–30. Kluwer Academic Publishers, 1996.

[Ralston *et al.*, 1995] J. V. Ralston, D. B. Pisoni, , and John W. Mullennix. Perception and comprehension of speech. In Syrdal, Bennet, and Greenspan, editors, *Applied Speech Technology*, pages 233–287. CRC Press, 1995.

[Russell and Norvig, 1995] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach.* Prentiss Hall, Englewood Cliffs, N.J., 1995.

[Seneff *et al.*, 1995] Stephanie Seneff, Victor Zue, Joseph Polifroni, Christine Pao, Lee Hetherington, David Goddeau, and James Glass. The preliminary development of a displayless PEGASUS system. In *ARPA Spoken Language Technology Workshop*, 1995.

[Shriberg *et al.*, 1992] Elizabeth Shriberg, Elizabeth Wade, and Patti Price. Human-machine problem solving using spoken language systems (SLS): Factors affecting performance and user satisfaction. In *Proceedings of the DARPA Speech and NL Workshop*, pages 49–54, 1992.

[Siegel and Castellan, 1988] Sidney Siegel and N. J. Castellan. *Nonparametric Statistics for the Behavioral Sciences.* McGraw Hill, 1988.

[Simpson and Fraser, 1993] A. Simpson and N. A. Fraser. Black box and glass box evaluation of the SUNDIAL system. In *Proceedings of the Third European Conference on Speech Communication and Technology*, pages 1423–1426, 1993.

[Smith and Gordon, 1997] Ronnie W. Smith and Steven A. Gordon. Effects of variable initiative on linguistic behavior in human-computer spoken natural language dialog. *Computational Linguistics*, 23(1), 1997.

[Smith and Hipp, 1994] Ronnie W. Smith and D. Richard Hipp. *Spoken Natural Language Dialog Systems: A Practical Approach.* Oxford University Press, 1994.

[Sparck-Jones and Galliers, 1996] Karen Sparck-Jones and Julia R. Galliers. *Evaluating Natural Language Processing Systems.* Springer, 1996.

[Sproat and Olive, 1995] R. Sproat and J. Olive. An approach to text-to-speech synthesis. In W. B. Kleijn and K. K. Paliwal, editors, *Speech Coding and Synthesis*, pages 611–633. Elsevier, 1995.

[Sutton, 1991] Richard S. Sutton. Planning by incremental dynamic programming. In *Proceedings Ninth Conference on Machine Learning*, pages 353–357. Morgan-Kaufmann, 1991.

[Tesauro, 1992] G. Tesauro. Practical Issues in Temporal Difference Learning. *Machine Learning*, 8(3–4):257–277, 1992.

[Walker *et al.*, 1998a] Marilyn Walker, , Jeanne Fromer, Giuseppe Di Fabbrizio, Craig Mestel, and Donald Hindle. What can I say: Evaluating a spoken language interface to email. In *Proceedings of the Conference on Computer Human Interaction (CHI 98)*, 1998.

[Walker *et al.*, 1998b] Marilyn Walker, Jeanne Fromer, and Shrikanth Narayanan. Learning optimal dialogue strategies: A case study of a spoken dialogue agent for email. In *ACL/COLING98: Proceedings of the 36th Annual Meeting of the Association of Computational Linguistics*, 1998.

[Walker *et al.*, 1997] Marilyn Walker, Donald Hindle, Jeanne Fromer, Giuseppe Di Fabbrizio, and Craig Mestel. Evaluating competing agent strategies for a voice email agent. In *Proceedings of the European Conference on Speech Communication and Technology, EUROSPEECH97*, 1997.

[Walker, 1989] Marilyn A. Walker. Evaluating discourse processing algorithms. In *Proc. 27th Annual Meeting of the Association of Computational Linguistics*, pages 251–261, 1989.

[Walker, 1993] Marilyn A. Walker. *Informational Redundancy and Resource Bounds in Dialogue*. PhD thesis, University of Pennsylvania, 1993.

[Walker, 1996] Marilyn A. Walker. The Effect of Resource Limits and Task Complexity on Collaborative Planning in Dialogue. *Artificial Intelligence Journal*, 85(1–2):181–243, 1996.

[Walker *et al.*, 1997] Marilyn A. Walker, Diane Litman, Candace Kamm, and Alicia Abella. PARADISE: A general framework for evaluating spoken dialogue agents. In *Proceedings of the 35th Annual Meeting of the Association of Computational Linguistics, ACL/EACL 97*, pages 271–280, 1997.

[Walker and Whittaker, 1990] Marilyn A. Walker and Steve Whittaker. Mixed initiative in dialogue: An investigation into discourse segmentation. In *Proc. 28th Annual Meeting of the ACL*, pages 70–79, 1990.

[Watkins, 1989] C. J. Watkins. *Models of Delayed Reinforcement Learning*. PhD thesis, Cambridge University, 1989.

[Webber and Joshi, 1982] Bonnie Webber and Aravind Joshi. Taking the initiative in natural language database interaction: Justifying why. In *Coling 82*, pages 413–419, 1982.

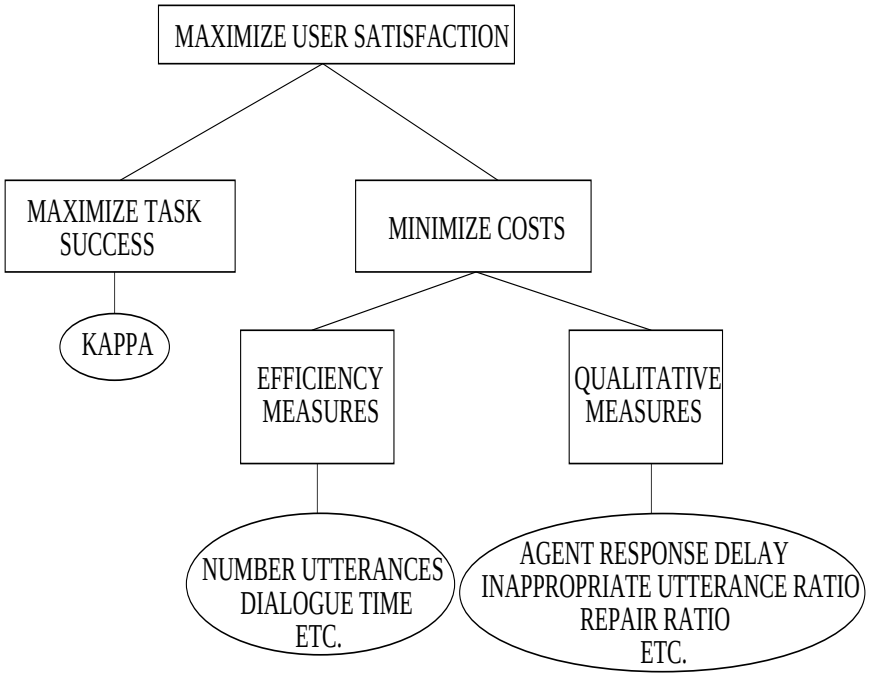

Figure 1: PARADISE's structure of objectives for spoken dialogue performance

attribute	possible values	information flow
depart-city (DC)	Milano, Roma, Torino, Trento	to agent
arrival-city (AC)	Milano, Roma, Torino, Trento	to agent
depart-range (DR)	morning,evening	to agent
depart-time (DT)	6am,8am,6pm,8pm	to user

Table 1: Attribute value matrix, simplified train timetable domain

A1:	Hello, This is Train Enquiry Service.	DC,AC,DR,DT
	Please speak after the tone.	DC,AC,DR,DT
	Which information do you need?	DC,AC,DR,DT
U1:	I want to go from Torino to Milano.	DC,AC
A2:	Do you want to go from Trento to Milano?	DC,AC
	Yes or No?	DC,AC
U2:	No.	DC,AC
A3:	Do you want to leave from Trento?	DC
	Yes or No?	DC
U3:	No.	DC
A4:	Where do you want to leave from?	DC
U4:	I want to leave from Torino.	DC
A5:	Do you want to leave from Torino?	DC
	Yes or No?	DC
U5:	Yes.	DC
A6:	Do you want to go to Milano?	AC
U6:	Yes.	AC
A7:	At which time do you want to leave?	DR
U7:	I want to travel in the evening.	DR
A8:	Do you want to leave between 6 and 9 p.m.?	DR
	Yes or No?	DR
U8:	Yes.	DR
A9:	There is a train leaving at 8:00 p.m.	DT
	...	

Figure 2: Extended Dialogue 1 (D1): Agent A dialogue interaction (Danieli and Gerbino, 1995)

B1:	Hello, This is Train Enquiry Service.	DC,AC,DR,DT
	Please speak after the tone.	DC,AC,DR,DT
	Which information do you need?	DC,AC,DR,DT
U1:	I want to travel from Torino to Milano.	DC,AC
B2:	At which time do you want to leave from Merano to Milano?	DC,AC,DR
U2:	No, I want to leave from Torino in the evening.	DC, DR
B3:	Do you want to leave from Torino between 6 and 11 p.m?	DC, DR
	Please answer Yes or No.	DC, DR
U3:	Yes.	DC, DR
B4:	A train leaves at 8 p.m.	DT
	...	

Figure 3: Extended Dialogue 2 (D2): Agent B dialogue interaction (Danieli and Gerbino, 1995)

attribute	actual value
depart-city	Torino
arrival-city	Milano
depart-range	evening
depart-time	8pm

Table 2: Attribute value matrix instantiation, scenario key for Dialogues 1 and 2

	KEY													
	DEPART-CITY				ARRIVAL-CITY				DEPART-RANGE		DEPART-TIME			
DATA	v1	v2	v3	v4	v5	v6	v7	v8	v9	v10	v11	v12	v13	v14
v1	**22**		1		3									
v2		**29**												
v3	4		**16**	4			1							
v4	1	1	5	**11**			1							
v5	3				**20**									
v6						**22**								
v7			2		1	1	**20**	5						
v8			1		1	2	8	**15**						
v9									**45**	10				
v10									5	**40**				
v11											**20**		2	
v12											1	**19**	2	4
v13											2		**18**	
v14											2	6	3	**21**
sum	30	30	25	15	25	25	30	20	50	50	25	25	25	25

Table 3: Confusion matrix, Agent A

	KEY													
	DEPART-CITY				ARRIVAL-CITY				DEPART-RANGE		DEPART-TIME			
DATA	v1	v2	v3	v4	v5	v6	v7	v8	v9	v10	v11	v12	v13	v14
v1	**16**		1		4				3	2				
v2	1	**20**	1			3								
v3	5	1	**9**	4	2		4	2						
v4	1	2	6	**6**			2	3						
v5	4				**15**				2	3				
v6	1	6				**19**								
v7			5	2	1	1	**15**	4						
v8		1	3	3	1	2	9	**11**						
v9	2				2				**39**	10				
v10									6	**35**				
v11											**20**	5	5	4
v12												**10**	5	5
v13											5	5	**10**	5
v14												5	5	**11**
sum	30	30	25	15	25	25	30	20	50	50	25	25	25	25

Table 4: Confusion matrix, Agent B

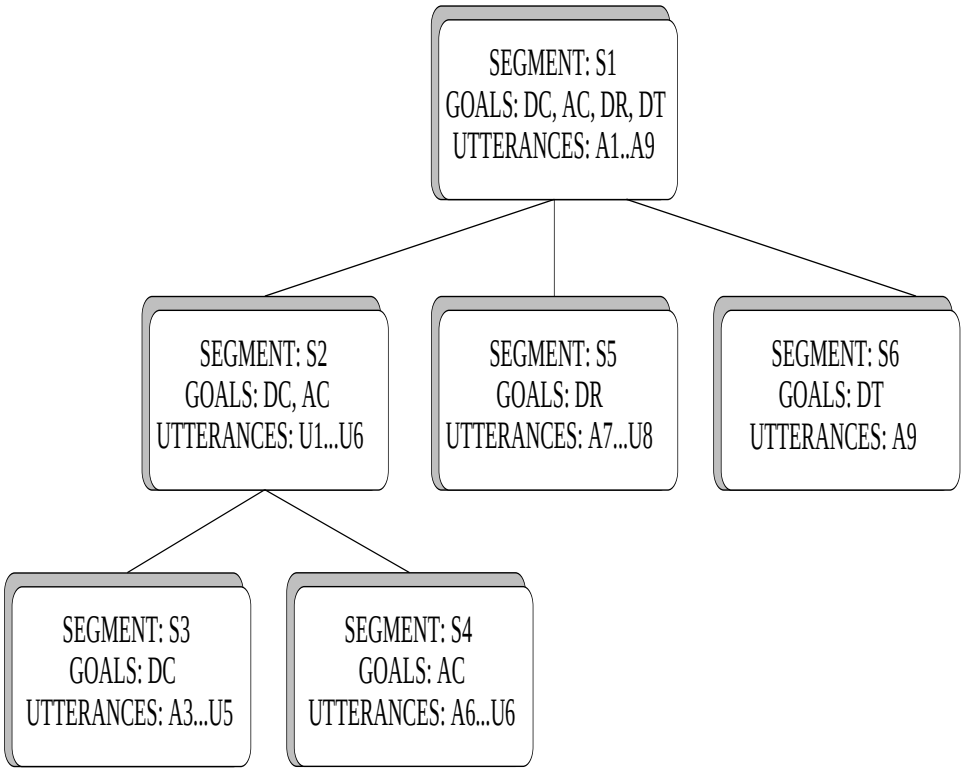

Figure 4: Task-defined discourse structure of Agent A dialogue interaction

user	agent	US	κ	c_1 (#utt)	c_2 (#rep)
1	A	1	1	46	30
2	A	2	1	50	30
3	A	2	1	52	30
4	A	3	1	40	20
5	A	4	1	23	10
6	A	2	1	50	36
7	A	1	0.46	75	30
8	A	1	0.19	60	30
9	B	6	1	8	0
10	B	5	1	15	1
11	B	6	1	10	0.5
12	B	5	1	20	3
13	B	1	0.19	45	18
14	B	1	0.46	50	22
15	B	2	0.19	34	18
16	B	2	0.46	40	18
Mean(A)	A	2	0.83	49.5	27
Mean(B)	B	3.5	0.66	27.8	10.1
Mean	NA	2.75	0.75	38.6	18.5

Table 5: Hypothetical performance data from users of Agents A and B

attribute	actual value
Selection Criteria	Kim $\vee$ Meeting
Email.att1	10:30
Email.att2	2D516

Table 6: Attribute value matrix instantiation, email scenario key for TASK 1.1

user	task	agent state/strategy choices	US	κ	ET	STs	TOs	MRS	.....
1	1	S_1:$A_{1,1}$, S_2:$A_{2,1}$, ...	24	1.0	220	18	15	.90	.....
1	2	S_1:$A_{1,1}$, S_2:$A_{2,1}$, ...	27	.80	180	24	16	.79	.....
1	3	.....							.....
.....									.....
48	3	.....							.....

Table 7: Form of Logged Information about Strategy Choices and Performance Metrics for a PARADISE Case Study: US = User Satisfaction, ET = Elapsed Time, Sts = System Turns, TOs = Timeout Prompts, MRS = Mean Recognition Score

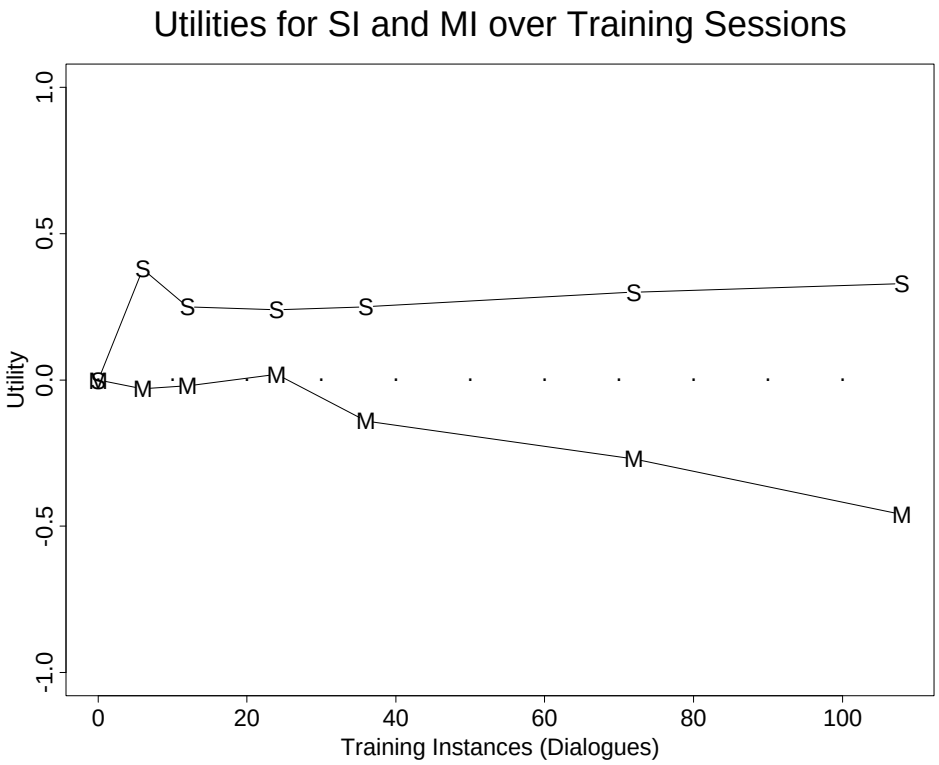

Figure 5: Results of applying Adaptive Dynamic Programming to ELVIS data for 108 Dialogues. S=Utility of System Initiative Strategy (SI). M=Utility of Mixed Initiative Strategy (MI).

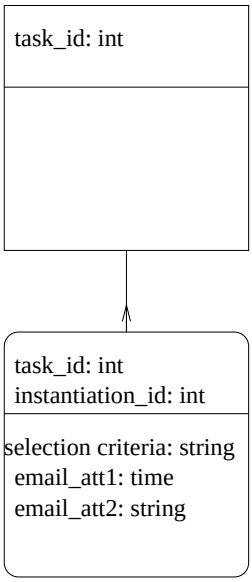

Figure 6: Scheme for the email scenario key for TASK 1.1

task-id
1

task-id	instantiation-id	selection criteria	Email.att1	Email.att2
1	1	Kim	10:30	2D516
1	2	Meeting	10:30	2D516

Figure 7: Scheme instantiation for the email scenario key for TASK 1.1