Computer-Aided Analysis and Design for Spoken Dialogue Systems Based on Quantitative Simulations

Bor-shen Lin and Lin-shan Lee, *Fellow, IEEE*

***Abstract*—Corpus-based analysis and design of spoken dialogue systems have been widely used. However, in such approaches the dialogue performance cannot be predicted before the system is on line, and the dialogue corpora need to be recollected if the system is modified or different conditions are assumed. Also, the effects of different factors, from the system's dialogue strategies, speech recognition and understanding conditions and accuracy, to the user's response pattern, etc., on the dialogue system performance cannot be quantitatively identified and analyzed, because they cannot be precisely controlled in different corpora.**

In this paper, a complete development of computer-aided analysis and design approaches for spoken dialogue systems based on quantitative simulations is presented. With this approach the various performance metrics of a dialogue system can be flexibly defined and numerically evaluated, such that the behavior and performance of the dialogue system can be well predicted and efficiently analyzed before the implementation of the real spoken dialogue system is completed. How the different dialogue performance measures vary with respect to each of the many very complicated factors, regardless of whether it is caused by an individual component, by the overall system design, or by users' response patterns, can be separately identified, because all such factors can be precisely controlled in the simulation. Several analysis examples are presented to show how the approach can be used, including selection and tuning of speech understanding front end, system strategy design considering query factors and confirmation factors, and objective estimates of user's degree of satisfaction. This approach is therefore very useful for the analysis and design of spoken dialogue systems, although the online test, corpus-based analysis and user survey can always follow after the system is online.

***Index Terms*—Computer simulation, dialogue strategy, dialogue system design, performance analysis, spoken dialogue.**

I. Introduction

SPOKEN dialogue systems are typically developed by building a series of prototype systems, where the improvements from one system to the next are made by collecting a corpus of dialogues between users and the prototype to see how it performed [1]–[8]. The performance can be assessed in terms of relatively simple metrics such as number of dialogue turns or transaction success rate, or more sophisticated analysis such as confusion matrices for keywords [1]. Because the primary assessments are based on analyzing a large corpus of real human-machine dialogues, there exist some difficulties for such design approaches. One major difficulty is that the performance of a dialogue system cannot be estimated before the system is on line, and very little about the dialogue performance can be known before many people get involved in the online test. Some further difficulties also occur even if the prototype system can be on line. For example, when any of the system dialogue strategies are modified, not only the whole analysis based on the previous strategy is no longer valid, but the dialogue corpus collected previously becomes useless for the analysis of new strategies because the new analysis needs to be based on new interactions. As a result, both the data collection and corpus analysis need to be repeated from the scratch after each modification, as long as the overall dialogue performance is to be analyzed. This makes the cycle for testing and modifying spoken dialogue systems relatively long. In addition, it is not rigorous nor reliable enough to compare the dialogue performance for different system strategies simply using the different corpora collected under different conditions, because many other factors existing in the processes of producing the corpora, such as the user's response pattern, the slot accuracy or word error rate, cannot be precisely controlled to be exactly identical. All these difficulties are intrinsic in such design and analysis approaches based on analyzing human-machine dialogues for prototype systems.

Another well-known approach, i.e., the "Wizard of Oz" test, has been developed to assist the design and test of spoken dialogue systems [9], [10]. This approach includes a human being in the process, and is helpful to obtain some insight into the user's dialogue behavior or initial dialogue system performance before the prototype system is accomplished. A large corpus of spontaneous speech for the desired task can be collected in the process as well for training and evaluating the speech recognition and understanding modules. However, the cost of such approach is relatively high, because not only a human being is involved in the process, but the data collection takes time. Another approach, of course, is to gain insight into the dialogue behavior by directly observing or analyzing a corpus of human-human dialogues without building any prototype system. How human beings really behave in the real world is definitely a good guide for designing spoken dialogue systems. However, human-human dialogue corpus is not always available, specially for dialogue tasks intended for new services. Also, in natural human-human dialogues many factors cannot be controlled at all, and the interactions within human-human dialogues are certainly quite different from those in human-machine dialogues.

Qualitative analysis approaches based on subjective evaluations from user survey have also been widely used [7], [8], [11]. However, very often the data obtained in such approaches are the results of general feeling due to many relevant factors. For example, when 20% of users are not satisfied with the system, their

Manuscript received September 1, 2000; revised January 19, 2001. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Andreas Stolcke.

The authors are with the Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan, R.O.C. (e-mail: lsl@iis.sinica.edu.tw).

Publisher Item Identifier S 1063-6676(01)05545-6.

dissatisfaction may be due to many different reasons at different degrees. These analyses are helpful to some extent, but do not necessarily indicate precisely the right directions for improvements. Some very useful design principles obtained empirically were also proposed [12], [13]. However, human experiences are typically not precise enough, and it is thus not easy to estimate objectively and quantitatively the price paid and the gain obtained for each principle.

On the other hand, some dialogue analysis approaches completely based on analytical or mathematical models without using any online corpus have been reported [14], [15]. Such studies are always highly desired. In some cases, such metrics as dialogue turns for different dialogue strategies can even be expressed in closed forms. However, in order to achieve analytical solutions, usually some simplifying assumptions have to be made. It is thus not easy to extend such analytical solutions to all desired cases for many complicated design options for dialogue systems.

In these discussions, it is clear that computer-aided analysis and design for spoken dialogue systems based on detailed quantitative simulations are very attractive, because much of the analysis can be performed without the corpus before the system is completed, and many of the factors such as the user's response pattern and word error rate can be precisely controlled. The choice of different system strategies can also be made numerically. Some of such simulation methods have been previously proposed [16]–[20], and the benefits of using simulation have been discussed in great details [17], [18]. In some cases, a simulated user based on a bigram model was used to interact with a dialogue system on the intention level using some assumed data, but the effect of recognition or understanding errors was not well considered [16], [17]. In some other cases, dialogue was simulated on the text level with some deletion errors considered [18], but such simulation was performed only for a specific task of shortest route search in the literature. There was another work of simulation of dialogues on the text level even with sophisticated user patterns, but the approach seems to be specific to the particular task [19]. There was also other approach with users' behavior based on a predefined user's goal, but the discussions on the system's strategies were dismissed [20]. In other words, all the prior simulation frameworks were somewhat incomplete, and more or less constrained to the specific dialogue task being studied, at least not easily extensible to other tasks. Also, in most cases, complete in-depth and systemized analysis seems to be missing yet. In this paper, one such simulation-based approach with complete analyses and discussions is presented. The basic methodologies and principles of the proposed approach are very simple, general, and generic, and therefore quite flexible and extensible to many different conditions. With this scheme, the various performance measures of a spoken dialogue system can be flexibly defined and numerically evaluated, such that the behavior and performance of the system can be estimated appropriately and analyzed efficiently before the implementation of prototype system is completed. How the various performance measures vary with respect to each factor, from recognition accuracy to dialogue strategies, can be individually identified in advance, because all such factors can be precisely controlled in the simulation. The quality of service or the user's degree of satisfaction for spoken dialogue systems can also be flexibly defined and efficiently estimated by simulation. Selection and tuning of the speech understanding front end, tradeoff among performance goals such as accuracy and efficiency, and design of system strategies in dialogue flow are all practically feasible and can be numerically determined. All these are illustrated by examples in this paper. This approach is therefore very useful for the design and analysis of spoken dialogue systems. The online test, corpus-based analysis, and user survey can always follow and be very helpful after the system is completed.

Of course, it should be pointed out here that there are natural limitations for any simulation-based approach. The simulation results can never be better than what can be said by the model that the simulation is based on. For example, in the approach presented in this paper, the speech recognition and understanding errors are modeled by long-term statistics of slot errors, which is definitely not good enough in describing some specific speech understanding conditions in the real world. Over-simplified models for user behavior are also used here, which is for sure inadequate in modeling the real users. As will be shown in Section VII, finer models with more parameters are always possible, but the success of such models still rely on whether they can really describe the real situations. In other words, the key point is in fact whether the model used is good enough for the practical purposes, which is not really answered in this paper.

The rest of the paper is organized as follows. Section II describes the complete quantitative simulation approach, including statistical analysis of different parameters and performance metrics. A series of analysis examples are then given in Sections III–VI. Section III presents the selection and tuning of speech understanding front end for some performance goal. In Section IV and Section V, the design of different systems' strategies considering query factors, confirmation factors and users' response patterns are discussed in great detail. Section VI shows how objective estimates of the user's degree of satisfaction can be obtained based on the proposed approach. In Section VII, some possible approaches to extend to much more complicated dialogue scenario than the simplified schemes used in the examples are discussed. Section VIII finally summarizes the contributions of this paper.

II. Quantitative Simulation Approach

A. State Representation

In the proposed approach, a dialogue is modeled as the processes that a set of semantic slots is transmitted from the user to the system. The finite state machine for each semantic slot, s_i, is first represented in a two-tuple expression, as shown in Fig. 1. The first argument denotes the state of the semantic slot as unknown (u), known but not yet verified (k), or verified (v), while the second argument denotes the correctness of the slot value as correct (c) or error (e), and when a slot is unknown, the correctness of its value is meaningless (x). Assuming that there are a total of n semantic slots necessary for a transaction, the overall dialogue state S can therefore be represented as n finite state machines, that is

$$S = (s_1, s_2, \ldots, s_n). \tag{1}$$

With this definition, the initial state S_i of the overall system is then

$$S_i = (s_1 = (u, x), s_2 = (u, x), \ldots, s_n = (u, x)) \quad (2)$$

while the final state S_f is

$$S_f = (s_1 = (v, y), s_2 = (v, y), \ldots, s_n = (v, y)) \quad (3)$$

where the symbol y can be either correct or error. The purpose of the dialogue is therefore to make each of the finite state machines to transit from the state (u, x) to the state (v, y) as shown in Fig. 1, such that the overall state S may transit from the initial state S_i to the final state S_f. A successful transaction then occurs when all the semantic slots in the final state are correctly verified, i.e.,

$$S_f = (s_1 = (v, c), s_2 = (v, c), \ldots, s_n = (v, c)). \quad (4)$$

How these states actually transit is determined by the simulation scheme given below.

B. Simulation Scheme

The simulation scheme can be represented with the pseudocodes as shown in Fig. 2. The cycle of a dialogue turn can be simulated by four segments: system's prompt, user's response, speech understanding, and system's update, as shown within the for-loop in Fig. 2. In the "system's prompt" segment, how the system decides which slots should be queried and which slots should be confirmed is simulated. In the "user's response" segment, how the user decides to respond to the system's prompt is simulated. The schemes simulated in these two segments are referred to as "system's prompt strategy" and "user's response pattern," respectively. In the "speech understanding" segment, the slots can be considered as being transmitted from the user to the system through an unreliable channel, and the effect of speech recognition and understanding errors is simulated as transmission errors that influence the actually received slots. The model used in this segment is therefore referred to as "channel effect" in this paper. In the "system update" segment, how the system controls the state transition is simulated. The scheme simulated in this segment is referred to as "system's update strategy" in this paper. For example, the system may decide that all slots being confirmed are verified (from (k, y) to (v, y)) based on the condition that a "yes" is detected or the condition that these slots are consistent with the previously received slots. With the simulation scheme previously described, the dialogue performance is a function of four sets of parameters, the system's prompt strategy S_P, the user's response pattern U, the channel effect C and the system's update strategy S_U

$$P_D = F(S_P, U, C, S_U) \quad (5)$$

where P_D can be any set of metrics for dialogue performance.

C. Speech Understanding or Channel Effect

Conventionally, the speech understanding errors are often measured by slot error rate [21], which includes the rates

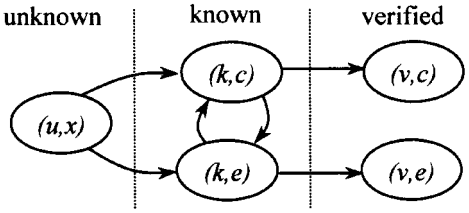

Fig. 1. Finite state machine for each semantic slot.

```
for(N_l=0;!Goal();N_l ++) {
   SystemPrompt();  // system's prompt strategy
   UserResponse();   // user's response pattern
   SpeechUnderstanding();//understanding performance
   SystemUpdate();   // state transition
}
if(AllSlotCorrect()) T_s = 1;    // successful transaction
else T_s = 0                     // error transaction
E_s = n/n';    // slot transmission efficiency
```

Fig. 2. Pseudo codes for simulation of a dialogue.

for inserted, deleted, and substituted slots, R_{ins}, R_{del}, and R_{sub}, respectively. The inserted slots are those causing misunderstanding and therefore regarded as "misunderstanding slots" here, while the deleted slots are those lost in the slot transmission channel and regarded as "lost slots" here. In this way, each substituted slot can be considered as an inserted slot plus a deleted slot, or a misunderstanding slot plus a lost slot. The understanding error can therefore be represented by the following two parameters:

$$R_m = R_{ins} + R_{sub}, \quad R_l = R_{del} + R_{sub} \quad (6)$$

where R_m is the slot misunderstanding rate and R_l is the slot lost rate, both including the case of substituted slots. As a result, for each slot transmitted by the user, two error events may occur. One is that the transmitted slot may be lost with probability R_l, and the other is that some other undesired slot may be received with probability R_m. The "channel effect" segment can therefore be simulated using random tests defined by these two parameters R_l and R_m, as shown in Fig. 3. When $R_m = R_l = 0$, the channel is error-free and the simulation results account for the text-mode dialogue.

It should be pointed out that the real channel effect for a spoken dialogue system is a much more sophisticated random process than the simplified model as given here, depending on the sentence generation and pronunciation processes of the user and the recognition and understanding processes of the system. The characteristics of the channel effect may therefore have to do with the recognition or understanding front ends, the speakers, the speaking mode and speed, the kinds of slots, the background environments and so on. Also, the true understanding accuracy may be different for different slots. As a result, an ideal simulation for the channel effect can be very difficult, and depend on many conditions. The channel effect model proposed here in Fig. 3 is a simplified, "unbiased" model, independent of the many conditions. Of course, when any of these conditions are to be considered, more complicated models with more parameters can always be constructed in a similar way to take into account those conditions. Furthermore,

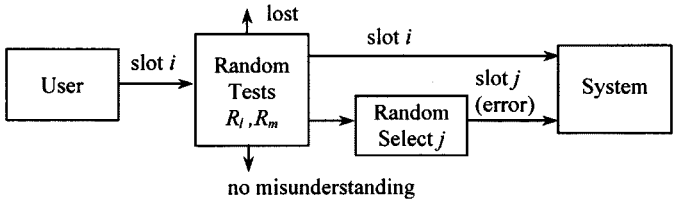

Fig. 3. Channel effect for speech understanding.

the random tests for the lost slot and misunderstanding slot in Fig. 3 can be correlated with additional correlation coefficient that may be dependent of the many conditions mentioned above sophisticatedly. For simplicity, in this paper the "uncorrelated" channel effect model proposed in Fig. 3 is used for simulation. However, if some specific correlation effect is to be investigated, it can be easily taken into account in Fig. 3.

D. Systems' Strategies and Users' Response Patterns

There are still other factors, i.e., the system's strategies and user's response patterns in the other three segments in Fig. 2, which may significantly influence the dialogue performance just as the channel effect. Because the goal of dialogue here is to have the finite state machines for all required semantic slots transit from the unknown state, through the known state, into the verified state, the factors in these three segments can therefore be classified according to their major effects on the state transition. Those factors primarily influencing the state transition from the unknown state to the known state are referred to as "query factors" here, while those primarily influencing the state transition from the known state to the verified state as "confirmation factors." For example, the former includes how the system queries among those slots in the unknown state, how the user responds given the queried slots, and how the system updates the states according to the received slots, while the latter includes how the system prompts among those slots in the known state for confirmation, how the user responds given those slots to be confirmed, and how the confirmation is accomplished based on the received slots, and so on. Although most of such factors are difficult to parameterize, it is possible to use simplified models to specify these factors. Some examples for such simplified models are presented below for illustration purposes, and it will be shown later on that more complicated situations can always be extended from these simplified models.

For the parameter sets in (5) for the other three segments, the system's prompt strategy S_P, the user's response pattern U and the system's update strategy S_U, each set can be divided into two subsets, one for query factors and the other for confirmation factors. For example, the simplest model for the system's prompt strategy S_P may be

$$S_P = (AQ, AC) \tag{7}$$

which means all slots in the unknown state are queried [all queried (*AQ*)], while all slots in the known state are prompted for confirmation [all confirmed (*AC*)]. The former specifies the query factors, while the latter the confirmation factors. Of course this is an over-simplified model, but more sophisticated strategies can always be modeled and simulated in the same way. Similarly for the user's response pattern U, a simplified model may be

$$U = (AR/NQNT, YC). \tag{8}$$

The first part means all queried slots are replied to the system [all replied (*AR*)], but those unknown slots not queried are not transmitted [not queried not transmitted (*NQNT*)]. For those slots to be confirmed, on the other hand, the second part of (8) means that a "yes" is transmitted if all correct, otherwise incorrect slots are retransmitted with "no" [yes if correct (*YC*)]. For the system's update strategy S_U, a simplified model may be

$$S_U = (KR/SI, VSC). \tag{9}$$

This means all queried slots in the unknown state will enter the known state if they are received [known state if received (*KR*)], while those slots in the known state being confirmed will enter the verified state if they are all received consistently [verified by slot consistency (*VSC*)]. All other slots are not updated, or system-initiative (*SI*).

E. Fundamental Statistical Analysis

For the simulation of each dialogue, after the four segments are iterated for enough number of times, the final state can be achieved and the dialogue terminated, as shown in Fig. 2. Various characteristic parameters of the dialogue can then be extracted. In the following are some examples. First, the transaction success flag, T_s, which equals one if a successful transaction is achieved and zero if not, can then be determined by checking the second argument of the finite state machines, i.e., to see if $S_f = (s_1 = (v, c), s_2 = (v, c), \ldots, s_n = (v, c))$. Second, the number of dialogue turns, N_t, can be obtained in the for-loop in Fig. 2. Furthermore, the slot transmission efficiency, E_s, can be defined as

$$E_s = \frac{n}{n'} \tag{10}$$

where n' is the total number of transmitted slots, and can be observed in the "user's response" segment. This slot transmission efficiency E_s indicates whether the user can transmit the slots efficiently, whose value ranges from zero to one. For example, if E_s is 50%, this means $n' = 2n$, or each slot has to be transmitted twice in average in order to complete the dialogue, which may be very boring for the user.

The above example characteristic parameters, T_s, N_t and E_s, are all random variables, whose samples can be extracted after each dialogue is completed. After the simulation is performed for a large number of dialogues, the mean values of these random variables, $\bar{T}_s$, $\bar{N}_t$ and $\bar{E}_s$, can be estimated. They are, respectively, the transaction success rate, the average dialogue turns, and the average slot transmission efficiency, all very useful in analyzing the dialogue performance. In fact, not only the mean values of these random variables are obtainable, but the complete distributions of them, $P(T_s)$, $P(N_t)$, and $P(E_s)$, are available after the simulation. Many other parameters such as the variance for each random variable can also be readily estimated.

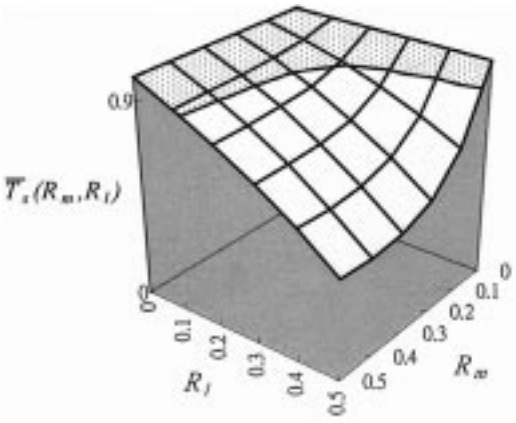

Fig. 4. Transaction success rate as a function of R_m and R_1 (dotted area: $\bar{T}_s > 0.9$).

As an example for illustration, for a dialogue system with the simplified models S_P, U, S_U defined in (7)–(9) how the transaction success rate $\bar{T}_s$ decreases with the degradation of the speech understanding performance is shown in Fig. 4, where the number of semantic slots, n, is taken to be five, and for each (R_m, R_l) pair, 100 000 dialogues were simulated. This simplified dialogue system will be referred to as the "baseline system." It can be found in Fig. 4 that, though the slot misunderstanding rate R_m and slot lost rate R_l both degrade the dialogue performance, their effects are actually different. R_m is the source of incorrect slots, while R_l is the source of slot retransmissions and incorrect verifications due to lost correcting slots. The more detailed variation of the transaction success rate with respect to individual R_l and R_m is further shown in Fig. 5(a) and (b), respectively. It can also be observed in Fig. 5(a) that when R_l increases, the transaction success rate degrades persistently because the rate of correctly received slots, $(1 - R_l)$, decreases. However, in Fig. 5(b), the degradation of the transaction success rate eventually saturates with increased R_m in this simplified model. This is because the system's update strategy *VSC* in S_U in (9), is more robust and less sensitive for higher misunderstanding rates, since for large R_m the further incorrectly received slots will be identified and will not cause further performance degradation. Here the different effects of R_l and R_m on the transaction success rate show that the proposed quantitative simulation approach can provide more insights about the dialogue systems. Similar simulations and analyses for other performance metrics can be performed in the same way.

F. Analysis with State Transition Probabilities

In addition to the fundamental statistical analysis mentioned above, here we will show that the state transition probabilities for the finite state machine shown in Fig. 1 easily obtained in the simulated dialogues can also provide very useful characteristics for the dialogue. First, the state transition probabilities can be defined as in Fig. 6(a) by five parameters, c, p, q, r, and s, where c is the transition probability from (u, x) to (k, c), p and q are those from (k, c) to (k, e) and (v, c), respectively, and r and s from (k, e) to (k, c) and (v, e), respectively. Since the transitions from (k, c) to itself (with probability $1 - p - q$) and from (k, e) to itself (with probability $1 - r - s$) do not influence the accuracy of the dialogue, the state transition diagram in Fig. 6(a) can be reduced for simplicity to Fig. 6(b) with new parameters p' and s'

$$p' = \frac{p}{p+q}, \tag{11}$$

$$s' = \frac{s}{r+s} \tag{12}$$

if only the accuracy is concerned. Now a very important probability can be defined

$$P[(v,e)] = \text{prob}[\text{any slot in } (u,x) \text{ finally enters } (v,e)]. \tag{13}$$

Apparently, the higher this probability $P[(v, e)]$ is, the lower the accuracy or transaction success rate of the dialogue will be. This probability can be obtained by the following two probabilities:

$$P_1 = \text{prob}[\text{any slot in } (k,c) \text{ finally enters } (v,e)] \tag{14}$$

$$P_2 = \text{prob}[\text{any slot in } (k,e) \text{ finally enters } (v,e)]. \tag{15}$$

One can have the following relations from Fig. 6(b):

$$P_1 = p' P_2, \tag{16}$$

$$P_2 = (1 - s')P_1 + s'. \tag{17}$$

Equations (16) and (17) lead to the following solution:

$$P_1 = \frac{s' \cdot p'}{1 - p'(1 - s')} \tag{18}$$

$$P_2 = \frac{s'}{1 - p'(1 - s')} \tag{19}$$

and the probability $P[(v, e)]$ is simply

$$P[(v,e)] = cP_1 + (1-c)P_2 = \frac{(cp' + 1 - c)s'}{(1 - p'(1 - s'))}. \tag{20}$$

In (20), $P[(v, e)]$ actually increases with the increase of both p' and s' as plotted in Fig. 7, which is also intuitively reasonable. As will be shown later on in the examples below, the increase of $P[(v, e)]$ directly degrades the transaction success rate, and Fig. 6, (20) and the various probabilities will be very useful in analyzing dialogue systems. In fact, when all the finite state machines for the semantic slots of a dialogue system are independent with the same state transition probabilities, the closed form for the transaction success rate $\bar{T}_s$ can be easily obtained from $P[(v, e)]$ in (20)

$$\bar{T}_s = (1 - p[(v,e)])^n \tag{21}$$

where n is the total number of slots. However, in the "baseline system" mentioned in Section II-E, the condition of independency among the finite state machines for all slots does not hold because the state transitions for all slots to be confirmed in the strategy *VSC* are simultaneously decided together, and thus (21) cannot be applied.

G. Quality of Service and Operating Region

Just as in many other areas, the "quality of service" (QOS) for the spoken dialogue systems can be defined as the probability that a user may acquire the service above some minimum

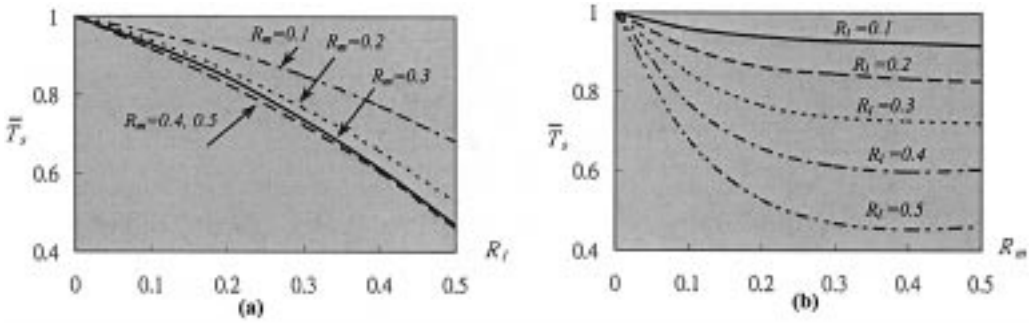

Fig. 5. Transaction success rate versus (a) $R_l(R_m = 0.1, \ldots 0.5)$ and (b) $R_m(R_l = 0.1, \ldots, 0.5)$.

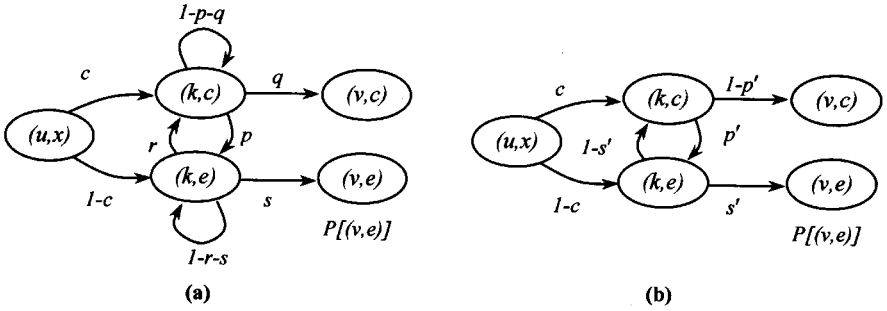

Fig. 6. (a) State transition diagram for semantic slot and (b) effective state transition diagram if only accuracy is concerned.

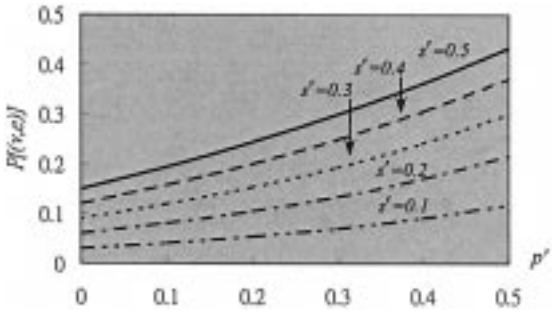

Fig. 7. $P[(v, e)]$ as a function of p' and s' at $c = 0.7$.

acceptable criteria. For example, with the characteristic parameters defined above, two possible metrics of QOS for spoken dialogue systems can be defined as follows:

$$P_{nt} \equiv Pr\,(N_t < N_t^*) \tag{22}$$

$$P_{es} \equiv Pr\,(E_s > E_s^*)\,. \tag{23}$$

The first metric P_{nt}, or probability of acceptable number of turns, is the probability that a user may complete the transaction within a predefined maximum acceptable number of dialogue turns, N_t^*. The second metric P_{es}, or probability of acceptable slot transmission efficiency, is the probability that a user may complete the transaction with slot transmission efficiency above a predefined minimum acceptable value, E_s^*. Because all performance metrics (including $\bar{T}_s$, $\bar{N}_t$, $\bar{E}_s$, P_{nt}, and P_{es}) are functions of the channel effect, specified by the two parameters R_m and R_l; it is therefore easy to obtain these metrics as functions defined on a (R_m, R_l) plane. If some performance goal is set, say $G = \{P > P^*\}$ where P is any performance metric and P^* is the desired value, the region on (R_m, R_l) plane within which the desired goal G can be satisfied, referred to as the operating region here, can be directly obtained by numerical evaluation. For example, in Fig. 4 $\bar{T}_s$ as a function of (R_m, R_l) is shown, and the dotted area is the region for the goal $G = \{\bar{T}_s > 0.9\}$, whose projection on the (R_m, R_l) plane, as shown more clearly in Fig. 8, is the operating region for the specific performance goal. In this way, it is possible to determine not only whether a speech recognition/understanding front end is capable of achieving some performance goal, but also to what extent or in which direction this front end should be tuned or improved so as to meet the performance goal. Furthermore, the performance goal for a spoken dialogue system can be defined flexibly as arbitrary combinations of several conditions for different performance metrics, such as $G = \{P_1 > P_1^*, P_2 > P_2^*, \ldots, P_N < P_N^*\}$ where $P_1, P_2, \ldots, P_N$ are the chosen performance metrics, and the corresponding operating region can be derived accordingly.

It should be pointed out that the concept of operating region can be applied not only to the speech understanding or channel effect segment, but in fact equally applicable to other segments in Fig. 2 including the system's prompt strategy, the user's response pattern, and the system's update strategy as well, as long as the simulation models for those segments can be numerically characterized by parameters like R_m and R_l in the case of channel effect mentioned above. When the simulation model is specified by more than two parameters, the operating region can be obtained similarly, except on a multidimensional space. Furthermore, many other performance metrics based on dialogue behaviors can also be used in the above analysis. For example, the users may be bored by speaking the same slots repeatedly or conversing for too many turns, and thus give up the dialogue and hang up the phone. Such events can be similarly modeled as performance metrics and analyzed by operating regions as well.

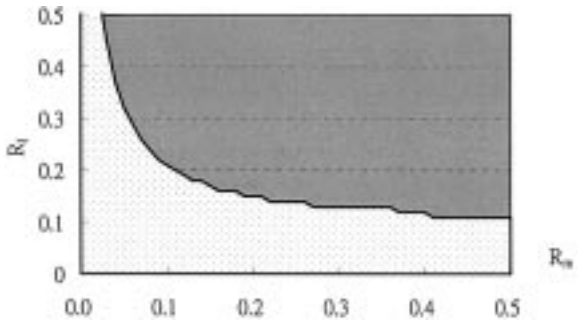

Fig. 8. Operation region for the performance goal $\bar{T}_s > 0.9$ for the baseline system.

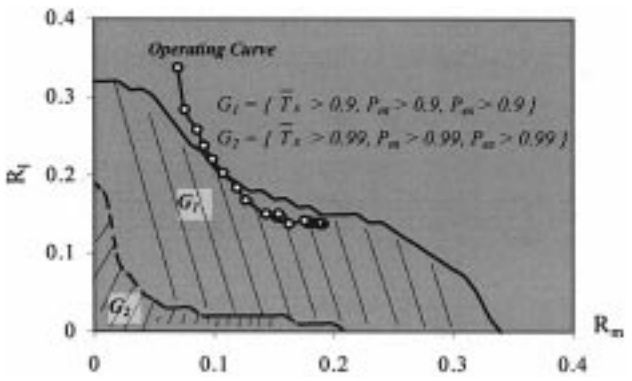

Fig. 9. Operating regions for different performance goals and the operating curve of an example speech understanding front end.

III. Analysis Example 1: Selection and Tuning of Speech Understanding Front End

Here, an example of dialogue analysis for selection and tuning of speech understanding front end is presented. Assume the "baseline system" mentioned in Section II-E is considered. So total number of semantic slots is five, the simplified models for S_P, U, and S_U as defined in (7)–(9) are used, and the channel effect is specified by (R_m, R_l). Assume the maximum acceptable number of turns, N_t^* in (22), is five and the minimum acceptable slot transmission efficiency, E_s^* in (23), is 0.5. Assume two sets of performance goals: $G_1 = \{\bar{T}_s > 0.9, P_{nt} > 0.9, P_{es} > 0.9\}$ and $G_2 = \{\bar{T}_s > 0.99, P_{nt} > 0.99, P_{es} > 0.99\}$, where P_{nt} and P_{es} are the probabilities for acceptable number of turns and for acceptable slot transmission efficiency as defined in (22) and (23). The operating region for the two performance goals G_1 and G_2 were then obtained numerically and shown as the left-bottom area of the respective curves in Fig. 9. As an example, taking a speech understanding front end in Mandarin Chinese with a task of train ticket reservation based on key-phrase spotting and a hierarchical tag-graph search scheme [22], the operating curve of this front end can be derived by tuning the threshold values in the key-phrase spotter and updating the spotting rates, which is also plotted in Fig. 9. It can be found that some part of the curve is within the operating region of G_1 , but some part of it is not. It is therefore possible to tune this speech understanding front end such that it can provide the desired performance goal G_1. However, it can also be observed that it is actually impossible for this front end to achieve the performance goal G_2 in any case by simply tuning the threshold values and spotting rates. Such analysis therefore gives a very good direction for selection and tuning of speech recognition front ends for designing spoken dialogue systems.

IV. Analysis Example 2: Considerations for Query Factors

As described in Section II-D, the simplified models for the query factors include the system's prompt strategy *AQ* as defined in (7), the user's response pattern [all replied/not queried not transmitted (*AR/NQNT*)] as in (8), and the system's update strategy [known state if received/system initiative (*KR/SI*)] as in (9), which primarily affect the state transition from the unknown state to the known state. Of course a spoken dialogue system does not have to use such simplified models. How the dialogue performance is influenced by each of these factors if more complicated models are used is considered in this section with an example, in which all other factors are the same as the "baseline system" mentioned in Section II-E with the simplified models defined in Section II-D if not specially mentioned. The considerations for the confirmation factors will be discussed in the next section.

A. System's Prompt Strategy

When the system prompts the user with unknown slots, it may ask for all unknown slots, just as *AQ* in the simplified model. But the system may also ask for only a part of the unknown slots [part queried (*PQ*)], or only one unknown slot at a time [one queried ($1Q$)]. In Fig. 10(a)–(c), the average dialogue turns, the transaction success rates and the average slot transmission efficiency are plotted for $R_l = 0.1$ as functions of R_m for the example system for these three system's prompt strategies: *AQ*, *PQ*, and $1Q$. In the case of *PQ*, the queried slots are randomly selected. The user's response pattern assumed is *AR*, or users who strictly follow system's guidance. It can be found from Fig. 10(a)–(c) that under such user's response pattern, prompting more unknown slots (e.g., *AQ*) leads to lower average dialogue turns and higher transaction success rates, but lower average slot transmission efficiency. The results of Fig. 10(a) can be easily understood. To achieve the user's goal with fewer dialogue turns, the finite state machines for the semantic slots should transit from the unknown state to the known state as soon as possible, and the system therefore had better prompt the user with as many unknown slots as possible. The results of Fig. 10(b), on the other hand, are primarily due to the system's prompt strategy for confirmation, *AC*, as defined in (7), and the system's update strategy for confirmation, *VSC*, as defined in (9). When all unknown slots are prompted (*AQ*), they tend to enter the known state earlier, and the probability of confirming more slots at a time will be higher. With *VSC*, higher transaction success rate can be achieved if more slots are confirmed at a time, which will be discussed in detail in Section V-A. This is why the transaction success rate of *AQ* is higher in Fig. 10(b). However, there is no free lunch. As will be seen in Section V-A, for *VSC* the higher transaction success rate is in fact obtained at the cost of more rejections during confirmations, which inevitably degrade the average slot transmission efficiency as shown in Fig. 10(c). Of course, it

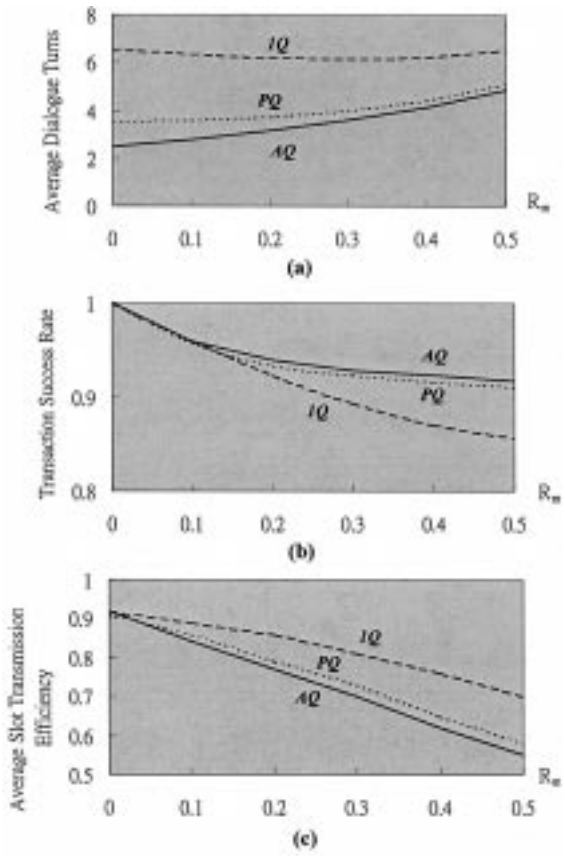

Fig. 10. (a) Average dialogue turns, (b) transaction success rates, and (c) average slot transmission efficiency for $R_l = 0.1$ plotted as functions of R_m for three different system's prompt strategies: *AQ*, *PQ*, and $1Q$.

should be pointed out that prompting too many slots within an utterance might confuse the user (e.g., prompting four or more unknown slots at a time), and in such case the user may simply reply only a part of them. Such conditions are not considered here, though can definitely be simulated.

B. User's Response Pattern

In Fig. 10(a), it was found that a system's prompt strategy *AQ* achieves better average dialogue turns given the user's response pattern *AR*. However, this is not always true for different kinds of users. Fig. 11 shows the average dialogue turns for different users' response patterns given the system's prompt strategy *AQ* and $R_l = 0.1$, where one replied ($1R$) denotes the user's pattern that only one of the queried slots is replied, part replied (*PR*) denotes that only part of the queried slots are replied, and *AR* (all replied) is the original user's pattern. As can be found in Fig. 11, different behavior of the user may lead to quite different performance, approximately ranging from the curve for *AR* to that for $1R$. In other words, a good system's prompt strategy *(AQ)* does not always lead to the desired good dialogue performance for all users. The dialogue is the interaction between two parties, the system and the user. The system may guide the user very efficiently and reliably, but the user probably simply responds in his own way. The system cannot promise fewer dialogue turns one-sidedly without considering whether the user is cooperative

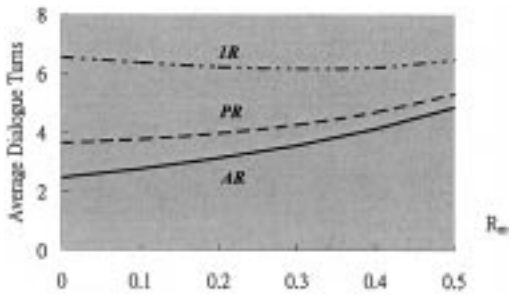

Fig. 11. Average dialogue turns for different users' response patterns (*AQ*, $R_l = 0.1$).

or not. Figs. 10(a) and 11 show that either the system or the user may become the bottleneck of the dialogue performance. While comparing Fig. 10(a) with Fig. 11, it is interesting to find that the curve for $1Q$ (given *AR*) in Fig. 10(a) approximately coincides with the curve for $1R$ (given *AQ*) in Fig. 11. In other words, the average dialogue turns for an inefficient system strategy $(1Q)$ with obedient users *(AR)* is effectively equivalent to those for efficient system's strategy *(AQ)* on inactive users $(1R)$ statistically. This also indicates that it is not necessarily very reliable for corpus-based analysis approaches to compare different system strategies using different corpora collected under different system strategies, because the user's response is critical, but usually not precisely controlled in those approaches. On the other hand, given the results in Fig. 11, it may make better sense to obtain some dialogue system behavior for users with a given distribution, say certain percentage responding as *AR*, certain *1R*, and so on.

C. System's Update Strategy

The system's update strategy certainly also influences the dialogue performance. In general, those slots in the unknown state will transit into the known state if received (*KR*). If such transitions are constrained to those slots queried by the system, the update strategy is *SI*, otherwise user-initiative *(UI)*. Apparently, if the system queries all slots *(AQ)*, it makes no difference for the system to adopt system-initiative strategy or user-initiative strategy. However, if the system does not query all unknown slots ($1Q$ or *PQ*, as previously mentioned) and the user is so experienced that he might reply some slots not queried by the system, the user-initiative strategy may become more efficient. Fig. 12 shows the average dialogue turns for system-initiative strategy and user-initiative strategy, assuming one slot is queried by the system at a time ($1Q$) and some other slots randomly chosen in addition to this slot may be replied by the user. We can see from Fig. 12 that, for inefficient system's prompt strategy such as $1Q$, smaller average dialogue turns can be achieved if the user is allowed to reply the slots out of the scope. Conventionally, the design of dialogue flows based on different system strategies is an art. The analysis approach proposed here is able to provide numerical solutions to such design, including selection among various system strategies. Although very simple examples are used here, much more complicated situations can in fact be considered in the same way.

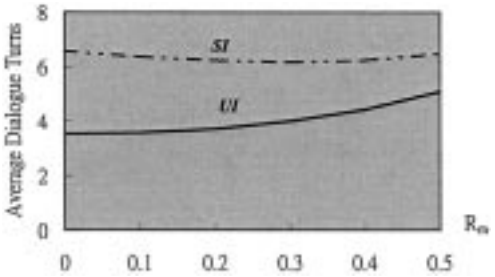

Fig. 12. Average dialogue turns for different system's update strategies $(1Q, R_l = 0.1)$.

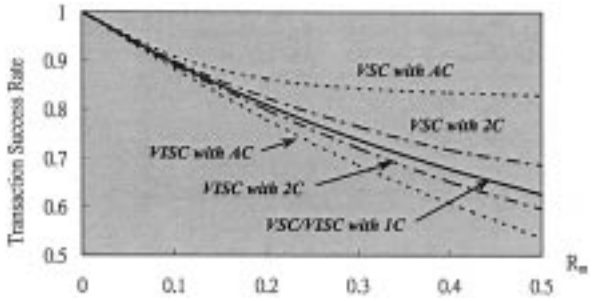

Fig. 13. Transaction success rates for *VSC* and *VISC* $(R_l = 0.2)$.

V. Analysis Example 3: Considerations for Confirmation Strategies

In the previous section, each query factor was tuned and analyzed individually for the given example. In this section, different system's prompt strategies and update strategies for confirmation will be considered and analyzed. In the simplified model mentioned in Section II-D, the prompt strategy used is prompting all slots in the known state simultaneously for confirmation (*AC*, all confirmed) as in (7), while the update strategy used is checking the slot consistency for those slots to be confirmed all together (*VSC*) as in (9). For the prompt strategy it is possible not to prompt all the slots in the known state. For example, it is possible to constrain the maximum number of simultaneously confirmed slots to m (mC, m confirmed, $m = 1, 2\ldots$). For the update strategy, there are also various ways of controlling the state transition of the known slots. Because the purpose of confirmation is to provide more reliable transactions in dialogue, here in this section more attention will be paid to improving the transaction success rate by way of selecting appropriate strategies.

A. Verified by Slot Consistency

In the strategy *(VSC)* as in the simplified model in Section II-D, all the slots to be confirmed are regarded as a whole and enter the verified state simultaneously if no inconsistency is found. So any inconsistency in the slots to be confirmed will reject the whole confirmation completely even if some slots are consistent, thus the probability for any slot to enter the verified state will be lower compared to the case that each slot is checked individually. It is interesting to see what we can gain with this strategy. The transaction success rate for this strategy when only one slot is confirmed at a time ($1C$), two slots are confirmed at most at a time ($2C$), or all slots in the known state are confirmed together (*AC*), are compared in Fig. 13 as the upper three curves. It can be found in this figure that, with this *VSC* strategy plus the *AC* strategy the transaction success rate is not only the highest but much less sensitive to the increase of R_m. To illustrate why this is the case, the state transition probabilities for *VSC* with *AC* and $1C$, respectively, at $R_l = 0.2$ and $R_m = 0.3$ are further shown in Fig. 14(a) and (b), respectively.As can be seen in this figure, for *VSC* with *AC* the transition probability from (k, e) state to (v, e) state, or wrong slots being incorrectly verified, can be significantly lower (0.0651) than that for *VSC* with $1C(0.1863)$. This makes the parameter s' in (12) much smaller (0.0847) in Fig. 14(a) than that in Fig. 14(b) (0.2004) and is the key for better transaction success rate. This obviously arises from the extra rejections due to the strict confirmation condition in *VSC* previously mentioned. Of course, the transition probability from (k, c) state to (v, c) state is also significantly lower, 0.4819 as compared to 0.9924, also due to such rejections, which is apparently not desired. But this is not too bad if only the accuracy is concerned, because the majority of those correct slots being rejected keep at (k, c) state (with probability 0.4621 as in Fig. 14(a)). All these give a relatively larger value for the parameter p' in (11) (0.1041 in Fig. 14(a) compared to 0.0076 in Fig. 14(b)). Because in *VSC* with *AC* in Fig. 14(a) the positive effect due to s' is higher than the negative effect due to p' (the effect due to c is comparable), all these effects finally give a smaller value of $P[(v, e)]$ (0.0342), and contribute to the transaction success rates in Fig. 13.

It is also of interest to see how the strategy *VSC* with *AC* behaves for other performance metrics. Fig. 15(a) shows that the strategy *VSC* with *AC* achieves significantly lower average dialogue turns than *VSC* with $1C$, which is intuitively reasonable. On the other hand, Fig. 15(b) shows that the better average dialogue turns and transaction success rates for *VSC* with *AC* are in fact obtained at the price of lower slot transmission efficiency, because some slots will be transmitted repeatedly due to rejections. Such trade-off among the performance metrics is similar to that in Section IV-A. Fig. 16 shows the tradeoff between transaction success rates and average slot efficiencies including other three prompt strategies, $2C$, $3C$ and $4C$, where mC is for confirming at most m known slots at a time, for $R_l = 0.2$ and $R_m = 0.3$. It can be seen in Fig. 16 that, when R_l and R_m are fixed, alternative prompt strategies can be chosen for different design goals considering the tradeoff.

B. Verified by Individual Slot Consistency

Of course, there is another widely used confirmation strategy, referred to as "verified by individual slot consistency" *(VISC)* here. Instead of considering the consistency of a few slots all together as in *VSC*, in this strategy the slot consistency is checked for each slot to be confirmed individually and the state transitions of these slots are separately decided. The transaction success rate for this strategy *VISC* with *AC*, $2C$, and $1C$ are also plotted in Fig. 13 as the lower three curves as compared with *VSC* discussed previously. It should be noted that when $1C$ is used *VSC* and *VISC* are the same, so they share the same curve in Fig. 13. Different from *VSC*, for *VISC* the strategy *AC* provides lower transaction success rates than $1C$ and $2C$ as in Fig. 13.

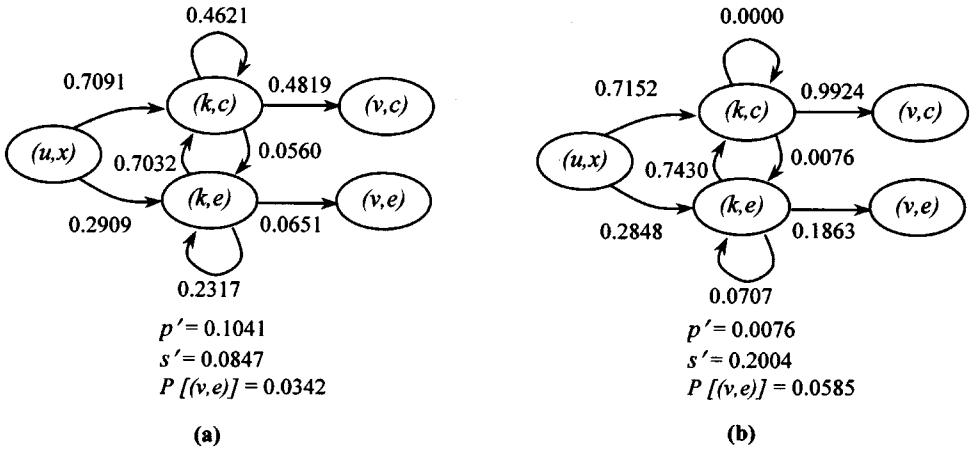

Fig. 14. Transition probabilities for (a) *VSC* with *AC* and (b) *VSC* with $1C$ at $R_t = 0.2$ and $R_m = 0.3$.

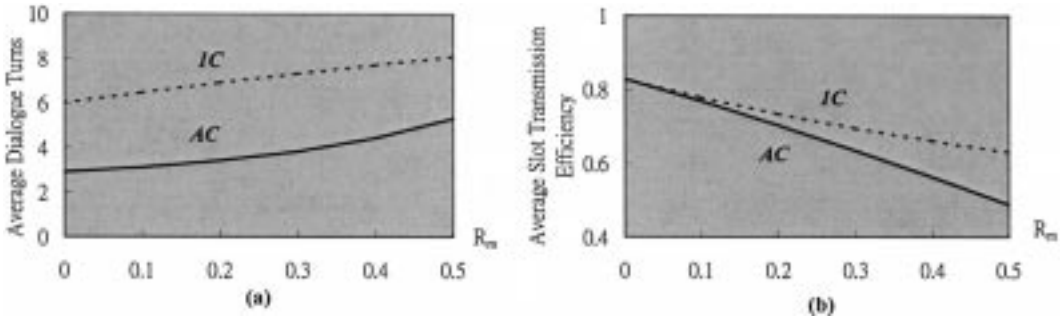

Fig. 15. (a) Average dialogue turns and (b) average slot transmission efficiency for different prompt strategies for confirmation $(R_t = 0.2)$.

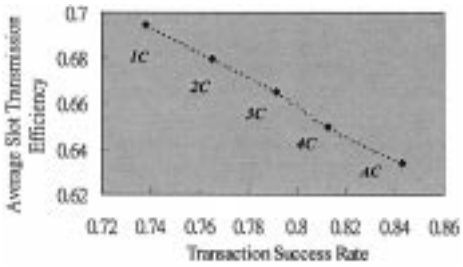

Fig. 16. Tradeoff between transaction success rate and average slot transmission efficiency ($R_t = 0.2$ and $R_m = 0.3$).

Intuitively, for *VISC* the maximum number of slots to be confirmed at a time seems not relevant to the transaction success rate, since each slot is confirmed individually. However, this is not exactly the case. In *VISC*, any slots prompted for confirmation are opened for updating and thus could be corrupted by possible incorrectly received slots. If more slots are confirmed at a time, the probability that these opened slots are corrupted by incorrectly received slots will be higher, which makes the transition probability from (k, c) to (k, e) higher. To look into such effects, the transition probabilities obtained from the simulated dialogues for *VISC* with *AC* and $1C$ are further shown in Fig. 17(a) and (b), respectively. Note that Figs. 17(b) and 14(b) are the same because with $1C$ strategy, *VSC* and *VISC* are the same. It can be found that, p' for *VISC* with *AC* (0.0705) in Fig. 17(a) is much higher than that for *VISC* with $1C(0.0076)$ in Fig. 17(b) because the transition probability from (k, c) to (k, e) for *VISC* with *AC* is much higher, while s' for *VISC* with *AC* and $1C$ are very close. This makes $P[(v, e)]$ for *VISC* with *AC* relatively higher (0.0721) than *VISC* with *1C* (0.0585), which leads to lower transaction success rate for *VISC* with *AC* as shown in Fig. 13. In fact, Fig. 13 also shows that the transaction success rate for the common sense strategy *VISC* is not necessarily satisfactory not only because it is lower, but because it is more sensitive to the understanding performance. That is, the transaction success rate is less stable if R_m varies with respect to the environment or the user. The situation is much better in the *VSC* strategy as discussed in the previous subsection.

C. Verified by Yes Detection

Another possible confirmation strategy may be "verified by yes detection" *(VYD)*, i.e., the verification is completed as long as a "yes" is detected, otherwise confirmation is rejected. This strategy relies highly on the correct detection of the word "yes." In Mandarin Chinese, the word meaning "no" is easily confused with that meaning "yes," which may lead to incorrect verification in the *VYD* strategy, though such problem may not exist at all in other languages. In this *VYD* strategy, the probability for the error event of recognizing "no" as "yes" (denoted as R_{ny} here) thus seriously affects the dialogue performance, which is shown in Fig. 18. It can be found in Fig. 18 that, if R_{ny} is high, the detection of "yes" is not trustable at all, and it is thus better to choose *VSC* instead of *VYD*. If R_{ny} is moderate, e.g., 0.1, the curves for *VSC* and *VYD* intercept with each other, and the strategy with higher transaction success rate needs to be determined by R_{ny} and R_m. If R_{ny} is small, e.g., 0.05, *VYD* should be chosen because it is more reliable.

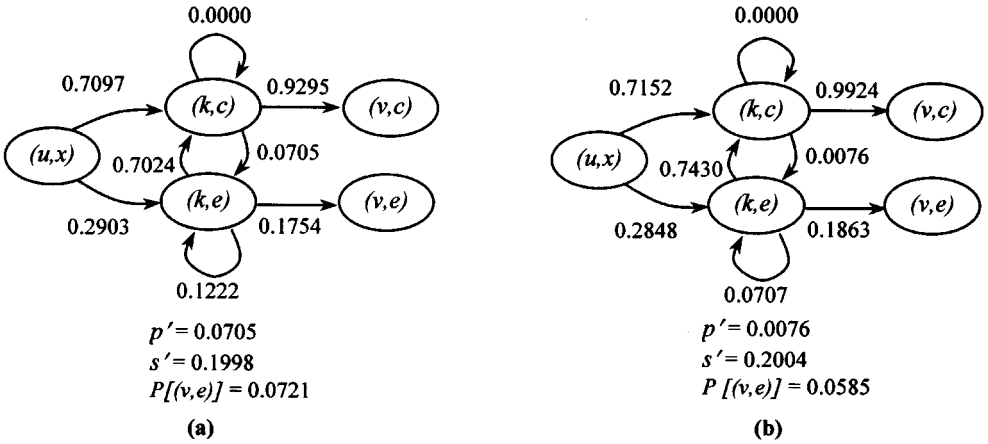

Fig. 17. Transition probabilities for (a) *VISC* with *AC* and (b) *VISC* with $1C$ at $R_l = 0.2$ and $R_m = 0.3$.

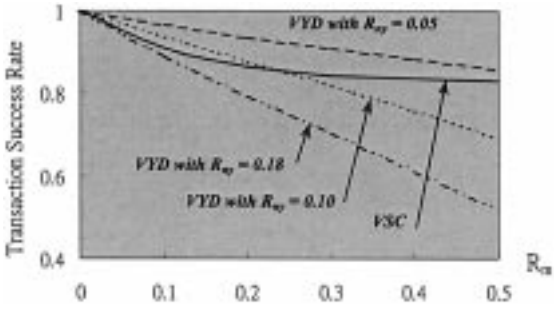

Fig. 18. Transaction success rates for different system's update strategies *VSC* and *VYD* with different values of $R_{ny}(R_{ny} = 0.05, 0.10, 0.18, R_l = 0.2)$.

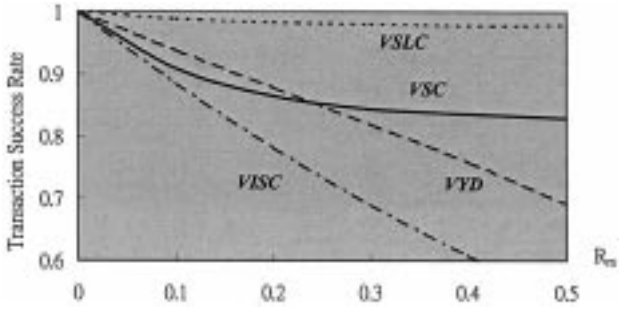

Fig. 19. Transaction success rate for *VISC, VSC, VYD,* and *VSLC* $(R_{ny} = 0.1, R_l = 0.2)$.

D. Verified by Slot and Logic Consistency

Because in the previous subsections both strategies *VSC* and *VYD* make sense, it is very natural to try to integrate the concepts of the two. Assume the current system prompt for confirmation is, "Would you like to go to Taipei tomorrow?" and the user's reply is recognized as, "No, I would like to go to Taipei tomorrow." For *VSC*, because no slot inconsistency occurs, the slots, "Taipei" and "tomorrow" are therefore both verified. But such verifications are not reasonable because there is logical contradiction due to the existence of "no." On the other hand, the user's reply may be recognized as "Yes, I would like to go to Hsinchu tomorrow." For *VYD*, the detection of "Yes" makes those slots verified, but again there is logic contradiction due to the slot inconsistency of "Hsinchu" with "Taipei." These logic contradictions in either *VSC* or *VYD* imply possible incorrect verifications. A better strategy may be *VSLC*, in which both the detection of "yes" or "no" and the slot consistency are simultaneously considered, and those utterances with logic contradictions are rejected. Fig. 19 shows the transaction success rates for *VSC*, *VISC*, *VYD*, and *VSLC*, respectively, with the prompt strategy *AC*. It can be observed in this figure that *VSLC* can achieve apparently the highest and the most stable transaction success rates among all strategies. Of course, the increased rejections created by logic contradictions for *VSLC* is the price paid for the higher transaction success rates, which inevitably leads to higher average dialogue turns or lower slot transmission efficiency, as shown in Fig. 20(a) and (b), respectively. Again, this is the tradeoff among performance metrics and should be up to the designer's choice. Also, it is interesting to see that in order to achieve higher transaction success rates, the range for tuning in Fig. 19 by selecting among update strategies is apparently larger than that in Fig. 16 by selecting among the prompt strategies. Also, Fig. 19 indicates that that even if the speech understanding front end cannot be improved or tuned, it is still possible to achieve significantly more reliable transactions through the selection of dialogue strategies.

To see how the improvement of transaction success rate for *VSLC* is achieved, the state transition diagram for *VSLC* with *AC* is further shown in Fig. 21. As can be found by comparing this figure with Fig. 14(a) for *VSC*, due to the rejections caused by logic contradictions in *VSLC*, the transition probability from (k, e) to (v, e) is now reduced from 0.0651 in Fig. 14(a) to 0.0080 here, which reduces s' from 0.0847 to 0.0105, while p' for *VSC* and *VSLC* are very close. The probability $P[(v, e)]$ is thus significantly lower here (0.0042 as compared to 0.0342).

VI. Analysis Example 4–Objective Estimates for User's Degree of Satisfaction

It is highly desired for dialogue system designers to have some kind of "user's degree of satisfaction" about the system, which is usually obtained by subjective evaluations obtained from questionnaires filled up by the users. In this section, we will show how some objective metrics very similar to user's degree of satisfaction can be obtained with the simulation approach proposed here, and how these metrics can be used in

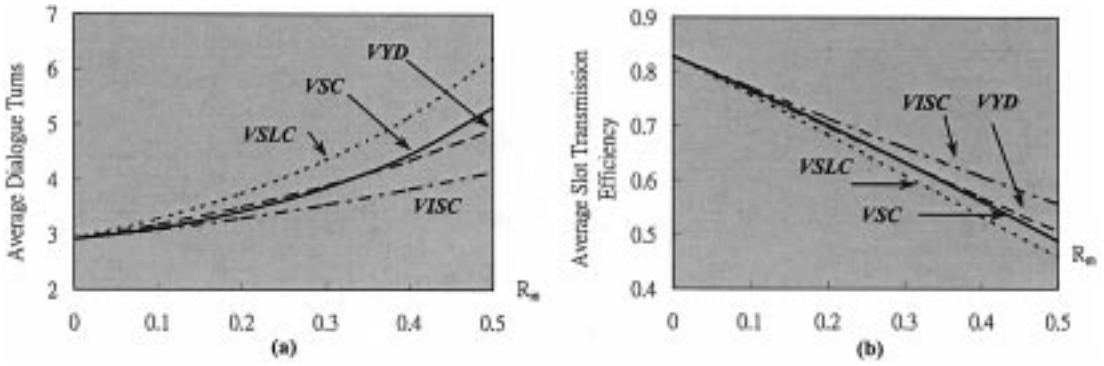

Fig. 20. (a) Average dialogue turns and (b) average slot transmission efficiency for *VISC*, *VSC*, *VYD*, and *VSLC* ($R_{ny} = 0.1, R_l = 0.2$).

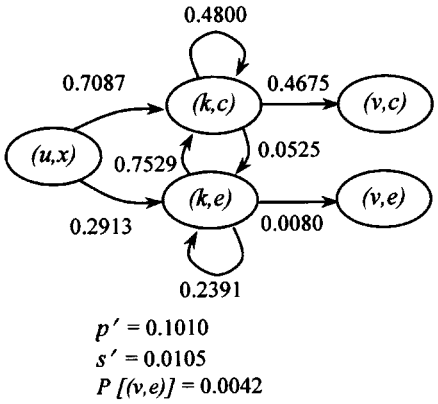

Fig. 21. Transition probabilities for *VSLC* with *AC* at $R_1 = 0.2$, $R_m = 0.3$, and $R_{ny} = 0.1$.

improving the user's degree of satisfaction in dialogue system design.

A. Percentage of Confused Users

As discussed in Section V-A, for *VSC* prompting more known slots for confirmation (*AC*) achieves higher transaction success rates. However, the more the known slots are prompted for confirmation, the higher the probability that some of them are incorrect will be. If more than one incorrect slots are prompted for confirmation, the user may feel confused, and probably give up the dialogue or complain about the system. A possible metric for measuring a user's degree of satisfaction along this direction can therefore be defined as the percentage of the dialogues in which there is at least one system's prompt utterance for confirmation including more than one incorrect slots. This metric is referred to as "percentage of confused users" here, which can be easily observed directly from the simulated dialogues. This percentage of confused users for the "baseline system" as in Section II-E with confirmation strategy *AC* at different understanding performance (R_m, R_l) is shown in Fig. 22(a). As can be found in this figure, the percentage of confused users increases significantly with R_m. Therefore, reducing R_m can be a good approach to reduce the percentage of confused users.

On the other hand, the above percentage of confused users can also be reduced by controlling the number of slots prompted for confirmation. For example, if at most two known slots can be prompted for confirmation simultaneously ($2C$), the probability of more than one slots being incorrect will be lower when compared with *AC*. But the smaller number of slots prompted for confirmation will lead to the degradation of transaction success rates, as discussed in Section V-A. Therefore, there exists a tradeoff between the transaction success rate and the percentage of confused users, which is shown in Fig. 22(b). As can be observed in Fig. 22(b), by reducing the number of slots to be confirmed, the percentage of confused users can be significantly reduced at the price of a slight degradation in transaction success rate. For example, the percentage of confused users can be reduced from 23.79% for *AC* to only 6.61% for $2C$, with the transaction success rate degraded from 93.81% for *AC* to 91.44% for $2C$. This is another example of achieving some desired goal by choosing system strategies.

It should be pointed out that, the percentage of confused users here is only one example of many possible metrics which can be flexibly defined to reflect the user's degree of satisfaction and objectively estimated using the proposed approach. The occurrence of more than one incorrect slots prompted for confirmation is only one example "undesired event" for the users out of many other possible "undesired events." Such "undesired events" may be identified from questionnaires filled up by the users or from the observation of the dialogue corpus. More similar metrics can be defined with such events.

B. Percentage of Satisfied Users

It is a common experience in dialogue design that high accuracy and high efficiency very often cannot be obtained simultaneously. One example is shown in Fig. 16, which shows higher transaction success rate and higher average slot transmission efficiency are sometimes conflicting design goals. Considering the user's degree of satisfaction, on the other hand, accuracy and efficiency are both highly desired. Along this direction, a possible metric to be objectively estimated in simulations can be defined as the percentage of dialogues in which not only correct transaction is achieved, but the slot transmission efficiency in (10) is higher than some desired value, say 70%. This metric is referred to as "percentage of satisfied users" in this paper. In such a definition, both accuracy and efficiency are considered simultaneously. Of course, this metric is only a simple example. More complicated definitions are certainly possible. For example, an extra condition may be 80% or more of the slots are transmitted only once in a dialogue, or something similar.

Fig. 23(a) shows the percentage of satisfied users for the "baseline system" in Section II-E but with different system's

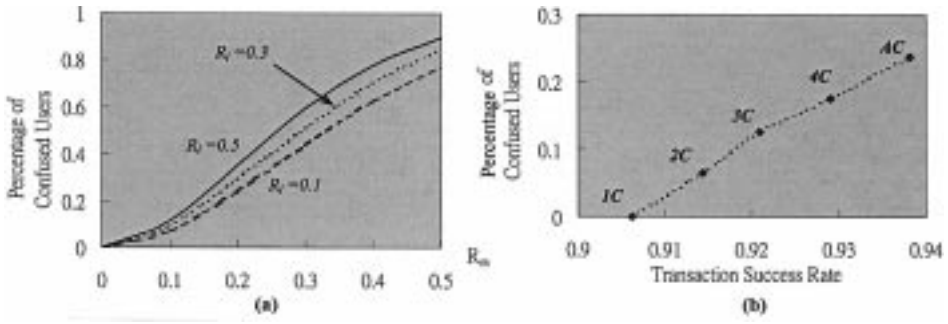

Fig. 22. (a) Percentage of confused users with system's prompt strategy *AC* for different understanding performance $(R_l 0.1, 0.3, 0.5)$ and (b) tradeoff between transaction success rate and the percentage of confused users for different confirmation strategies $(R_l = 0.1, R_m = 0.2)$.

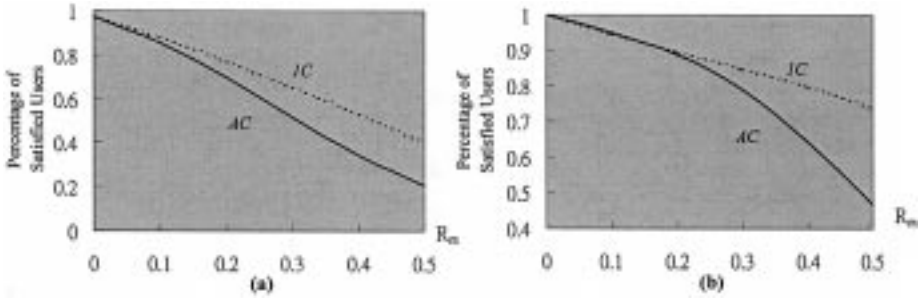

Fig. 23. Percentage of satisfied users for different system's prompt strategies *AC* and $1C$ $(R_l = 0.1)$ when the slot transmission efficiency is required to exceed (a) 70% (b) 50% in a dialogue.

prompt strategies for confirmation, $1C$ and *AC*, for different R_m given $R_l = 0.1$. It can be seen in Fig. 23(a) that, for the percentage of satisfied users defined here, $1C$ performs better than *AC*. The trend of the curves in Fig. 23(a) is similar to that of the curves in Fig. 15(b) where the average slot transmission efficiency (or the goal of efficiency) is the performance metric, but opposite to that of the two corresponding curves for *VSC* with *AC* and $1C$ in Fig. 13 where the transaction success rate (or the goal of accuracy) is the performance metric. Because in the new metric here both goals of accuracy and efficiency are somehow included, the results here indicate that the goal for efficiency actually dominates, or the condition of slot transmission efficiency being higher than 70% for each satisfied user is more stringent. It may be interesting to see what happens if in this metric the condition of slot transmission efficiency for each satisfactory dialogue is relaxed from 70% to 50%. Fig. 23(b) shows the percentages of satisfied users for *AC* and $1C$, respectively, for such case. We can find in this figure that when the condition for efficiency is relaxed, *AC* becomes slightly better when R_m is small, and the two curves actually intercept with each other at somewhere around $R_m = 0.15$. Because this percentage of satisfied users is a function defined on the two-dimensional (2-D) plane (R_m, R_l), the situation in Fig. 23(b) implies that the 2-D functions for the two strategies $1C$ and *AC* in fact intercept with each other. In other words, in such cases, neither of the two competing goals can dominate the performance trend, and the better system's prompt strategy needs to be quantitatively determined by simulations for different operating points on the (R_m, R_l) plane. Similar phenomena may be observed in more sophisticated situations where some metrics are defined for multiple conditions. If none of the conditions actually dominates the performance trend, the functions for these metrics for different strategies may intercept sophisticatedly, but the selection of strategy can always be determined numerically as in Fig. 23(b).

VII. Possible Extensions from the Simplified Model

All the previous discussions and analyses have been based on over-simplified models which may seem far from reality for practical dialogue system designers. In fact, all that can be done in this paper is to use over-simplified models to show the basic methodologies and general principles, and it will not be too difficult for practical dialogue system designers to extend those methodologies and principles to many different realistic situations. In this section, we will use a few examples to illustrate such extensions and show the flexibility of the proposed approach.

A. State Transition Models

There can be many state transition models different from that in Fig. 1. First, in practical dialogue systems, not all slots should necessarily start with the unknown state. In some cases, assigning some slots with initial default values may make the dialogue more natural. For example, the date, the departure and destination station for a flight ticket reservation dialogue system may have initial values if the user profile is given. In such conditions, all one needs to do is to set the initial states of these slots as (k, c) or (k, e) instead of (u, x) in the simulation, and the variations in dialogue performance due to such changes in dialogue design can be easily observed objectively in the simulation. Second, in practical dialogue systems, not all the slots necessarily always need to be confirmed. For example, confirmation may be saved for those slots with confidence

measures higher than a threshold. Fig. 24 shows an alternative topology for such case. Furthermore, in the over-simplified examples given above in this paper, the state transition rules have been assumed to be identical for all different slots. Of course, it may not be the case for many practical dialogue systems, and one does not have to do the simulation this way. In a train ticket reservation dialogue system, for example, the receipt of "3 o'clock" for the slot "time" may automatically imply "p.m." for the slot "time-of-day" (assuming "time" and "time-of-day" are different slots) considering the knowledge in the train schedule database. Apparently the state transition models and rules of these two slots "time" and "time-of-day" should be different and dependent in this case. Such condition can certainly be simulated, except that the state transition models and rules should be precisely written according to the specific requirements or designs as in this example.

B. System's Prompt Strategies

In these discussions, the system's prompts are randomly generated with all slots equally handled, by such rules as "confirming at most two slots at a time." In many practical situations, however, very often deterministic rules are used in generating the prompts with different slots treated differently. For example, for ticket reservation the system may ask and confirm the destination slot before the time slot. Such deterministic rules are usually implemented using some sort of script language, as shown in the example in Fig. 25(a), which is quite different from the simulation scenario described above. In fact, this is because in the implementation in Fig. 25(a) the dialogue management is tangled with the sentence generation. A minor modification as shown in Fig. 25(b) may make the dialogue management handle only the slot level interaction (e.g., which slot to confirm and which slot to query). In this way, we can separate clearly the sentence generation from dialogue management, and the prompt strategies can be easily simulated with the scenario described above, which can also be in good parallel with the operation of the real system.

C. Channel Effect

In Section II-C, the speech recognition and understanding process is modeled as a slot transmission channel with two parameters R_m and R_l. In the simulation given in the previous sections, the event for incorrectly arrived slots (R_m) and that for lost slots (R_l) are simulated using two independent random tests. Of course the correlation between these two parameters may need to be considered, which can be simulated in the way given as follows.

Define X and Y as two random variables, both of them can have only two values, 0 and 1. $X = 1$ means receiving an incorrect slot, and $Y = 1$ means losing a slot. Let $P(x, y)$ be the probability density function of the two random variables. $P(x, y)$ can have only four discrete values as follows:

$$\begin{aligned} P(x,y) &= k \quad x=0, \quad y=0 \\ P(x,y) &= l \quad x=1, \quad y=0 \\ P(x,y) &= m \quad x=0, \quad y=1 \\ P(x,y) &= n \quad x=1, \quad y=1 \end{aligned} \tag{24}$$

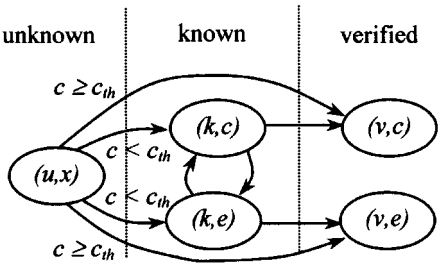

Fig. 24. State transition considering confidence measures.

```
if destination.isempty
        prompt "Where would you like to go"
else if NOT destination.isempty AND NOT destination.verified
        prompt "Would you like to go to" & destination.value
else if date.isempty
        prompt "What date would you like to go"
....
```

(a)

```
if destination.isempty
        destination.prompt_query = true
else if NOT destination.isempty AND NOT destination.verified
        destnation.prompt_confirm = true
else if date.isempty
        date.prompt_query = true
....
```

(b)

Fig. 25. (a) Dialogue manager tangled with sentence generation and (b) dialogue manager separated from sentence generation.

where $1 \geq k, l, m, n \geq 0$ and $k + l + m + n = 1$. It is then easy to see that

$$E(X) = l + n = R_m \tag{25}$$

$$E(Y) = m + n = R_l \text{ and} \tag{26}$$

$$C_{xy} = E(XY) - E(X)E(Y) = n - R_m \cdot R_l \tag{27}$$

where C_{xy} is the covariance for the two random variables X and Y. These above probability density function $P(x, y)$ can be used to simulate the condition when X and Y are correlated. X and Y are uncorrelated when $C_{xy} = 0$. In the tests of a real speech understanding front end, we obtained the parameters $R_m = 0.096\,59$, $R_l = 0.2014$, and $C_{xy} = 0.034\,15$ from the statistics of 2117 utterances. Table I shows the simulation results for the baseline schemes described previously at $R_m = 0.096\,59$, $R_l = 0.2014$, with $C_{xy} = 0$ and $0.034\,15$, respectively. As can be seen in this table, the simulation results for the two cases are quite close. Note the value of C_{xy} here may depend on the recognizer, the speakers, and many other factors. On the other hand, one can of course build other finer models for the channel effect. For example, the parameters may be different for different slots, or depend on the number of slots transmitted, with higher complexity in modeling and simulation.

D. Users' Patterns

In the analysis examples in the above sections, fixed patterns of user's response are assumed in the over-simplified model. Of course the real user behavior is much more complicated.

TABLE I
SIMULATION RESULTS FOR THE BASELINE SCHEMES AT $R_m = 0.09659$, $R_l = 0.2014$

C_{xy}	N_t	T_s	E_s
0 (uncorrelated)	3.11	0.8633	0.7687
0.03415 (correlated)	3.04	0.8700	0.7774

More complicated models can always be assumed and simulated though, as were mentioned at the end of Section IV-B

VIII. CONCLUSION

In this paper, we have presented a complete development of computer-aided analysis and design approaches for spoken dialogue systems based on quantitative simulations. Such an approach is very useful before the system implementation is completed. With this approach, how the different dialogue system performance measures vary with respect to different system factors and design parameters can be analyzed individually because all these factors and parameters can be precisely controlled in the simulation. Examples indicate that selection and tuning of speech understanding front end and design of complicated dialogue strategies for given performance goals, as well as objective estimation of user's degree of satisfaction, which are always difficult problems in conventional dialogue system design, can now be performed numerically with the proposed approach. The online test, corpus-based analysis and user survey can always follow after the system is online. It should be noted that although all discussions here are based on simplified models and very general scenarios, more specific analysis and design can definitely be performed for a specific dialogue task or system, as long as the specific conditions are given. Therefore the proposed approach not only is a powerful tool for developing spoken dialogue systems, but can provide an important basis for further research on spoken dialogue systems.

REFERENCES

[1] M. A. Walker *et al.*, "Evaluating spoken dialogue agents with PARADISE: Two case studies," *Comput. Speech Lang.*, vol. 12, pp. 317–347, 1998.

[2] H. Aust and H. Ney, "Evaluating dialog systems used in the real world," in *Proc. Int. Conf. Acoustics, Speech, Signal Processing*, 1998.

[3] G. Hanrieder, P. Heisterkamp, and T. Brey, "Fly with the EAGLES: Evaluation of the "ACCeSS" spoken language dialogue system," in *Proc. Int. Conf. Speech Language Processing*, 1998.

[4] C. Kamm *et al.*, "Evaluating spoken dialog systems for telecommunication services," in *Proc. EUROSPEECH*, 1997, pp. 2203–2206.

[5] C. A. Lavell, M. de Calmes, and G. Perennou, "Confirmation strategies to improve correction rates in a telephone inquiry dialog system," in *Proc. EUROSPEECH*, 1999, pp. 1399–1402.

[6] J. Sturm, E. den Os, and L. Boves, "Dialogue management in the Dutch ARISE train timetable information system," in *Proc. EUROSPEECH*, 1999, pp. 1419–1422.

[7] L. Devillers and H. Bonneau-Maynard, "Evaluation of dialog strategies for a tourist information retrieval system," in *Proc. Int. Conf. Speech Language Processing*, 1998.

[8] L. Lamel and S. Bernacef *et al.*, "User evaluation of MASK kiosk," in *Proc. Int. Conf. Speech Language Processing*, 1998, pp. 2875–2878.

[9] E. Hurley, J. Polifroni, and J. Glass, "Telephone data collection using the World Wide Web," in *Proc. Int. Conf. Speech Language Processing*, 1996, pp. 1898–1901.

[10] A. Life and I. Salter *et al.*, "Data collection for the MASK kiosk, WOz vs. prototype system," in *Proc. ICSLP*, 1996, pp. 1672–1675.

[11] L. J. M. Rothkrantz *et al.*, "An appreciation study of an ASR inquiry system," in *Proc. EUROSPEECH*, 1997, pp. 1715–1718.

[12] N. O. Bernsen, H. Dybkjar, and L. Dybkjar, *Designing Interactive Speech Systems. From First Ideas to User Testing*. , Germany: Springer-Verlag, 1998.

[13] S. Rosset, S. Bennacef, and L. Lamel, "Design strategies for spoken language dialogue systems," in *Proc. EUROSPEECH*, 1999, pp. 1535–1538.

[14] Y. Niimi, T. Nishimoto, and Y. Kobayashi, "Analysis of interactive strategy to recover from misrecognition of utterances including multiple information items," in *Proc. EUROSPEECH*, 1997, pp. 2251–2254.

[15] Y. Niimi and T. Nishimoto, "Mathematical analysis of dialogue control strategies," in *Proc. EUROSPEECH*, 1999, pp. 2251–2254.

[16] W. Eckert, E. Levin, and R. Pieraccini, "User modeling for spoken dialogue system evaluation," in *Workshop Automatic Speech Recognition Understanding*, 1997.

[17] ——, "Automatic evaluation of spoken dialogue systems," AT&T Labs Res., TR98.9.1, 1998.

[18] M. Araki and S. Doshita, "Automatic Evaluation Environment for Spoken Dialogue Systems," in *Dialogue Processing in Spoken Language Systems*. Berlin, Germany: Springer-Verlag, 1998, pp. 183–194.

[19] T. Watanabe, M. Araki, and S. Doshita, "Evaluating dialogue strategies under communication errors using computer-to-computer simulation," *IEICE Trans. Inform. Syst.*, vol. E81-D, no. 9, Sept. 1998.

[20] K. Scheffler and S. Young, "Probabilistic simulation of human-machine dialogues," in *Proc. Int. Conf. Acoustics, Speech, Signal Processing*, 2000, pp. 1217–1220.

[21] M. Boros and W. Eckert *et al.*, "Toward understanding spontaneous speech: Word accuracy vs. concept accuracy," in *Proc. Int. Conf. Speech Language Processing*, vol. 2, 1996, pp. 1009–1012.

[22] B. Lin, B. Chen, H. Wang, and L. Lee, "Hierarchical tag-graph search for spontaneous speech understanding in spoken dialogue systems," in *Proc. Int. Conf. Speech Language Processing*, 1998.

Bor-shen Lin was born in Taiwan in 1968. He received the B.S. and M.S. degrees in electrical engineering from National Taiwan University (NTU), Taipei, Taiwan, R.O.C., in 1989 and 1993, respectively. He received the Ph.D. degree in electrical engineering from NTU in 2001.

He was with R&D Department, Bicom Technology Corporation, Taipei, from 1993 to 1996. His research interests include speech recognition and understanding, search, evaluation methodology, performance and error analysis, dialogue modeling, and spoken dialogue system.

Lin-shan Lee (S'76–M'77–SM'88–F'93) received the Ph.D. degree in electrical engineering from Stanford University, Stanford, CA.

He has been a Professor of electrical engineering and computer science at the National Taiwan University, Taipei, since 1982; he was a Department Head from 1982 to 1987. He holds a joint appointment as a Research Fellow of Academia Sinica, Taipei, and was an Institute Director there from 1991 to 1997. His research interests include digital communications and Chinese spoken language processing. He developed several of the earliest versions of Chinese spoken language processing systems in the world, including text-to-speech system, natural language analyzer, and dictation systems.

Dr. Lee was the Guest Editor of a special issue on intelligent signal processing in communications of the IEEE JOURNAL OF SELECTED AREAS IN COMMUNICATIONS in December 1994 and January 1995. He was the Vice President for International Affairs (1996 –1997) and the Awards Committee Chair (1998 –1999) of the IEEE Communications Society. He has been a member of Permanent Council of International Conference on Spoken Language Processing (ICSLP).