COMPUTER-AIDED DESIGN/ANALYSIS FOR CHINESE SPOKEN DIALOGUE SYSTEMS

Bor-shen LIN
Department of Electrical Engineering, National Taiwan University, Taipei
bsl@speech.ee.ntu.edu.tw

Lin-shan LEE
Department of Electrical Engineering, National Taiwan University, Taipei
lsl@iis.sinica.edu.tw

ABSTRACT
Conventionally design principles for spoken dialogue systems are drawn either from experiences or from corpus-based analysis. However, human experiences are usually not precise enough for engineering design, while for corpus-based analysis many factors such as speech recognition or understanding performance and user's behavior can never be precisely controlled. Recently, a new design/analysis approach by computer simulation was proposed. This paper presents the experiences of using this approach to design Chinese spoken dialogue systems. The simulation indicated the following observations and design principles. The transaction success rate (reliability) and slot transmission efficiency (efficiency) are usually conflicting design goals, and trade-off between them thus exists. Since reliability is more important than efficiency in general, it is desirable to achieve higher reliability at the price of reduced slot transmission efficiency when the reliability is not adequate. According to the simulation results, when the speech recognition accuracy cannot be improved, there still exists limited flexibility for tuning the dialogue performance by selecting among the strategies and considering the trade-offs. It is not only possible to select among the strategies considering the design goals, but to estimate the gain obtained and the price paid in the selection. New dialogue strategies can also be designed and numerically verified in this way.

1. INTRODUCTION
Conventionally design principles for spoken dialogue systems are drawn either from experiences or from corpus-based analysis [1-4]. However, human experiences are usually not precise enough for engineering design, while for corpus-based analysis many factors such as speech recognition or understanding performance and user's behavior can never be precisely controlled. Recently, a new design/analysis approach by computer simulation was proposed [5]. In this approach, the state space of a dialogue, $S$, is represented as a set of finite state machines, that is, $S = (s_1, s_2, \ldots, s_n)$. Each dialogue turn is simulated in four carefully modeled segments: system's prompt strategy, user's response pattern, speech recognition or understanding and system's update strategy. Different performance metrics, such as average dialogue turn or transaction success rate and so on, can thus be obtained by directly observing the simulated dialogue processes. With this approach, the effect of each design strategy or system component can be precisely identified and the system designed more efficiently. This paper presents the experiences of using this approach to design Chinese spoken dialogue systems.

In the example dialogue system described here, train schedule information was queried in Mandarin Chinese. Different system confirmation strategies were simulated and compared. The simulation indicated the following observations and design principles. First, transaction success rate (reliability) and slot transmission efficiency (efficiency) are usually conflicting design goals, and trade-off between them thus exists. In general, reliability is more important than efficiency. When the reliability is not adequate, it is desirable to achieve higher reliability at the price of reduced efficiency. Of course, the dialogue system performance can be improved by improving the speech recognition accuracy. However, according to the analysis, even if the speech recognition accuracy cannot be improved, there still exists limited flexibility for tuning the dialogue performance by selecting among the strategies and considering the trade-offs. It is not only possible to select among the strategies considering the design goals, but to estimate the gain obtained and the price paid in the selection. New dialogue strategies can also be designed and numerically verified in this way.

2. COMPUTER SIMULATION

2.1 State Representation
In the proposed approach, a dialogue is modeled as the processes that a set of semantic slots to be transmitted from the user to the system. The finite state machine for each semantic slot, $s_e$, is first represented in a two-tuple expression, as shown in Figure 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,

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

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

$S_i = (s_1=(u,x), s_2=(u,x), \ldots, s_n=(u,x))$.  \hspace{1cm} (2)
while the final state $S_f$ is

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

(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 Figure 1, such that the overall state $S$ may transit from the initial state $S_0$ to the final state $S_f$. A successful transaction then occurs when in the final state all the semantic slots are verified and correct, i.e.,

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

(4)

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

unknown    known   verified

(u,x)      (k,c)    (v,c)

(v,c)      (k,e)    (v,e)

Figure 1. Finite state machine for each semantic slot

2.2 Simulation Scheme

The simulation scheme can be represented with the pseudo codes as shown in Figure 2. The cycle of a dialogue turn can be simulated by four segments: system prompt, user response, speech understanding and system update, as shown within the for-loop in Figure 2. In the 'system prompt' segment, how the system decides which slots should be queried and which slots should be confirmed is simulated. In the 'user 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 so as to decide 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 described above, the dialogue performance is a function of 4 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 $S_U$,

$$P_D = F(S_P, U, C, S_U),$$

(5)

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

```
for(N=0;Goal&&N<++;N++)
{
    SystemPrompt(); // system’s prompt strategy
    UserResponse(); // user’s response pattern
    SpeechUnderstanding(); //understanding performance
    SystemUpdate(); // state transition
}
```

if(AllSlotCorrect()) $T_i = 1$; // successful transaction
else $T_i = 0$; // error transaction

$E_i = n/h$; // slot transmission efficiency

Figure 2. Pseudo codes for simulation of a dialogue

2.3 Speech Understanding or Channel Effect

Conventionally, the speech understanding errors are often measured by slot error rate, which includes the rates for inserted, deleted, and substituted slot errors, $R_{ins}$, $R_{del}$ 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 as mentioned above 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 metrics:

$$R_{m} = R_{ins}+R_{del}, \quad R_l = R_{del}+R_{sub}$$

(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 Figure 3.

```
```

```
```

Figure 3. Channel effect for speech understanding

2.4 System’s Strategies and User’s 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 here the goal of dialogue 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 effects on the state transition. Those factors mainly determining the state transition from the unknown state to the known state are referred to as ‘query factors’ here, while those mainly determining 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, 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 such factors are difficult to parameterize, it is possible to use simplified models to specify these factors. Some simple examples for such models are presented below for illustration purposes. A simple model for the system’s prompt strategy $S_P$ is

$$S_P : (AQ, AC),$$

(7)

which means all slots in the unknown state are queried (All Queried, AQ), while all slots in the known state are confirmed (All confirmed, AC). The former specifies the query factors, while the latter the confirmation factors. Similarly for the user’s response pattern $U$.

$$U : (AR/NQNT, YC),$$

(8)
the first part means all queried slots are replied (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, 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 is


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 all are received consistently (Verified by Slot Consistency, VSC). All other slots are not updated, or system-initiative (SI).

### 2.5 Fundamental Statistical Analysis

For the simulation of each dialogue, after the four segments are iterated for enough number of times, the final state is achieved and the dialogue terminated, as shown in Figure 2. Various characteristic parameters of the dialogue can then be extracted. Below are some examples. First, the number of dialogue turns, $n$, can be obtained in the for-loop in Figure 2. The transaction success flag, $T_s$, which equals to 1 if a successful transaction is achieved and 0 if not, can then be determined by checking the second argument of the finite state machines, i.e., to see if $S_f = (s_1=(v,c), s_2=(v,c), \ldots, s_n=(v,c))$. The slot transmission efficiency, $E_s$, can be defined as

$$E_s = n/n',$$

where $n'$ is the total number of transmitted slots, which can be observed in the ‘user response’ segment. This slot transmission efficiency $E_s$ indicates whether the user can transmit the slots efficiently, whose value ranges from 0 to 1. 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.

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, which are useful parameters to measure 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.

### 3. A DESIGN/ANALYSIS EXAMPLE FOR CONFIRMATION STRATEGIES

#### 3.1 System’s Prompt Strategy

First, consider the system’s prompt strategies for confirmation. In the simplified model mentioned in Section 2.4, all slots in the known state are simultaneously prompted for confirmation (AC, All Confirmed) as in equation (7). Another possible strategy for comparison may be prompting the known slots for confirmation one by one (1C, 1 Confirmed). Figure 4(1) shows the strategy AC achieves lower average dialogue turns than the strategy 1C, which is intuitively reasonable. Figure 4(2) further shows that the strategy AC achieves the transaction success rates not only better but also more robustly, i.e., less sensitive to the misunderstanding rate $R_m$, than the strategy 1C. Of course, there is no free lunch. Figure 4(3) shows that, on the other hand, the better average dialogue turns and transaction success rates for the strategy AC is in fact obtained at the price of worse slot transmission efficiency. Figure 4(4) shows the trade-off between transaction success rates and average slot efficiencies including other three prompt strategies: nC for confirming at most $n$ known slots ($n=2-4$), as compared to 1C and AC for $R_m = 0.3$. Figure 4(4) shows that different alternatives for confirmation strategies can be chosen for different design goal. It is not only possible to select among the strategies considering the design goals, but to estimate the gain obtained and the price paid in the selection.
3.2 System’s Update Strategy

For the system’s update strategies for confirmation, the simplified models in Section 2.4 used the strategy of ‘Verified by Slot Consistency’ (VSC) as in equation (9). Another possible strategy can be ‘Verified by Yes Detection’ (VYD), i.e., the verification is completed as long as a ‘Yes’ is detected. This strategy relies on the detection of the word ‘Yes’. In Mandarin Chinese, the word meaning ‘No’ (‘不’) is easily confused with that meaning ‘Yes’ (‘是’), which may lead to incorrect verification in the VYD strategy, though such problem may not exist at all in other languages. For Mandarin Chinese the probability for such error event of recognizing ‘No’ as ‘Yes’ (R_m) affects the dialogue performance if VYD is used, which is shown in Figure 5. It can be found in Figure 5, VYD can be better than VSC in transaction success rate only when both R_m and R_l are small. This shows that better confirmation strategy can in fact be decided quantitatively.

Another two strategies are further designed and compared. One strategy, similar to VSC in concept, is ‘Verified by Individual Slot Consistency’ (VISC), i.e., the slot consistency is checked for each slot individually. For VSC the slot consistency is checked for the confirmed slots all together and these confirmed slots therefore either enter the verified state or stay in the known state simultaneously, while for VISC the slot consistency is checked for the confirmed slots individually and therefore the state transitions of these slots are probably different. The other strategy is ‘Verified by Slot Consistency plus Logic consistency’ (VSCL), which tries to integrate both the slot consistency and the detection of ‘Yes’ or ‘No’. Assume the current system prompt for confirmation is, ‘Would you like to go to Taipei tomorrow morning?’, and the user’s reply is recognized as, ‘No, I would like to go tomorrow.’. For VISC the slot values ‘Taipei’ and ‘tomorrow’ are verified because there is no slot inconsistency, but such verification is not reasonable due to the detection of ‘No’ here. Similarly, if the user’s reply is recognized as, ‘Yes, I would like to go to Hsinchu tomorrow.’, for VYD the confirmation will pass due to the detection of ‘Yes’, but again this is unreasonable because the slot value ‘Hsinchu’ is inconsistent with ‘Taipei’. If ‘logic inconsistency’ occurs as in the above two cases, there must be recognition errors in either the detection of ‘Yes’ or ‘No’, or the slots to be confirmed. Since there is no way to find out which error actually occurs, the recognition result is not reliable and had better be discarded. Figure 6 shows the transaction success rates for VISC, VSC, VYD, and VSCL. It can be observed in this figure that, VSC has higher transaction success rate than VISC, while VSCL achieves even higher rate. Of course, the increased rejections created by ‘simultaneous confirmation’ for VSC and by ‘logic inconsistency’ for VSCL respectively is the price paid for the higher transaction success rate (or higher reliability), which inevitably leads to higher average dialogue turns or lower slot transmission efficiency, as shown in Table 1 and 2 respectively. Again, this is the trade-off among performance metrics and should be up to the designer’s choice. Of course, according to Figure 6, an efficient way to improve reliability is to improve the speech understanding performance (e.g. reducing R_m). However, even if the understanding performance can’t be further improved, it is still possible to achieve more reliable communication by choosing appropriate strategy.

![Figure 5](image1.png)

**Figure 5.** Transaction success rates for VYD and VSC with different values of \( R_m \) (\( R_m = 0.1, 0.14, 0.18 \), \( R_l = 0.2 \))

![Figure 6](image2.png)

**Figure 6.** Transaction success rate for VISC, VSC, VYD and VSCL (\( R_m = 0.1, R_l = 0.2 \))

### Table 1. Average dialogue turns for VISC, VSC, VYD, and VSCL (\( R_m = 0.1, R_l = 0.2 \))

<table>
<thead>
<tr>
<th>( R_m )</th>
<th>VISC</th>
<th>VSC</th>
<th>VYD</th>
<th>VSCL</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>3.08</td>
<td>3.11</td>
<td>3.17</td>
<td>3.28</td>
</tr>
<tr>
<td>0.2</td>
<td>3.29</td>
<td>3.42</td>
<td>3.49</td>
<td>3.74</td>
</tr>
<tr>
<td>0.3</td>
<td>3.54</td>
<td>3.85</td>
<td>3.88</td>
<td>4.35</td>
</tr>
<tr>
<td>0.4</td>
<td>3.82</td>
<td>4.45</td>
<td>4.34</td>
<td>5.13</td>
</tr>
<tr>
<td>0.5</td>
<td>4.13</td>
<td>5.31</td>
<td>4.91</td>
<td>6.21</td>
</tr>
</tbody>
</table>

### Table 2. Slot transmission efficiency for VISC, VSC, VYD, and VSCL (\( R_m = 0.1, R_l = 0.2 \))

<table>
<thead>
<tr>
<th>( R_m )</th>
<th>VISC</th>
<th>VSC</th>
<th>VYD</th>
<th>VSCL</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>0.7713</td>
<td>0.7677</td>
<td>0.7646</td>
<td>0.7581</td>
</tr>
<tr>
<td>0.2</td>
<td>0.7143</td>
<td>0.7018</td>
<td>0.699</td>
<td>0.6845</td>
</tr>
<tr>
<td>0.3</td>
<td>0.6599</td>
<td>0.6332</td>
<td>0.6363</td>
<td>0.6091</td>
</tr>
<tr>
<td>0.4</td>
<td>0.6075</td>
<td>0.5621</td>
<td>0.5686</td>
<td>0.5338</td>
</tr>
<tr>
<td>0.5</td>
<td>0.5572</td>
<td>0.4886</td>
<td>0.5063</td>
<td>0.4586</td>
</tr>
</tbody>
</table>

## 4. CONCLUSION

This paper presents design/analysis of Chinese spoken dialogue system using computer simulation. Based on this approach, appropriate strategy can be selected considering the performance goal and trade-off among performance metrics. New dialogue strategies can also be designed and numerically verified in this way.

## 5. REFERENCES


