Incremental Dialog Processing in a Task-Oriented Dialog

Fabrizio Ghigi[1], *Maxine Eskenazi*[2], *M. Ines Torres*[1], *Sungjin Lee*[2]

[1] Dpto. Electricidad y Electrónica, Universidad del País Vasco, Bilbao, Spain
[2] Language Technologies Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania
fabrizio.ghigi@ehu.es, max@cs.cmu.edu, manes.torres@ehu.es, sungjin.lee@cs.cmu.edu

Abstract

Incremental Dialog Processing (IDP) enables Spoken Dialog Systems to gradually process minimal units of user speech in order to give the user an early system response. In this paper, we present an application of IDP that shows its effectiveness in a task-oriented dialog system. We have implemented an IDP strategy and deployed it for one month on a real-user system. We compared the resulting dialogs with dialogs produced over the previous month without IDP. Results show that the incremental strategy significantly improved system performance by eliminating long and often off-task utterances that generally produce poor speech recognition results. User behavior is also affected; the user tends to shorten utterances after being interrupted by the system.

Index Terms: spoken dialog systems, incremental dialog processing

1. Introduction

Traditional Spoken Dialog Systems (SDS) assume that conversations observe strict human-system turn-taking [1], [2]. Yet human conversations are highly interactive, with many interruptions and turn overlaps (See Figure 1). Humans process what they hear little by little, using many sources of information to determine when is appropriate to speak [3]. To model this behavior, the system must be able to incrementally process the user utterance, barging in if needed.

Incremental Dialog Processing (IDP) enables an SDS to generate a response before an utterance ends by processing each minimal input unit of user speech as it is captured by the system [4] then triggering a system action as soon as some conditions are verified [5]. In this way, we can trigger a system response without waiting for the end of the user turn.

In previous IDP work, the goal is usually focused on increasing the naturalness of an interaction in a micro domain. In this paper we use IDP in a real application to prevent incorrect input from being incorporated in dialog state estimation. We found, in real world, task-oriented dialog data, that many long utterances are often incorrectly recognized and understood. This consequently results in users getting off track

U1: The bus is late but .
U2: *Uhmm, yeah.*
U1: I think it will arrive in few minutes
U2: *Hmmm.*
U1: we won't be late.
U2: *Yeah, I suppose.*

Figure 1: Example dialog with backchannels and turns overlapping.

and eventual task failure. More than 70% of the long utterances in this task were misrecognized by the system[1]. Additionally, task success for dialogs containing incorrectly recognized long utterances was low (~45%). This analysis also revealed that we can get most of the information from the early part of the utterance. Thus we can improve task success and system responsiveness by choosing an optimal barge-in point. Figure 2 shows an effective barge-in point. Interrupting the user after the first time he says "64C", saves the time needed to listen to, and recognize the rest of the utterance. Then we inhibit the system from recognizing "1C" and "54C", in the second part of the user utterance. It is, however, non-trivial to select optimal barge-in timing. This would minimize user annoyance and incorrect input while maximizing correct input. To achieve a balance, we tuned the system to choose the barge-in point from a set of points where user input maintains a stable semantic parse with some following pause.

Section 2 describes previous work on IDP. The analysis of the long utterances contained in the corpus collected by the system in the past years is in Section 3. Section 4 describes our IDP strategy. In Section 5 we describe real user studies. In Section 6 we present our conclusions and propose future work.

U: I want the 1C .
S: The 1C. Did I get that right?
U: The 64C, I said the 64C, not 1C, I want the 64C.

```
t1. The 64C
t2. The 64C, I said
t3. The 64C, I said the 64C
t4. The 64C, I said the 64C, not 1C
t5. The 64C, I said the 64C, not 1C, I want
t6. The 64C, I said the 64C, not 1C, I want the 54C.
           ↑     ↑        ↑      ↑    ↑       ↑
           t1    t2       t3     t4   t5      t6
```

S: The 64C. Did I get that right?
U: Yes, that's right.

Figure 2: In this example t1 is an effective cut-in point. We can interrupt the user at t1 without losing important information and saving time needed to listen and recognize the whole utterance.

2. Related Work

IDP was previously tested in micro-domains [6], using incremental prosodic analysis, a reactive connection between ASR and TTS, and a fully incremental architecture. Results

[1] An utterance lasting over 4 seconds is defined to be long.

show that IDP improves system reactiveness. Additionally, humans found the dialog to be more natural despite the fact that there had been no changes in task success rate [7]. Further work focused on integrating IDP with POMDPs [8]. In this case each partial result was treated as a final result, which triggers a belief state update. When integrating IDP with POMDPs, there are substantial improvements in the interpretation accuracy of the incremental results. In order to predict and complete the user utterance using partial results, [9] trained various natural language understanding models for varying partial result lengths. The system could react when confident that results would not improve in the last part of the utterance. IDP had not yet been tested in a real world system with the goal of improving task success, the goal of this paper.

Systems that use lexical entrainment can modify the behavior of the user [11]. That system produces words and expressions, as well as prosodic modifications, that can be mimicked by the user. In the present case, we believe that we can similarly influence users to produce shorter, more easily recognizable utterances. Having the system barge in on the user's turn should lead the user to imitate the system and produce more concise answers. Interrupting the user during off-task speech should, in turn, help the user produce more on task speech.

3. Corpus Analysis

Analysis of a set of 100 real user dialogs and their transcriptions (between October 2008 and September 2009) reveals 4 groups of long utterances:

- user speaks off-task, no useful information added (10.83%).
- user adds several pieces of information in a single turn (21.67%).
- user repeats the same information several times, often following a system misrecognition on the previous turn (21.25%).
- noisy utterances: the system did not detect the utterance endpoint due to background noise (46.25%).

We distinguish between dialogs with every long utterance correctly recognized from dialogs with at least one long misrecognized utterance. We compared correctly recognized

	Correct	**Incorrect**
Avg. Length (sec)	4.41	5.01
Avg. Length (words)	4.96	5.72
Avg. #Values in parse	2.61	2.67
Avg. Turn Number	13.33	13.21
#Long Utt. Per Dialog	1.49	2.21

Table 1: Comparison between Correct Long Utterances and Incorrect Long Utterances.

	DLU	**DILU**	**DCLU**
Task Success	59.00%	45.68%	78.95%
Avg. Length	40.28	42.43	31.1

Table 2: Task Success rate and Average Length for Dialogs with Long Utterances (DLU), Dialogs with some Incorrect Long Utterances (DILU) and Dialogs with only Correct Long Utterances (DCLU).

long utterances to long utterances containing recognition errors in Table 1. We observed no significant difference between correct and incorrect long utterances in terms of average length, number of values, or position of the utterance in the dialog. The only significant difference was the average number of long utterances repeated in a dialog (1.49 vs. 2.21). Misrecognized long utterances tend to be followed by another misrecognized long utterance. The user tends to repeat a long utterance if the previous one was wrongly recognized. Repetition, rewording and other phenomena like shouting or hyperarticulation have also been investigated for those long utterances. We note that most of the long utterances (68.02%) were pronounced in a *normal* manner, that is, without repetition, rewording, shouting or hyperarticulation.

Dialogs with correctly recognized long utterances, however, have a high estimated success rate (78.95%). That estimate drops to 45.68% if at least one long utterance is misrecognized, as reported in Table 2. This indicates that recognizing long utterances correctly is critical to system performance. Additionally, these recognition errors increase dialog length as user and system work on error recovery.

We divided long utterances temporally into 3 evenly-divided parts. About 60% of the information contained in those long utterances is found in the beginning or the middle of the utterance, as shown in Table 3.

Correct Slot Position	
Beginning	30.16%
Middle	28.74%
End	41.09%

Table 3: The position of the semantic items in the utterance, based on the timestamp of the partial results.

Incorrect Slot Distribution	
Beginning	36.60%
Middle	47.62%
Ending	40.33%

Table 4: Percentage of incorrect slots in each part of the long utterances.

Table 4 shows the percentage of incorrect slots in each part of the utterance. The lowest percentage of incorrect slots is the beginning of the utterance (36.60%). Thus the most reliable information is obtained without the last part of the utterance.

4. Incremental Dialog Processing Strategy

A conventional Interaction Manager (IM) coordinates turn-taking [10] by determining when the user utterance starts and when that utterance has ended. The latter action triggers the beginning of the system response. An end-of-utterance decision is made in three cases. The first case is if signal power as detected by the Voice Action Detection module has been below a threshold for more than a given time. The second case is if the module has been waiting for a partial hypothesis for too long (indicating a problem with the sensors). Finally, end of utterance may be related to a pause that was longer than a given threshold. On top of this decision process, our IDP strategy determines when to interrupt the user's utterance and initiate the system's response based on the following decision process.

4.1. Decision Parameters

The crucial 4 parameters, for determining the optimal barge-in point, are:

Last N Partials Equals. This checks the parses of the latest N partials received by the IM. It is set to true if the last N partial results sent by the semantic parser have the same parse. This means that in the last N partials the semantic representation of what the user said did not change. This could be due to the fact that the user is not adding any useful information, possibly due to off-task speech.

Utterance Duration. This represents the minimum duration of the user turn before the system provokes a barge-in. The system should interrupt the user if long utterances are detected, since they have been shown to cause errors and system delays.

Confidence Score. This checks the partial result confidence score. Barge-in is taken into consideration only if the partial result has been recognized with sufficiently high confidence.

Final Pause Duration. This determines if a sufficient pause is present at the end of a partial result. A short pause could simply indicate the end of a word or hesitation. A longer pause is a good place to barge in.

4.2. Decision Process

Given the decision parameters, the decision process works as follows:

1. When the first partial result for the current utterance arrives, we record the initial timestamp and parse.
2. Then we receive a partial result every ~175ms. We check if the last N partials are the same, and then measure the duration of the utterance up to this partial result by computing the difference between the current timestamp and the initial timestamp of the utterance. Finally we retrieve the confidence score and the following pause duration for the current partial.
3. If the last N partials have the same parse and the duration of the utterance, the confidence score and the final pause duration of these partials are higher than the threshold values, we exit the loop and we send a signal to the input device to indicate the termination of input processing[2]. The Dialog Manager produces a new action and the corresponding system prompt is generated, interrupting the user turn.

Depending on the values used for these parameters, IDP can be more or less aggressive, increasing or decreasing system responsiveness. For example, if we want IDP to set an aggressive strategy we can use just the last 2 partials and a low confidence score. If, on the other hand, we do not want to interrupt the user too often, we can mandate that a larger number of partials must have the same parse, or we can use a higher confidence score. Increasing the thresholds for the parameters will thus produce fewer barge-ins. The parameter tuning process for our real world application is described in the next section.

[2] The system prompt triggered by the IDP must be non-interruptible. Otherwise, if the user is still speaking after the system has barged in, a user barge-in is immediately generated and the rest of the system prompt is prevented to be heard.

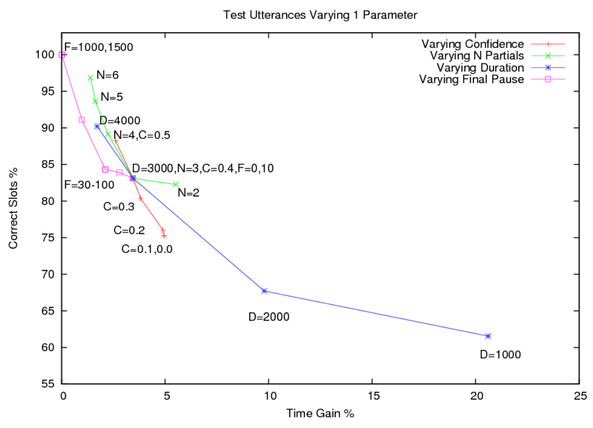

Figure 3: How the percentage of correct slots and time gain vary.

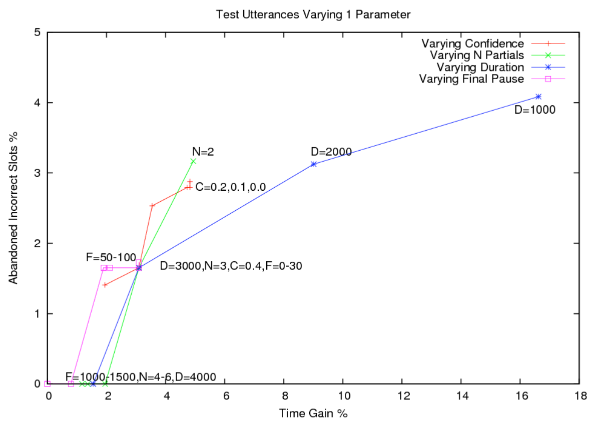

Figure 4: How the percentage of abandoned incorrect slots and time gain vary.

4.3. Parameter Tuning

In order to choose the best values for the parameters, we processed the log files from the real application corpus data mentioned above. The logs recorded every partial ASR result received by the IM. We applied the IDP strategy to each utterance, determining whether the system should have interrupted the user. In this way we can compute, for each different set of values assigned to the decision parameters, how much time we could have saved for an utterance and how many slots we could have recognized if we had interrupted the user before the end of the utterance.

We performed a coordinate-wise search to see how different configurations affect response time and the percentage of (in-)correct slots. The trade-off between response time and the percentage of (in-)correct slots is illustrated in Figures 3 and 4.

For our study, we used the last 2 partials, with a 3000ms minimum utterance duration, a 0.4 confidence score threshold and a 10ms final pause duration; these values give us ~83% of the slots spoken by the user with a time gain of ~6%.

By varying the parameters from these values, we saw a significant loss in either response time or information quality.

5. Experiments

In order to evaluate IDP in real time with real users, we deployed it on the live system that provides schedules and route information to the general public. We let it run with real users for the entire month of October 2013, and then compared the results obtained by the system using the IDP strategy to the results collected by the former system over the whole month of September 2013. We collected 1708 dialogs for October 2013 and 1837 dialogs for September 2013.

5.1. Metrics

To evaluate the improvements, we use the following metrics.
Observed Dialog Success Rate (ODSR). Measures the success of the system in providing the user with the information requested.
Average Dialog Length (ADL). The average number of turns in a dialog. This is used to check whether the IDP strategy produced variations in dialog length.
Average Dialog Duration (ADD). The average dialog duration in seconds. It also checks how the IDP strategy affects the dialog length.
Average Number of Long Utterances (ANLU). The average number of long utterances per dialog. It checks if IDP has an effect on user behavior when it interrupts long utterances.
Long Utterance(s) After Long Utterance (LUALU[*N*]). This measures how many times the user produced a long utterance in N turns after the first long utterance. It is used to evaluate how IDP affects user behavior in the short term. We expect to see the user producing less subsequent long utterances.

5.2. Results

In this section, we show the results obtained using IDP for the month of October 2013 and compare them to the dialogs collected using the previous system for the month of September 2013. 32.96% of the dialogs collected during the experiments contain at least one turn where end-of-utterance was determined by the IDP strategy. The average length of turns where IDP was activated is 3448.5ms.
The results presented in Table 5 clearly show that using the IDP strategy to deal with long user utterances significantly improves system performance. This improvement is reflected in the increased dialog success rate in the ODSR metric (66.67% vs.74.34%). The Z-test of the difference between success proportions shows that this difference is statistically significant (p-value < 0.05) for the ODSR metric. Notably, dialog duration in terms of number of turns and in terms of time in seconds is slightly increased (18.76 vs. 21.15 turns for ADL, 153.99 vs. 162.68 seconds for ADD). The Z-test of the difference of means shows that the difference between ADL means is statistically significant (p-value < 0.05) whereas the difference between ADD means is not (p-value = 0.1074). However, in this case the F upper one-tailed test rejected the hypothesis for equality of both variances with F = 1.268, which is slightly greater than the critical value of 1.138 for a confidence of 0.95. Although we have shortened single utterance duration by interrupting the user, the overall dialog length slightly increased because every time we barged in, the user had to say what the system had not allowed her to say previously. Also, for each user turn a system turn is added to the dialog. This trend results in slightly longer dialogs, although individual user utterances are shorter on average.
ANLU per dialog decreased from 2.93 to 2.21. Users repeat a long utterance immediately after the first long utterance only 18% of the time (LUALU1). This further decreased to 2% (LUALU2) and 0% (LUALU3) in the second and third utterances, respectively. This result indicates that IDP influenced users to produce more concise answers.

Listening to the audios where the system barged in on the user utterance, we observe that this behavior is well-accepted by the users. ASR errors force users to repeat the same information until the system recognizes it correctly. The user response to system barge-ins is to stop speaking and carefully listen to what the system has to say. Frequent barge-in and turn overlaps are common in a human-human conversation, so the enhanced system initiative seems natural to the users.

Metric\Period	Sep2013	Oct2013
ODSR	66.67%	74.34%
ADL	18.76	21.15
ADD	153.99	162.68
ANLU	2.93	2.21
LUALU1	25%	18%
LUALU2	6%	2%
LUALU3	2%	0%

Table 5: Evaluation results: the first column shows the results without IDP, the second column with IDP.

6. Conclusions & Future Work

In conclusion, we have tested an Incremental Dialog Processing strategy on a real user bus information system, and compared two months of real user data. The system is enriched with the chance to take the initiative and barge into the user's speech if the utterance lasts too long and the semantic representation of the utterance has not changed. This strategy slightly increases dialog duration due to the higher number of shorter turns, but this small increase in duration is compensated by the significant improvement in dialog success rate. The IDP strategy used here is completely independent of the Dialog Manager, thus the new Interaction Manager enhanced with the IDP strategy can be reused in any SDS without changes to the Dialog Manager. Future work will include testing this strategy on other domains, possibly in different languages, and the exploration of new parameters to determine the stability of the partial results, like the variance of the confidence score. We also plan to measure user satisfaction on the dialogs obtained with the incremental strategy.

7. Acknowledgements

This work has been partially supported by the Spanish Ministry of Science under grants TIN2011-28169-C05-04, BES-2009-028965 and EEBB-I-13-06006, and by the Basque Government under grants IT685-13 and S-PE12UN061.
We would like to thank also James Valenti for his help in setting up the live system.

8. References

[1] J. Allen, G. Ferguson, and A. Stent, "An architecture for more realistic conversational systems", en *Proceedings of the 6th international conference on Intelligent user interfaces*, 2001, pp. 1–8.

[2] C. Howes, M. Purver, P. G. Healey, G. Mills, and E. Gregoromichelaki, "On incrementality in dialogue: Evidence from compound contributions", *Dialogue Discourse*, vol. 2, n.º 1, pp. 279–311, 2011.

[3] H. H. Clark, *Using language*, vol. 4. Cambridge University Press Cambridge, 1996.

[4] K. Sagae, G. Christian, D. DeVault, and D. R. Traum, "Towards natural language understanding of partial speech recognition results in dialogue systems", en *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, 2009, pp. 53–56.

[5] A. Kilger and W. Finkler, "Incremental generation for real-time applications", Saarländische Universitäts - und Landesbibliothek, Postfach 151141, 66041 Saarbrücken, 1995.

[6] G. Skantze and D. Schlangen, "Incremental dialogue processing in a micro-domain", en *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, 2009, pp. 745–753.

[7] O. Buß, T. Baumann, and D. Schlangen, "Collaborating on utterances with a spoken dialogue system using an isu-based approach to incremental dialogue management", en *Proceedings of the 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, 2010, pp. 233–236.

[8] E. O. Selfridge, I. Arizmendi, P. A. Heeman, and J. D. Williams, "Integrating incremental speech recognition and pomdp-based dialogue systems", en *Proceedings of the 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, 2012, pp. 275–279.

[9] D. DeVault, K. Sagae, and D. Traum, "Can I finish?: learning when to respond to incremental interpretation results in interactive dialogue", en *Proceedings of the SIGDIAL 2009 Conference: The 10th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, 2009, pp. 11–20.

[10] A. Raux and M. Eskenazi, "A Multi-Layer Architecture for Semi-Synchronous Event-Driven Dialogue Management", en *Automatic Speech Recognition and Understanding Workshop (ASRU), 2007 IEEE*, 2007, pp. 514–519.

[11] J. Lopes, M. Eskenazi, and I. Trancoso, "Automated two-way entrainment to improve spoken dialog system performance", en *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, 2013, pp. 8372–8376.