Turn-taking phenomena in incremental dialogue systems

Hatim Khouzaimi
Orange Labs
LIA-CERI, Univ. Avignon
hatim.khouzaimi@orange.com

Romain Laroche
Orange Labs
Issy-les-Moulineaux
France
romain.laroche@orange.com

Fabrice Lefèvre
LIA-CERI, Univ. Avignon
Avignon
France
fabrice.lefevre@univ-avignon.fr

Abstract

In this paper, a turn-taking phenomenon taxonomy is introduced, organised according to the level of information conveyed. It is aimed to provide a better grasp of the behaviours used by humans while talking to each other, so that they can be methodically replicated in spoken dialogue systems. Five interesting phenomena have been implemented in a simulated environment: the system barge-in with three variants (resulting from either an unclear, an incoherent or a sufficient user message), the feedback and the user barge-in. The experiments reported in the paper illustrate that how such phenomena are implemented is a delicate choice as their impact on the system's performance is variable.

1 Introduction

A spoken dialogue system is said to be incremental when it does not wait until the end of the user's utterance in order to process it (Dohsaka and Shimazu, 1997; Allen et al., 2001; Schlangen and Skantze, 2011). New audio information is captured by an incremental Automatic Speech Recognition (ASR) at a certain frequency (Breslin et al., 2013) and at each new step, the partial available information is processed immediately. Therefore, the system is able to replicate a rich set of **turn-taking phenomena** (TTP) that are performed by human beings when talking to each other (Sacks et al., 1974; Clark, 1996). Replicating these TTP in dialogue systems can help to make them more efficient (e.g. (El Asri et al., 2014)) and enhance their ability to recover from misunderstandings (Skantze and Schlangen, 2009).

Several contributions already explored different TTP like end-point detection (Raux and Eskenazi, 2008), backchannels (Meena et al., 2014; Visser et al., 2014), feedback (Skantze and Schlangen, 2009) or barge-in (Selfridge et al., 2013; Ghigi et al., 2014). However, these studies have been performed separately with no unified view and no comparison of respective merits, importance and co-influence of the different TTP. In order to have a better grasp on the concept of turn-taking in a dialogue and a guideline for the implementation, we felt the need to introduce a *taxonomy* of these TTP. Our motivation is to clarify which TTP are interesting to implement given the task at hand. As an illustration, five TTP (which we assume have the best properties to improve the dialogue efficiency) have been implemented and compared in a slot-filling simulated environment.

Section 2 introduces the TTP taxonomy and Section 3 describes the simulated environment, the experimental setup and the results. We then conclude in Section 4.

2 Turn-taking phenomena taxonomy

In linguistics and philosophy of language, a distinction is made between two different levels of a speech act analysis: *locutionary acts* and *illocutionary acts* (Austin, 1962; Searle, 1969). Loosely speaking, a *locutionary act* refers to the act of uttering sounds without taking their meaning into account. When the semantic information is the object of interest, it is an *illocutionary act*. In (Raux and Eskenazi, 2009), four basic turn-taking transitions are presented: *the turn transitions with gap*, *the turn transitions with overlap*, *the failed interruptions* and *the time outs* where only the mechanics of turn-taking are studied at a locutionary level. In (Gravano and Hirschberg, 2011), the authors propose a *turn-taking labeling scheme*, which is a modified version of the original *classification of interruptions and smooth speaker-switches* introduced in (Beattie, 1982). This classification is richer than the one in (Raux and Eskenazi, 2009) as the meaning of the turn-taker utterance

Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pages 1890–1895,
Lisbon, Portugal, 17-21 September 2015.

Table 1: *Turn-taking phenomena taxonomy. The rows/columns correspond to the levels of information added by the floor giver/taker. The phenomena in black have been implemented in the simulator.*

	T_REF_IMPL	**T_REF_RAW**	**T_REF_INTERP**	**T_MOVE**
G_NONE	FLOOR_TAKING_IMPL			INIT_DIALOGUE
G_FAIL	FAIL_IMPL	FAIL_RAW	FAIL_INTERP	
G_INCOHERENCE	INCOHERENCE_IMPL	INCOHERENCE_RAW	INCOHERENCE_INTERP	
G_INCOMPLETE	BACKCHANNEL	FEEDBACK_RAW	FEEDBACK_INTERP	
G_SUFFICIENT	REF_IMPL	REF_RAW	REF_INTERP	BARGE_IN_RESP
G_COMPLETE	REKINDLE			END_POINT

is taken into account. From a computational point of view, it is more interesting to add high-level information to classify these behaviours as semantics clearly influence turn-taking decisions (Duncan, 1972; Gravano and Hirschberg, 2011). In this paper, a more fine-grained taxonomy of TTP is introduced where utterances are considered both at locutionary and illocutionary levels.

During a floor transition, the person who starts speaking will be called T (Taker) whereas the person that was speaking just before will be called G (Giver). At the beginning of the dialogue, the person that initiates the dialogue will be called T and the other G by convention. We classify TTP given two criteria: the *quantity of information that has been injected by G* before the floor transition (rows in Table 1) and the *quantity of information that T tries to add by taking the floor* (columns in Table 1). Table 2 gives the meaning of the different criteria's labels.

Table 2: *Taxonomy labels*

G_NONE	No information given
G_FAIL	Failed trial
G_INCOHERENT	Incoherent information
G_INCOMPLETE	Incomplete information
G_INSUFFICIENT	Insufficient information
G_SUFFICIENT	Sufficient information
G_COMPLETE	Complete utterance
T_REF_IMPL	Implicit ref. to G's utterance
T_REF_RAW	Raw ref. to G's utterance
T_REF_INTERP	Reference with interpretation
T_MOVE	Dialogue move (with improvement)

At the beginning of the dialogue (G_NONE), T can implicitly announce that she wants to take the floor by using hand gestures or by clearing her throat for instance (FLOOR_TAKING_IMPL), or she can directly initiate the dialogue (INIT_DIALOGUE). If G is already speaking, her message can be not understandable by T (G_FAIL). T can warn G implicitly by frowning for example (FAIL_IMPL) or explicitly, in a raw manner by saying *Sorry?* (FAIL_RAW) or by pointing out what has not been understood (FAIL_INTERP). In addition, even if the meaning of the message has been understood, it can be incoherent with the interaction context (G_INCOHERENT, e.g. trying to book a flight from a city with no airport). Again, T can warn G implicitly (INCOHERENCE_IMPL) or explicitly, either by explaining the reason of the problem (INCOHERENCE_INTERP) or not (INCOHERENCE_RAW).

In the case G's utterance is not problematic but yet incomplete (G_INCOMPLETE), T can let her understand that she understands what has been said so far by performing a BACKCHANNEL (*Yes*, *uhum* etc.), by repeating his words exactly (FEEDBACK_RAW) or by commenting them (FEEDBACK_INTERP), for example: *Yesterday I went to this new Chinese restaurant in town... / Yeah Fing Shui / ...and it was a pretty good deal*). If G utters enough information to move the dialogue forward (G_SUFFICIENT), T can refer to an element in G's utterance implicitly (*Aha*) by reacting at the proper timing (REF_IMPL), or explicitly in a raw (REF_RAW, for example *Ok, Sunday*) or interpreted manner (REF_INTERP, for example *Yeah, Sunday is the only day when I am free*). T can also interrupt G to add some information that is relevant to the course of the dialogue (BARGE_IN_RESP). Finally, she can wait until G has finished his utterance (G_COMPLETE) and warn him that he should add more information (REKINDLE, for example: *And?*) or start a new dialogue turn (END_POINT).

In the rest of this paper, five incremental TTP that are the more used in general, and therefore studied, have been tested in a simulated environment: FAIL_RAW (Ghigi et al., 2014), INCOHERENCE_INTERP (DeVault et al., 2011), FEEDBACK_RAW (Skantze and Schlangen, 2009) and BARGE_IN_RESP from both sides, user (Selfridge et al., 2013) and system interruptions (DeVault et al., 2011), along with

INIT_DIALOGUE and END_POINT that already exist in traditional systems.

3 Simulation

3.1 Service task

A personal agenda assistant task has been implemented in the simulated dialogue system (referred to as the *service* hereafter). The user can add events to her agenda as long as they do not overlap with existing events. She can also move events in the agenda or delete them (ADD, MODIFY and DELETE actions). An event corresponds to a title, a date, a time slot, a priority, and the list of alternative dates and time slots where the event can fit, in the case the main date and slot are not available. For example: {**title:** *house cleaning*, **date:** *January* 6^{th}, **slot:** *from 18 to 20*, **priority:** *3*, **alternative 1:** *January* 7^{th}*, from 18 to 20*, **alternative 2:** *January* 9^{th}*, from 10 to 12*}.

3.2 User simulator

3.2.1 Overview

The architecture of the User Simulator (US) is built around five modules: the Natural Language Understanding (NLU) module, the Intent Manager, the Natural Language Generator (NLG), the Verbosity Manager, the ASR Output Simulator, and the Patience Manager. These modules are described in the following.

NLU module: The NLU module is very simple as the service's utterances are totally known by the US and no parsing is involved. Each one of them is associated with a specific dialogue act.

Intent Manager: The Intent Manager is somehow the brain of the US, as it determines its next intent given the general goal and the last NLU result. The general goal depends on the scenario at hand, which is in turn determined by two lists of events: the initial list (*InitList*) and the list of events to add to the agenda during the dialogue (*ToAddList*). The Intent Manager tries to add each event from the latter given the constraints imposed by the former. If the events of both lists cannot be kept, those with lower priorities are abandoned or deleted until a solution is reached.

The service asks for the different slot values in a mixed initiative way. At first, the user has the initiative in the sense that she is asked to provide all the slot values in the same utterance. If there is still missing information (because the user did not provide all the slot values or because of ASR noise), the remaining slot values are asked for one by one (system initiative).

NLG module: The NLG figures out the next sentence to utter given the current Intent Manager's output. A straightforward sentence is computed, for example, *Add the event meeting Mary on July* 6^{th} *from 18:00 until 20:00*.

Verbosity Manager: The Verbosity Manager randomly expands the NLG output with some usual prefixes (like *I would like to...*) and suffixes (like *please*, *if possible...*). Also, a few sentences are replaced with off-domain words or repeated twice as it is the case in real dialogues (Ghigi et al., 2014). For questions concerning a specific slot, neither prefixes nor suffixes are added.

Patience Manager: When the dialogue lasts too long, the US can get impatient and hang up. The US patience corresponds to a threshold on each task duration. It is randomly sampled around a mean of 180 seconds for the experiments. A speech rate of 200 words per minute is assumed for the dialogue duration estimation (Yuan et al., 2006). Moreover, a silence of one second is assumed at each regular system/user transition and a two second silence is assumed the other way round. For interruptions and accurate end-point detection, no silence is taken into account.

3.3 ASR Output Simulator

The US can run either in a *traditional mode* in the sense that it provides a complete utterance to the system then it waits for a response, or in an *incremental mode* where a growing utterance is outputted at each new word. For example: *I*, *I want*, *I want to*, *I want to add*...etc. In incremental dialogue systems, the turn increment (called the *micro-turn* in this case) could be different than the word (a small duration for example).

The ASR output simulator can be used in both modes, but as the traditional mode is a special case of the incremental one, we describe the latter only. This module computes a noisy version of each word (substitution, deletion, or insertion). It also associates a confidence score with each new partial utterance. Moreover, a word in the ASR output can change later as new words pop in (Selfridge et al., 2011; McGraw and Gruenstein, 2012). In the following, this mechanism is referred to as the *ASR instability*. At each micro-turn, the sys-

tem can keep listening to the US or decide to take the floor (see Section 3.4).

3.4 Scheduler

The same architecture as in (Khouzaimi et al., 2014) is used. A *Scheduler* module is inserted between the service and the user simulator. As the ASR output utterance grows, the partial utterances are sent, at each micro-turn, to the Scheduler. In turn, the latter transfers them to the service and waits for its responses.

The aim of this module is to make turn-taking decisions. Given the last system's response and some other features and rules determined by the designer, or learned from data, the Scheduler decides whether or not to convey that response to the client immediately or not.

3.5 Dialogue example

In the following example, the user has to delete an event, before adding another one (ASR noise is not introduced here):

SYSTEM: Hi. Welcome to your agenda management service. How can I help you?

USER: I would like to add the event birthday party on January 6^{th} from 6 pm to 11 pm if it is possible.

SYSTEM: The time slot from 6 pm to 11 pm on January 6^{th} overlaps with the event house cleaning on January 6^{th} from 7 pm to 9 pm. How can I help you?

USER: Delete the event house cleaning.

SYSTEM: Ok. So you want to delete the event house cleaning that takes place on January 6^{th} from 7 pm to 9 pm. Right?

USER: Yes.

SYSTEM: Got it. Do you want to perform some other action?

USER: Yes.

SYSTEM: Add the event birthday party on January 6^{th} from 6 pm to 11 pm.

USER: Ok. So you want to add the event birthday party on January 6^{th} from 6 pm to 11 pm. Right?

USER: Yes.

SYSTEM: Got it. Do you want to perform some other action?

SYSTEM: No.

USER: Thank you. Bye!

3.6 TTP implementation

Replicating some turn-taking phenomena like backchannels makes the system seems more realistic (Meena et al., 2014). In this work, the focus is on dialogue efficiency, therefore, the following TTP have been chosen for the implementation: FAIL_RAW, INCOHERENCE_INTERP, FEEDBACK_RAW and BARGE_IN_RESP from the user's and the system's point of view.

At each micro-turn, the system has to pick an action among three options: to wait (WAIT), to retrieve the last service's response to the client (SPEAK) or to repeat the word at position $n-2$ (if n is the current number of words, because of the ASR instability) in the current partial request (REPEAT). To replicate each selected TTP, a set of rules have been specified to make the proper decision. We review the triggering features related to each TTP accommodated to the task at hand (agenda filling).

FAIL_RAW: Depending on the last system's dialogue act, a threshold relative to the number of words without detecting a key concept in the utterance has been set. In the case of an open question (where the system waits for all the information needed in one request), if no action type has been detected after 6 words, a FAIL_RAW event is declared. The system waits for 3 words in the case of a yes/no question, for 4 words in the case of a date and for 6 words in the case of slots (some concepts need more words to be detected and the user may use additional off-domain words).

INCOHERENCE_INTERP: This event is useful to promptly react to partial requests that would eventually lead to an error, not because they were not correctly understood, but because they are in conflict with the current dialogue state. If such an inconsistency is detected, the system waits for two words (ASR instability) and if it is maintained, it takes the floor to warn the user.

FEEDBACK_RAW: If at time t, a new word is added to the partial utterance and the ratio between the last partial utterance's score and the one before last (which corresponds to the score of the last increment) is lower than 1/2, then the system waits for two words (because of the ASR instability), and if the word is still in the partial utterance, a REPEAT action is performed.

BARGE_IN_RESP (System): This TTP depends on the last system dialogue act as it determines which kind of NLU concept the system is

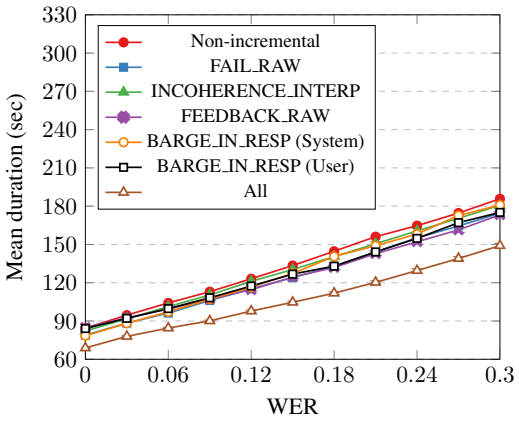

Figure 1: Simulated dialogue duration for different noise levels

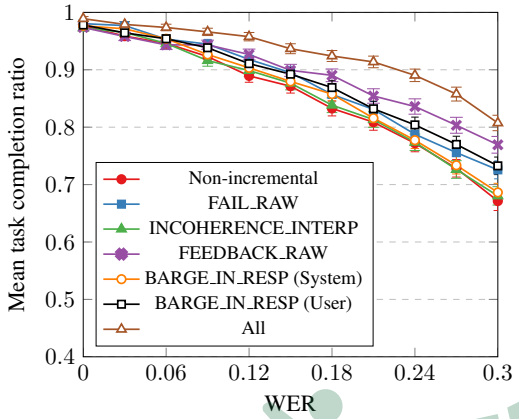

Figure 2: Simulated dialogue task completion for different noise levels

waiting for. Once it is detected, the system waits for two more words (ASR instability) and if the concept is maintained, it performs a SPEAK.

USER_BARGE_RESP (User): This event is triggered directly by the user (no system decision is involved). For each system dialogue act, the moment when a familiar user would barge-in is manually defined in the simulator.

Dialogue duration and task completion are used as evaluation criteria. The task completion rate is the ratio between the number of dialogues where the user did not hang up (because of her patience limit) and the total number of dialogues.

The five implemented TTP have been tested single-handled and in an aggregated manner (referred to as *All* strategy). They have also been been compared to a non-incremental baseline (see Figure 1 and 2). Three dialogue scenarios and different WER levels were tested. For each strategy and each WER, 1000 dialogues have been simulated for each scenario. Figure 1 (resp. Figure 2) represents the mean duration (resp. the mean task completion), with the corresponding 95% confidence intervals, for the different strategies and for WER varying between 0 and 0.3.

The FEEDBACK_RAW strategy performs best whereas INCOHERENCE_INTERP does not improve over the baseline. This is due to the fact that the system has to deal with an open slot (which set of possible values is not closed and known a priori): the event's description. The system mostly performs ADD actions, so the description slot can take any value and is never compared with existing data. This is the case of many application like message dictation for example. However, in the case of service at hand, an initial concept must be detected (the action), therefore, FAIL_RAW improves the performance. BARGE_IN_RESP from user's side is also useful here as dialogues can be long and may contain repetitive system dialogue acts. The users get familiar with the systems and may infer the end of the system's question before it ends. Obviously it is questionable that users may be patient enough (up to several minutes) to achieve such simple tasks in real life. But for the sake of the simulation it was necessary to generate dialogues long enough to have the studied TTP influence them. In a next step, increasing the service capacities (and complexity) will remedy that as a side effect. Finally, BARGE_IN_RESP from the system's side does not bring any improvement either which is due to the fact that in this task and because of input noise, in most cases, the response to the initial open question is not enough to fill all the slots. The responses to single-slot questions do not contain suffixes which explains the inefficiency of the last strategy (the US stops speaking as soon as the slot value is given).

4 Conclusion and future work

This paper introduces a new taxonomy of turn-taking phenomena in human dialogue. Then an experiment where five TTP are implemented has been run in a simulated environment. It illustrates the potentiality of the taxonomy and shows that some TTP are worth replicating in some situations but not all. In future work, we plan to perform TTP analysis in the case of real users and to optimise the hand-crafted rules introduced here to operate the floor management in the system (when to take/give the floor and according to which TTP scheme) by using reinforcement learning (Sutton and Barto, 1998; Lemon and Pietquin, 2012).

References

James Allen, George Ferguson, and Amanda Stent. 2001. An architecture for more realistic conversational systems. In *6th international conference on Intelligent user interfaces*.

J.L. Austin. 1962. *How to Do Things with Words*. Oxford.

Geoffrey Beattie. 1982. Turn-taking and interruption in political interviews: Margaret thatcher and jim callaghan compared and contrasted. *Semiotica*, 39:93–114.

Catherine Breslin, Milica Gasic, Matthew Henderson, Dongho Kim, Martin Szummer, Blaise Thomson, Pirros Tsiakoulis, and Steve Young. 2013. Continuous asr for flexible incremental dialogue. In *ICASSP*, pages 8362–8366.

Herbert H. Clark. 1996. *Using Language*. Cambridge University Press.

David DeVault, Kenji Sagae, and David Traum. 2011. Incremental interpretation and prediction of utterance meaning for interactive dialogue. *Dialogue and Discourse*, 2:143–170.

Kohji Dohsaka and Akira Shimazu. 1997. A system architecture for spoken utterance production in collaborative dialogue. In *IJCAI*.

Starkey Duncan. 1972. Some signals and rules for taking speaking turns in conversations. *Journal of Personality and Social Psychology*, 23:283–292.

Layla El Asri, Remi Lemonnier, Romain Laroche, Olivier Pietquin, and Hatim Khouzaimi. 2014. NASTIA: Negotiating Appointment Setting Interface. In *Proceedings of LREC*.

Fabrizio Ghigi, Maxine Eskenazi, M Ines Torres, and Sungjin Lee. 2014. Incremental dialog processing in a task-oriented dialog. In *Fifteenth Annual Conference of the International Speech Communication Association*.

Agustín Gravano and Julia Hirschberg. 2011. Turn-taking cues in task-oriented dialogue. *Comput. Speech Lang.*, 25(3):601–634.

Hatim Khouzaimi, Romain Laroche, and Fabrice Lefèvre. 2014. An easy method to make dialogue systems incremental. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*.

Oliver Lemon and Olivier Pietquin. 2012. *Data-Driven Methods for Adaptive Spoken Dialogue Systems*. Springer Publishing Company, Incorporated.

Ian McGraw and Alexander Gruenstein. 2012. Estimating word-stability during incremental speech recognition. In *Proceedings of the INTERSPEECH 2012 Conference*.

Raveesh Meena, Johan Boye, Gabriel Skantze, and Joakim Gustafson. 2014. Crowdsourcing street-level geographic information using a spoken dialogue system. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*.

Antoine Raux and Maxine Eskenazi. 2008. Optimizing endpointing thresholds using dialogue features in a spoken dialogue system. In *SIGDIAL*.

Antoine Raux and Maxine Eskenazi. 2009. A finite-state turn-taking model for spoken dialog systems. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, pages 629–637.

Harvey Sacks, Emanuel A. Schegloff, and Gail Jefferson. 1974. A simplest systematics for the organization of turn-taking for conversation. *Language*, 50:696–735.

David Schlangen and Gabriel Skantze. 2011. A general, abstract model of incremental dialogue processing. *Dialogue and Discourse*, 2:83–111.

John Searle. 1969. *Speech Acts: An Essay in the Philosophy of Language*. Cambridge University Press, UK.

Ethan O. Selfridge, Iker Arizmendi, Peter A. Heeman, and Jason D. Williams. 2011. Stability and accuracy in incremental speech recognition. In *Proceedings of the SIGDIAL 2011 Conference*.

Ethan Selfridge, Iker Arizmendi, Peter Heeman, and Jason Williams. 2013. Continuously predicting and processing barge-in during a live spoken dialogue task. In *Proceedings of the SIGDIAL 2013 Conference*.

Gabriel Skantze and David Schlangen. 2009. Incremental dialogue processing in a micro-domain. In *ACL*.

Richard S. Sutton and Andrew G. Barto. 1998. *Reinforcement Learning, An Introduction*. The MIT Press, Cambridge, Massachusetts, London, England.

Thomas Visser, David Traum, David DeVault, and Rieks op den Akker. 2014. A model for incremental grounding in spoken dialogue systems. *Journal on Multimodal User Interfaces*.

Jiahong Yuan, Mark Liberman, and Christopher Cieri. 2006. Towards an integrated understanding of speaking rate in conversation. In *INTERSPEECH Proceedings*.