Importance-Driven Turn-Taking for Spoken Dialogue Systems

Ethan Oliver Selfridge
B.A. Psychology, Reed College, 2006

Presented to the
Center for Spoken Language Understanding
within the Oregon Health & Science University
School of Medicine
in partial fulfillment of the
requirements for the degree
Doctor of Philosophy
in
Computer Science & Engineering

December 2013

© Copyright 2013 by Ethan Oliver Selfridge
All Rights Reserved

Center for Spoken Language Understanding
School of Medicine
Oregon Health & Science University

CERTIFICATE OF APPROVAL

This is to certify that the Ph.D. dissertation of

Ethan Oliver Selfridge

has been approved.

Dr. Peter A. Heeman, Thesis Advisor
Research Associate Professor

Dr. Jason D. Williams
Researcher, Microsoft Research

Dr. Brian Roark
Research Scientist, Google

Dr. Izhak Shafran
Associate Professor

Dr. Alison Presmanes Hill
Research Assistant Professor

Dedication

To Stephanie

For her unwaivering support, patience, and inspiration.

Acknowledgements

First and foremost, I want to thank my advisor, Dr. Peter A. Heeman, for his excellent guidance, advice, and support. I also want to thank Dr. Jason D. Williams for his superb contribution to my training and for his excellent insights over the years. I also want to thank Iker Arizmendi for his patience and advice along the way. I would not be where I am today without these three excellent professionals. Finally, I want to thank my parents who have given me so much love and support. Thank you.

Contents

List of Tables

List of Figures

Turn-taking is a critical aspect to any spoken dialogue system as it governs the timing and nature system utterances. Standard turn-taking approaches are overly rigid and unprincipled, only seeking to mimic the surface features of human turn-taking instead of addressing the underlying motivation behind the behavior. Here, we introduce *Importance-Driven Turn-Taking* which combines the importance of speaking with a variable strength turn-taking signal using reinforcement learning. We describe and evaluate theoretical, architectural, and practical aspects of the approach in both simulation and in live user experiments. Overall, we find that the importance-driven approach is more efficient than traditional methods as it can exhibit more flexible behaviors that can accommodate different types of users.

Chapter 1

Introduction

The next-generation of widely used speech systems will expand the simple command interface commonly used today into a more interactive experience. One necessary component of human–machine interaction is turn-taking, the process underlying the transition of speaking roles in a dialogue. It is through turn-taking that the interaction becomes a dialogue, rather than a single command that the system either does or does not understand. Even this limited “one-shot” interaction can be greatly improved by the addition of turn-taking; instead of dialing the wrong contact by mistake, the system could confirm the command by *taking the turn* and responding to the user. It is obvious, though not always apparent, that turn-taking plays a critical role in any speech system.

Since the system's goal is to have an effective and natural interaction, it is reasonable to base the system's turn-taking off that of humans. Sacks, Schlegoff, and Jefferson [50] characterize human turn-taking as “smooth”, meaning there are minimal overlaps and silences between turn transitions. Sacks et al. elaborate on this concept, stating that the speaker chooses when to release the turn by using specific predictive positions in their speech. Though not a central theme, Sacks et al. also posit that the turn is “interactively determined.” We take this to mean that the turn assignment emerges from a type of negotiation between conversants. Duncan and Niederehe [15] provide evidence for this negotiative process, finding that the strength and combination of a conversants turn-cues such as gesture and falling pitch is directly related to the likelihood of them having the turn. This use of varying turn-cue strength was also found by Yang and Heeman [85], who showed that volume was found to be reliable predictor of turn conflict outcomes. Yang and Heeman also found that the number of initiative conflicts grew as the conversant's contribution became more important. Similarly, Walker and Whittaker [76] reported that people will interrupt to correct a misunderstanding. These two studies show that, in some situations, the importance of the utterance is the underlying motivation for turn-taking behavior.

Many current turn-taking approaches are based on the Sacks et al. model of “smooth” transitions. We call this the *Keep-Or-Release* design, where the System will not attempt to take the turn unless it is sure the human has released it. Breaking from simple silence threshold approaches, sophisticated methods have recently been developed to predict this turn release using decision theory [46, 7], reinforcement learning [30], decision trees [52] and others. In the keep-or-release framework, this prediction is critical because it allows the system to begin speaking directly after the user is finished, thus minimizing silence between turns.

On the other hand, other approaches enable the system to intentionally interrupt the user. This decision usually takes one of two forms. Either the user has said something wrong and the system produces a *mid-utterance interruption* to correct (e.g. [1]), or the system produces an early onset once it has determined the user’s intention (e.g. [37]). Most overlapping methods are dependent on the use of incremental speech recognition, which provides recognition results during user speech.

Neither of these approaches model the rationale underlying the system’s turn-taking behavior. Due to this lack of motivation, the system’s behavior can become overly constrained and lead to awkward, unnatural, and ineffective turn-taking. For example, the keep-or-release systems are weak when the timing of the contribution is critical to successful and efficient task completion. Consider a food ordering task where the system’s purpose is to take the user’s order. If the user orders an item which is unavailable or requires additional specifications, the system must wait until the user is completely finished with their turn before informing them. On the other hand, overlapping system’s can interrupt before the user speaks and facilitate a more efficient dialogue. However, as uncertainty regarding the user’s intention may be quite high, this mid-utterance interruption behavior, while superficially advanced, may be quite error prone as well.

Furthermore, current approaches commonly ignore the variable nature of the system’s *own* turn-taking signal and fail to accommodate any of turn-taking's negotiative elements as proposed by Sacks et al.. For example, in the food ordering task the system can potentially give the user a large amount of information. How does the system decide how to structure this potentially very long turn to invite the user to take the turn without disregarding user's who want the information? While it is true that barge-in allows users to speak at any time, this is a rather unprincipled capability whose existence necessarily violates the "smooth" turn-taking principle that is being applied to the system.

Problem Statement: Turn-taking is a major challenge for spoken dialogue systems. We argue that this is primarily caused by a failure to provide an underlying rationale to govern system turn-taking behavior. Furthermore, this failure forces the system to make a binary turn-taking decision, causing the system to miss much of turn-takings negotiative element.

Thesis Statement: Based on the turn-taking literature, we hypothesize that using utterance importance as the underlying rationale to guide turn-taking behavior will lead to more natural and efficient interactions. Our approach, *Importance-Driven Turn-Taking*, provides a mechanism to direct turn-taking signals of variable strength, using reinforcement learning to determine the utterance and the strength of the turn-taking signal with respect to dialogue success as whole. Specifically, we define importance implicitly as the turn-taking signal determined by reinforcement learning. This structure frees the system of designer-imposed turn-taking constraints and enables the system to use a variable strength turn-taking signal, facilitating a negotiative process.

1.1 Aims

This thesis is broken into five aims, beginning in a theoretical presentation and ending in a practical implementation and evaluation in the real-world. We describe each aim and then highlight the contributions to the field.

Aim 1: Here, we give a complete description and preliminary evaluation of the Importance-Driven Turn-Taking model. Using an artificial domain we demonstrate the use of Reinforcement Learning to learn turn-taking and utterance behavior simultaneously. Utterance onset is used as the turn-taking signal and the first to speak after every utterance "wins" the turn. This leads to the turn being interactively determined and the conversant with the more pertinent utterance often gets the turn. We find importance-driven dialogues to be more efficient when compared against keep-or-release methods. We then explore the importance-driven approach, examining its performance in a rich, discourse linguistically motivated domain. We also show that one can further decompose the utterance decision to one of a sub-goal and a specific utterance itself. This decomposition results in far simpler dialogue systems, and we compare and contrast turn-taking behaviors with different sub-task selection strategies. Chapters 3, 4, and 5 cover this aim.

Aim 2: The practical aspects of increasing the contextual awareness of systems are considered. Since the importance of the utterance is reliant on the current complete context of the dialogue, it is desirable for the system to have access to user speech before they are finished. This relies on Incremental Speech Recognition. We first present a method for determining when to output incremental speech recognition results and the use of logistic regression to predict the stability and accuracy of partial results. We then describe the Incremental Interaction Manager, which enables strictly turn-based dialogue managers to be used with incremental results, and demonstrate how it can be integrated into statistical spoken dialogue system. Finally, we combine these techniques with aspects of importance-driven turn-taking and propose and evaluate a method to improve barge-in processing in a live dialogue task. Chapters 6 and 7 cover this aim.

Aim 3 We tackle the issues surrounding the development and training of temporally relevant behavior. We describe a temporal simulation framework that models not only the timing and content of conversant speech, but also the output of an incremental speech recognizer and voice activity detector. We propose and evaluate multiple methods to synthesize incremental results for the purpose of training and evaluating turn-taking approaches, and show how this approach can be used to simulate the timing of user utterances. Chapter 8 covers this aim.

Aim 4 The decision making components of a live importance-driven turn-taking system are outlined. We propose the *Tempest* architecture, which applies separate reinforcement learning agents to the turn-taking and utterance decisions. This semi-hierarchical multi-agent framework reduces the search space by increasing the generalizability of the policies and can be seamlessly integrated simultaneously into a live dialogue system and the Temporal Simulator described in Chapter 8. Furthermore, we highlight the *utterance* policy differences that arise from having different turn-taking capabilities. The first part of Chapter 10 covers this aim.

Aim 5 Here, we provide a live evaluation of importance-driven turn-taking, applying nearly all of the techniques described up to this point. Using a live Hands-Free email task, we compare hand-crafted strategies with and without incremental processing against importance-driven methods with and without incremental processing. We find that the importance-driven method is more efficient since it can handle multiple user types. This is the same result as found in the first aim, validating our earlier analysis. We also find high utility for mid-utterance interruption, validating and extending previous findings. The latter portion of Chapter 10 covers this aim.

1.2 Contributions

Aim 1: The first contribution is the importance-driven turn-taking model, which relies on reinforcement learning to implicitly judge utterance importance by learning optimal turn-taking behaviors. The second contribution is the demonstration and evaluation of the importance-driven component of the model. While some previous work has used reinforcement learning for turn-taking [30], it has not been used to optimize turn-taking for the dialogue as a whole, particularly in the context of a negotiative turn-taking process. The third contribution is a mature mixed-initiative simulation domain that is based off a number of psycholinguistic phenomena such as implicit acceptance and retraction. Finally, we split utterance and sub-task selection into distinct actions to be optimized separately, and illustrate how changes to sub-task selection effect turn-taking behavior.

Aim 2: The contributions of this aim primarily focused on improving incremental speech processing components and proposing behavioral models that exploit such components. Previous approaches, in general, use a delay in order minimize revisions that are common in incremental speech recognition (ISR), and use specialized incremental dialogue managers. Our contributions build on this. We first contribute an method that interpolates different types of partial results to improve the stability and accuracy of incremental speech recognition results. We then show a method to predict the accuracy and stability of these results. In a similar vein, we also present an approach to use incremental results with non-incremental dialogue managers. These techniques are applied to the challenge of barge-in processing and we propose and evaluate the Prediction-based Barge-in Response (PBR) Model that continuously predicts the eventual understanding of the barge-in input. In the case of predicting a successful understanding, the system suspends the prompt. On the other hand, if non-understanding is predicted or determined then the system either resumes or continues the prompt. In a live dialogue task with real users, we find this method to be far more effective and efficient than standard approaches to barge-in processing.

Aim 3: Here, the contributions lie in developing methods to train temporally sensitive dialogue systems. First, we propose a temporal simulator that models the timing and content of user and system speech as well as that of an incremental recognizer and voice-activity detector. All previous methods of turn-taking related simulations have focused on simulating prosody *not* content. Here we focus on the content, while allowing for the simulation of prosody if desired. The first version of the temporal simulator models incremental recognition quite simply, the reference text being incrementally passed to the dialogue manager after passing through an error simulator. However, for training real systems something a bit more realistic is required. We outline a method of simulating speech recognition by adding noise to synthesized speech before recognition. We then describe how this method can be extended to synthesizing incremental speech recognition results. While this is a *very* simple approach, we have not found any research explicitly examining its use.

Aim 4: The Tempest dialogue architecture and semi-hierarchical multi-agent framework are novel contributions to the field. The former is a complete dialogue system specification that can be easily reproduced and the latter is new approach to training utterance and turn-taking policies. Furthermore, we concretely show that the optimal *utterance* policy is substantially different when the system's turn-taking is more flexible than when it is not. This implies that the utterance and turn-taking policy should be learnt simultaneously to assure optimality.

Aim 5: The contribution of aim 5 is the evaluation of a live importance-driven dialogue system. The system is trained with varying user models in the temporal simulator. We compare multiple versions of the importance-driven systems with one that uses the keep-or-release turn-taking model. We find that the importance-driven approach is more efficient as it can accommodate different user turn-taking behavioral patterns whereas the KR system does not have this flexibility and must cater to the most basic and simple turn-taking behaviors.

Chapter 2

Related Work

2.1 Introduction

This chapter reviews the concepts and related work that underly this thesis. Section 2.2 reviews the basic structure of spoken dialogue systems. Section 2.2.2 deals with statistical approaches to dialogue management, emphasizing the popular reinforcement learning approach. Section 2.3 reviews human–human turn-taking. Section 2.4 describes Incremental Speech Recognition and its application to dialogue. Section 2.5 reviews current approaches to human–machine turn-taking that emphasize smoothness, whereas Section 2.6 describes approaches that emphasize overlap, including user barge-in. Finally, Section 2.7 addresses the work on simulation, for both turn-taking and speech recognition.

2.2 Spoken Dialogue Systems

Most spoken dialogue systems have a very simple pipeline. User speech is decoded by the speech recognizer, which passes ranked speech hypotheses, known as an n-best list, to a natural language understanding module. Once the speech is interpreted, it is passed to the dialogue manager, which decides what to say given the new input. The new dialogue move is translated into natural language and spoken to the user.

2.2.1 Dialogue Management

Dialogue management, the choice of what to say in a given dialogue context, has been the focus of research for quite some time. Recently, the Information State Update (ISU) approach has become increasingly popular [33]. This approach uses a number of rule sets to update an *Information State* that drives the decisions. Each rule set has a number of rules, consisting of a precondition and an effect. If the precondition is met the effect is "fired" and the Information State is updated. The modular nature of ISU makes system development simple and efficient since rule sets can easily be defined and introduced. For simple dialogues, using ISU to "hand-craft" the dialogue manager works quite well. However, as the complexity of the dialogue system grows and the number of states increases, statistical approaches are far more effective. These will be discussed in Section 2.2.2.

Initiative and Attention

The role of the dialogue manager is to decide what to say in a given dialogue context. When handling speech recognition errors, this decision is generally guided by thresholds that determine whether to reject the recognition, explicitly confirm it, implicitly confirm it, or accept it. When the recognition is accepted, the dialogue manager's decision is constrained by the initiative of the interaction: System initiative, where the system will direct the interaction; user initiative, where the user will; and mixed-initiative, where both conversants can take the initiative in the conversation [65, 75, 3].

We find it useful to combine these three different interaction patterns into a simple spectrum. On one end are the fixed-initiative tasks, the system-initiative and user-initiative interaction, where conversants have minimal contributional freedom. Consider a task where the system can only ask the user for input. Obviously in this system-initiative case, the user has no real freedom to contribute to solving the task. Similarly, many current mobile speech interfaces are user-initiative: the system's only response to user direction is to perform the command or not understand. In the middle of the spectrum is minimal mixed-initiative interaction. For instance, a planning task where the user, in general, has the initiative can become mixed-initiative if the system is allowed to notify the user of some problem with the plan. At the other end of the initiative spectrum is complex mixed-initiative interaction where conversants have far more contributional freedom. Here the role assignments are not fixed and either conversant can initiate a sub-dialogue or topic change. Systems inhabiting this end of the spectrum are far more sophisticated than those designed for fixed-initiative dialogue.

One concept highly related to initiative is *attention* or sub-task selection. In a fixed-initiative setting, only the initiator can select the sub-task to work on. For example, a car buying task has a number of sub-tasks such as color, make, and mpg. In a user initiative setting, the user might choose to work on the color sub-task first. Here, the system can only answer the user's question (e.g. "Is there a pink car?", "Yes"). If the task became slightly mixed, the system could respond with question of its own ("Is there a pink car?", "What about red?"). In a completely mixed-initiative interaction the system could completely disregard the user's question and initiate a new sub-task ("Is there a pink car?", "What mpg do you want?")[1]. The initiative spectrum corresponds to how large of a shift in sub-task the non-initiator can have, and the dialogue manager selects actions based on this constraint. Note that the action can be an utterance within the context of a sub-task ("Yes", "What about red?"), or a new sub-task and new utterance entirely ("What mpg do you want?"). This distinction has sometimes been blurred in the literature.

Many studies have equated sub-task selection and utterance selection. Levin et al. [35] describe a dialogue system that has just one utterance available per slot per state, and learns the best utterance/attention to use in that dialogue context. Georgila et al. [31] compared learning the order of slots to fill with using a reward constrained order. Similar to Levin, each slot had just one utterance available at one time for each slot. They found that the learning agent performed *worse* than the penalized one and argue that enforcing order is critical in some conditions.

Other work has rightly treated attention decisions as a distinct action but have failed to allow the system to switch sub-tasks before one is completed. Cuayhuitl et al. [11] give a description of a hierarchical reinforcement learning structure where attention is treated as a distinct action. However, the sub-task decisions are heavily constrained by a reward structure that forces a specific order and the agent can only change sub-tasks upon completion of its current one. Scheffler and Young [54] also enforce a strict order

[1]Note that according to Walker [74], the System would be implicitly answering "Yes" to the User's question of color.

on sub-task completion. Lemon et al. [34] take a different approach to attention, using a System that identifies and works on a task supplied by the User. All of these studies would fall somewhere in the middle of the initiative spectrum.

Attention decisions and initiative level impact the system's turn-taking behavior tremendously and the recent literature has not optimized turn-taking, utterance selection, and attention independently. Experiments in Chapter 5 address this.

2.2.2 Statistical Dialogue Management

This section briefly describes the current research on statistical dialogue management pertaining to this thesis. As we have described above, the goal of a dialogue manager is to select the dialogue move that is most appropriate for the current dialogue state. Since the state space grows considerably as the domain increases in complexity, statistical approaches to this decision have gained traction in recent years. Above all, Reinforcement Learning (RL) based approaches have become popular. As opposed to supervised learning approaches such as perceptrons, which require an input and output pair for learning, RL only requires a *reward signal.* This is ideal for dialogue systems, where the performance of the dialogue system can only be fully assessed after the dialogue has finished. Using this delayed reward, Reinforcement Learning can determine the optimal action for a given dialogue context.

Reinforcement Learning with Markov Decision Processes

Reinforcement Learning seeks to learn the cumulative expected reward, or Q-score, for each state $s \epsilon S$ and action $a \epsilon A$ pair [69]. An optimal policy is a mapping between s and a, where a leads to the highest expected return. Commonly, Q-scores are learned by balancing actions that have not led to the highest reward (so far) and actions that have. Using one approach, ϵ-*greedy* search, the system the $argmax_a Q(s, a)$ action with 1-ϵ probability and another action with probability ϵ. The value of ϵ is generally quite a low number such as 0.2, so the system follows the current best policy the majority of the time. Following some learning episode, the reward is used to update the Q-score estimates for each state. In general, this is either done with Temporal Difference learning, a dynamic programming algorithm, or Monte Carlo, a straight forward update using only the states and rewards seen during the particular episode.

The state space must be formulated as a Markov Decision Process (MDP) to update Q-scores properly. An MDP assumes that the transition and reward probability from some $s_t a_t$ pair is completely contained by that pair and is unaffected by the history $s_{t-1}a_{t-1}, s_{t-2}a_{t-2}, \ldots$. For the MDP assumption to be met, care is required when deciding which features to use for learning. If it is not met, Monte Carlo should be used, as it implicitly maintains the state transitions found in the episode and so is resistant (though not immune) to the detrimental effects of minor MDP violations.

Reinforcement Learning based dialogue management has gained wide-spread popularity since its introduction in 2000 [35]. The idea is that dialogue can be formalized as the requisite MDP, with the dialogue context and dialogue move being the state and action, respectively. It is simple to integrate RL and ISU. Using the preconditions to limit the search space of the learning agent, a condensed and compresse d subset of the IS — the Reinforcement Learning State — is used to learn which proposed action to take [25]. Recently, the popularity of the MDP approach has led to the use of Partially-Observable Markov Decision Processes (POMDPs) in order to cope with the inherent uncertainty of recognition results [82]. RL has, to name but a few, been used to learn appropriate referring expressions [29], handle young and old users [22], and select prompts for a tourist information system [36]. This basic approach has also been extended to handle the uncertainty present in the speech signal through Partially-Observable MDPs [82], and reduce the state space using semi-MDPs and hierarchical reinforcement learning [10]. One approach that has not received as much attention in the dialogue community is multi-agent RL [8], where separate MDPs cooperated or compete in some environment. A critical issue of cooperative multi-agent RL is the instability introduced by the joint optimization. Two approaches that have been shown to work are explicitly passing decision information between learning agents in a cascade and modularizing the Q-score so that one agent implicitly models the behavior of all others within some perception field [41].

2.3 Understanding Human-Human Turn-Taking

We now review the Human-Human turn-taking research that we feel is influential for turn management. We begin by describing the "smooth" turn-taking model. We then review studies that suggest that turn-taking signals produce a more negotiative process, and highlight work which implies that importance plays a primary role in motivating these signals.

2.3.1 Smooth Turn-Taking

The conversation model proposed by Sacks, Schegloff, and Jefferson [50] has been extremely prominent within the computational turn-taking community. Their model emphasizes the importance of conversational units called Turn Construction Units (TCU). A TCU can be a phrase, clause, sentence, or word, and necessarily has a predicable end; a Transition Relevance Place (TRP). At a TRP, the speaker may select the next person to speak, a listener may self-select and begin speaking, or the speaker may continue speaking.

Duncan [14] also views turn-taking as being speaker-centric and suggests the use of a wide array of turn-cues, offering non-syntactic cues such as gesture, falling pitch, and a change in gaze direction as strong indicators of the turn release. He states that the turn transitions happen smoothly when the speaker gives off a number of turn-yielding cues that the listener reacts to. Duncan proposed that when the listener attempts to take the turn without the presence of turn-yielding cues they are disregarding this mechanism. He creates a dichotomy between regular turn-taking, which relies on these turn-yielding cues, and take-turn attempts, which are without yielding cues.

A number of studies have analyzed turn-cues in the context of the smooth turn-taking model. Ferrer, Shriberg, and Stolcke [17] used decision trees to classify pauses as utterance-final or utterance-medial. The decision trees were trained using both prosodic and N-gram features, and they found that both feature types improved classification over a simple pause threshold baseline. Furthermore, they found that these two feature types were complementary and their combination achieved a relative error rate reduction of 45%. Schlangen [56] also found that prosodic and syntactic cues were able to discriminate between utterance-final and utterance-medial pauses. He also found that for both humans and machines, the prediction of utterance-medial pauses was far more successful than utterance-final. Using only a small number of prosodic features, Ward, Fuentes, and Vega [77] attempted to predict both the pitch and whether the system should be speaking over increasingly longer windows of the future. They achieved modest success on the Switchboard corpus.

Perhaps the best analysis of turn-cues was done by Gravano and Hirshberg [24], who analyze turn-yielding cues with an eye towards spoken dialogue systems. Seeking to confirm Duncan's speculation, Gravano and Hirshberg first investigated seven turn-yielding cues identified from the literature. They found that as the number of turn-yielding cues increased, the probability of a turn-taking attempt (either a smooth transition or pause interruption) increased. They also reported that textual completion, followed by voice quality and speaking rate, were most indicative of turn-taking outcomes. Gravano and Hirshberg next investigated backchannel-inviting cues, where they found a similar combination effect for these particular cues, the most predictive cue being the Part-Of-Speech bigram tag. For overlapping speech they found a pattern of turn-yielding cues similar to those preceding smooth transitions and that most overlaps are early onsets that happen towards the end of the speaker's utterance.

Smooth turn-taking, and studies that support it, emphasize the speaker-centric nature of turn-taking. We refer to it as the Keep-Or-Release model. Since the speaker controls the TRP, they control when another can take the turn. Turn-taking frameworks based on this will only seek to take the turn at a TRP. While this simplifies the turn-taking problem to one of prediction, it also forces it to be overly rigid. Furthermore, the originators of the keep-or-release model have also stated that turn-taking may be far more negotiative than their models of turn-taking suggest. This implies that a speaker-centric approach to computational turn-taking is intrinsically limited.

2.3.2 Negotiation and Motivation in Turn-Taking

Somewhat orthogonal to the speaker-centric nature of the smooth turn-taking model, both Sacks et al. and Duncan also argue that turn-taking is a negotiative process. Sacks et al.

state that “the turn as a unit is interactively determined (p. 727).” Duncan and Niederehe [15] found that turn-cue strength was the best predictor of who won the turn. Similarly, Yang and Heeman [85] found that volume, specifically a relative increase in volume, was a strong indicator of winning the turn in utterance overlap situations. They found that they were able to correctly classify winners 79 % of the time using the higher relative volume. Schlegoff [55] also details the incrementally increasing strength of cues, arguing that the stronger cues will win the turn. This suggests that it is the *relative* difference in strength that predicts the turn-assignment: it is not that the conversant is speaking loudly, it is that they are speaking loudly *relative* to that of the other conversant.

Studies in turn-conflict resolution can give some insight into the motivation behind these turn-taking signals. Schlegoff [55] looked at the overlapping speech as a whole. He states that there is rarely more than two parties in overlaps even when there are many people in the conversation. He also draws a distinction between quickly resolved overlaps, and ones that persist quite longer, which can be characterized as turn-conflicts. The time that the conflict persists is a function of the interest that the conversants have in winning the turn, which is indicates the use of utterance importance.

Yang and Heeman [85] looked at turn-conflicts in the Multi-Threaded Dialogue (MTD) corpus. The MTD corpus is a collection of dialogues in which two humans play two games; one ongoing and task-oriented, and one called the interruption game, which was short, constrained by time, and played during the other game. They found that as the time constraint of completing the interruption game became shorter the number of turn-conflicts grew as well. More specifically, the percentage of conflicts grew from 9% at 40 seconds to 24% for 10 seconds.

Similarly, Walker and Whittaker [76] claim that people will interrupt to remedy some understanding discrepancy. This interruption is of obvious utility and importance to the conversation’s success. The interrupter knows that an interruption *now* will be beneficial to the future. In other words, turn-taking is *not* a decision without a global consequence, and these decisions should be made within the context of the dialogue as a whole.

These findings suggests that turn-taking signals are influenced by the importance of the potential utterance, and that signal variability facilitates a negotiative process that is mostly disregarded by the keep-or-release model.

We have presented studies that argue for smooth, speaker-centric turn-taking as well as evidence that turn-taking is more negotiative in nature. We have also shown that the turn-taking signals used by conversants are motivated by utterance importance. This latter concept is the foundation of Importance-Driven Turn-Taking, which will be presented in Chapter 3.

2.4 Incremental Speech Recognition

We now describe modern speech recognition methodology, the production of partial phrase results, and the advantages and deficiencies of Incremental Speech Recognition (ISR). In this we seek only to provide a topical foundation, and not a comprehensive review. It is quite useful to understand current ISR techniques since many of the turn-taking approaches reviewed in the following sections rely on it.

Most modern speech recognition engines use Hidden-Markov Models and the Viterbi algorithm to decode words from audio. Decoding relies on three models: an acoustic model, which assigns probabilities to speech audio given a phone; a lexicon, which specifies phone sequences for a word; and a language model, which specifies the probability of a word sequence. The aim of the decoding process is to find the N most probable word sequences given these three models and audio input.

Two different forms of language models are commonly used in spoken dialogue systems. A *Rule-based Language Model* (RLM) specifies a list of valid sentences which may be recognized, usually via expansion rules or an explicit context-free grammar. Alternatively, a *Statistical Language Model* (SLM) specifies a vocabulary of words, allowing arbitrary sentences to be formed. Both models specify probabilities over their respective sets — RLMs via whole-sentence probabilities, and SLMs via data-driven probabilities of short word sequences called N-grams. In an SLM, special word symbols are used to represent the beginning and end of the phrase, so the probability of beginning or ending phrases with words can be modeled.

As audio frames (a vector representing 10 ms of sound) are received, the recognizer builds up a *lattice* that compactly describes the probable sequences of words decoded from the audio. In conventional turn-based speech recognition, decoding continues until the user finishes speaking. Once the user has finished, the engine searches the lattice for the most probable word sequence and returns this to the dialogue manager. Conversely, in ISR the engine inspects the lattice *as it is being built*, and returns *partial* results to the dialogue manager as they become available. A key issue for ISR is that partial results may later be revised, because a different path may become the most probable as the lattice is grown by more audio input. In other words, partial results are *unstable* in the sense that they may later be revised. Note that stability is not the same as accuracy: a partial result may be accurate (correct so far) but unstable, because it is later revised. Similarly, a stable result may not be accurate.

Studies have shown that incremental systems react faster than non-incremental ones, and are well-liked by users because of their naturalness [2, 64]. Aist et al. (2007) found that incremental speech recognition yielded 20% faster task completion. Furthermore, Aist also found that adding ISR improved users' satisfaction with the interaction, attributing this improvement to "naturalness": "incremental systems are more like human-human conversation than their non-incremental counterparts." Skantze & Schlangen (2009) observed a similar trend, finding that an incremental system was "clearly preferred" since it "was experienced as more pleasant and human-like", though it did not actually outperform the non-incremental system in a number dictation task.

Some recent work has focused on incremental natural language understanding (NLU). DeVault et al. [13] showed that, when using a relatively small number of possible interpretations, the correct interpretation could be predicted by early incremental results. Schlangen et al. [57] demonstrated that an incremental reference resolver could identify the correct reference 0.5 probability in a task where chance would be 0.12 probability. NLU uses context and other information to be somewhat resilient to errors, and word recognition inaccuracies may not yield a change in understanding.

A number of researchers have described methods for evaluating and improving the stability of ISR results [5, 18]. Baumann, Atterer, & Schlangen spoke directly to stability by comparing partial phrase results against the "final hypothesis produced by the ASR".

They show that increasing the amount of "right context" — the amount of speech after the end of the potential partial result — increases the stability of the partials. Fink et al. (1998) also used a right context delay to decrease the word error rate of ISR results.

Incremental Speech Recognition is critical to natural turn-taking behavior. It underlies the approaches described in the next two sections and it is used in Chapters 6, 8, 7, and 10.

2.5 Smooth Human-Machine Turn-Taking

This section describes recent work on human-machine turn-taking that focuses on minimizing the silence and overlap between speaking turns.

2.5.1 Generic Human-Machine Turn-Taking

Many computational approaches to turn-taking focus on mimicking surface features of human turn-taking behavior, and only a few are cognitively motivated [71]. Inspired by the smooth turn-taking reported in the observational studies on human turn-taking (e.g. Sacks et al., 1974), these approaches emphasize minimal silence and minimal overlap between turns. This emphasis is manifested in what we call the *Keep-Or-Release* design. Here, turn-taking is modeled as the speaker having control of the turn until they explicitly release it, at which point the other conversant will take the turn. Thus, this model requires the speaker to make release-turn and keep-turn signals, and the listener needs to be able to detect these. This is problematic in that turn is not interactively determined, as suggested by both Sacks et al. and Duncan. Moreover, the system operates on the assumption that because it believes the User to be finished it *must* take the turn. We now cover most of the major contributions in this area.

Most work done defining a overall turn-taking mechanism for a dialogue agent is consistent with the Keep-Or-Release approach. Traum and Hinkelman [72] propose a series of turn-taking actions: take-turn, keep-turn and release-turn. The system can only use a take-turn action after it has determined that the user performed a release-turn action. A variant of the three turn-action framework is used by Traum and Rickel [73], who add request-turn as a weaker version of take-turn. In a study using reinforcement learning in a artificial domain, English and Heeman [16] used only two turn-actions: keep and release. These two actions were learned by the two agents as the simulations progressed and neither agent could attempt to take the turn while the other still had it. Systems using the KR approach must detect the user's release-turn before they can speak, but this may difficult since it is sometimes ambiguous whether the speaker has finished or is only pausing. Release-turns are commonly identified in two ways: Either using a silence-threshold [70], or the predictive nature of utterances [50] which we discuss next.

2.5.2 Systems that predict the end of the user's turn

A large subset of the turn-taking work focuses on estimating the probability of a transition occurrence based on some syntactic and prosodic cues. Theoretically speaking, this behavior is consistent with the Keep-Or-Release model, which emphasizes smooth turn-taking. On the practical side, a successful prediction reduces the latency between utterance and reduces the number of overlaps.

Kronlid [32] details the design of a turn-manager. This turn-manager, designed specifically to adhere to the Sacks et al. model, has three primary components. One component handles the environment as a whole, another is concerned with overlap detection, and the last identifies TRPs as defined by Sacks et al.. The agent is designed to stop speaking at overlaps and only begin speaking at TRPs. In his implementation, Kronlid suggests the use of the TRP predictor proposed by Hulstijn and Vreeswijk [27], where the probability of the agent speaking is a ratio between the urgency of the agent to speak and the predicted distance to the TRP. While Kronlid suggests that this urgency may be derived from "the status, self-confidence etc. of each agent", there is no attempt to support this speculation.

Raux and Eskenazi [46] use a decision theory approach to predict appropriate places to take the turn. Following Jaffe and Feldstein [28], the conversational floor is modeled as a 6-state finite-state machine. This model has a free state that operates as a legal transition position. Raux and Eskanazi compute the expected cost of taking the turn based on a cost matrix and the probability that the state is free. The cost is increased as the pause or latency between turns increases, and the probability of the free-state is akin to TRP prediction. They find that the use of this cost function reduces both the latency between utterances and the rate of overlaps. The free-state probability is computed by logistic regression with lexical cues as the most informative feature.

In a multi-modal domain, Bohus and Horvitz [7] also used a decision theoretic approach, modeling three possible outcomes of the system turn-taking decisions: the user releases the turn to the system, the system releases the turn to the user, or there is a "floor transition battle." The probability of each outcome is dependent on three primary factors: the conversational dynamics, the system input processing delays, and system output processing delays. This probability can than be applied to a cost matrix that has been derived from annotator judgments of training data. Bohus and Horvitz evaluate this approach retrospectively by applying it to a corpus of turn transitions, finding that it theoretically reduces the turn-taking cost by over 50%. This is primarily achieved by "avoiding turn-initial overlaps" and "reducing response time following floor releases to the system." This result is similar to that shown by Raux and Eskenazi.

Sato et al. [52] used decision trees trained on corpora to determine whether to take the turn or not when the user pauses, attempting to only take the turn on turn-final pauses. Similar to Raux and Eskenazi [46], their features were mainly derived from prosody, duration, and (partial) recognition results. They also included understanding features that indicated the understanding state of the system. The decision trees were used to classify whether to take the turn or not following a user pause. They were trained on a large corpus of human-machine dialogues that had been annotated for the task. Sato et al. found that these features improved turn-taking accuracy significantly; they were a good predictor of when to take the turn. Upon analyzing the features, they found that the understanding and recognition based features were sufficient to attain high accuracy and that prosodic features (pitch and power) contributed far less.

In contrast to this, Jonsdottir, Thorissson, and Nivel [30] based their work on *content-free* turn-taking proposed in the Ymir Turn-Taking Model [71]. In this model, the content of the system's utterance is not considered for turn-taking decisions. Focusing only prosody, Jonsdottir, Thorissson, and Nivel used Reinforcement Learning to reduce latency and minimize overlap between turns. Using overlap as negative reward and shorter predicted pause duration as positive reward, the learning agent is able to reduce the number of interruptions and average turn taking silence with a individual user after approximately 40 artificial dialogues. Pause duration is used to decide whether to take the turn or not, so a shorter pause duration will lead to shorter latencies but more possibility of overlap. The authors state that since the agent is able to adjust relatively quickly, it can be used to adapt to users during the interaction and facilitate smooth human-like turn-transitions. While this approach is useful towards smoothing turn-transitions, it does not relate these transitions to the content of user speech, which has been shown by the majority of studies to play a central role in turn-taking.

These systems impose a simple and rigid form of turn-taking. They are represented as the baseline systems in Chapter 4 and Chapter 5, and a baseline in Chapter 10.

2.6 Overlapping Turn-Taking Approaches

In contrast to smooth turn-taking methods that penalize overlap, some approaches have been created with overlapping behavior in mind. While these approaches are certainly more flexible than those previously described, results have been mixed. We speculate that this is due to a lack of underlying motivation, which is the primary subject of this thesis.

Aist [1] presents a “Time-Sensitive Conversational Architecture” that provides support for content-driven interruptions. Using an incremental recognizer and a variety of hand-crafted turn-taking rules, the System could interrupt a user, who is reading, to offer a correction or back-channel. While the evaluation and domain is minimal, it appears to be the first instance of content-based turn-taking behavior.

More recently, Nishimura and Nakagawa [40] employed decision trees to choose the appropriate response time. However, instead of only predicting the appropriate place to take the turn, this system chooses the timing of back-channels, repetitions, and collaborative completions. While the *set* of candidate responses is dependent on the content of incremental speech recognition results, the selection of the response and its timing only relies on prosodic features. In contrast to a number of studies emphasizing content for turn-taking (e.g., [52], [46]) the authors report high scores for this prosodic-driven approach when compared to relatively strong baselines. It may be that the nature of the context-specific candidate set helped to overcome the reliance on allegedly insufficient prosody.

Both of these studies report gains from violating the minimal overlap assumption. However, Hirasawa et al. [26] report that system overlap back-channeling was not well liked within the user population. Using a Wizard Of Oz experimental procedure, they found that a system that back-channeled immediately was less easy to speak to, less easy to understand, and less easy to use than a system that performed more "orderly" turn-taking (i.e. no overlaps). However, it may be the case that this back-channel position was quite unnatural and so the scores reflected unfavorable dispositions towards unnatural overlap, and not overlap in general. Interestingly, they report that many user's stopped speaking when the system overlapped. While in this context it was detrimental, it does imply that a system interrupting with a more substantive contribution would likely be successful in taking the turn.

Lu, Nishimoto, and Minematsu [37] use the content of incremental speech recognition to drive turn-taking decisions. Using Reinforcement Learning to determine when to interpret a partial hypothesis, they evaluate three different turn-taking strategies: speak when the final recognition result has been produced, speak after user speech ends using the incrementally determined interpretation, and speak immediately following the, possibly premature, interpretation of of user speech. Using user self-report, they find that while the latter approach is considered much more efficient, it is less attractive than the other approaches. The second approach, which is more efficient than the first, was considered to be the most attractive. This finding lends modest support to Hirasawa et al. [26]: that system overlap should be used sparingly.

Dethfles et al. [12] also employed a reinforcement learning paradigm for understanding and producing back channels using incremental speech recognition. By using the information density of incremental results and a reward structure that guided the system to barge-in when information density was low, they produced a system which was judged favorably by human observers.

These approaches represent the state-of-the-art in turn-taking research. The performance of these systems will be applicable in the Chapter 10 where we evaluate the Importance-Driven approach on real users.

2.6.1 Barge-in

Barge-in is a special case of over-lapping speech processing. Spoken dialogue systems (SDS) communicate with users with spoken natural language; the optimal SDS being effective, efficient, and natural. Allowing input during system speech, known as *barge-in*, is one approach that designers use to improve system performance. In the ideal use case, the system detects user speech, switches off the prompt, and then responds to the user's utterance. Dialogue efficiency improves, as the system receives information prior to completing its prompt, and the interaction becomes more natural, as the system demonstrates more human-like turn-taking behavior. However, barge-in poses a number of new challenges; the system must now recognize and process input during its prompt that may *not* be well-formed system directed speech. This is a difficult task and standard barge-in approaches often stop the prompt for input that will not be understood, subsequently initiating a clarification sub-dialogue ("I'm sorry, I didn't get that. You can say...etc."). This non-understood barge-in (NUBI) could be from environmental noise, non-system directed speech, poorly-formed system directed speech, legitimate speech recognition difficulties (such as acoustic model mismatch), or any combination thereof. Currently back-channels, unless specifically accounted for, fall into this category.

Using the standard barge-in model, the system stops the prompt if barge-in is detected and applies the dialogue logic to the final recognition result. This approach assumes that the barge-in context should not influence the dialogue policy, and most previous work on barge-in has focused on detection: distinguishing system directed speech from other environmental sounds. Currently, these methods are either based on a VAD (e.g. [67]), ISR hypotheses [45], or some combination [48]. Both approaches can lead to detection errors: background speech will trigger the VAD, and partial hypotheses are unreliable [5]. To minimize this, many systems only enable barge-in at certain points in the dialogue.

One challenge in applying a dialogue policy that is interaction context-free is that, in the case of false barge-in, the system will be initiating a clarifying sub-dialogue ("I'm sorry I didn't get that...etc") to non-system directed audio. Since this false barge-in, which in most cases is background noise that is actually speech (e.g. the tv), is highly indicative of poor recognition performance in general, the system's clarifying response can only further degrade the satisfaction of the user.

Strom and Seneff strom2000intelligent provide, to our knowledge, the only mature work that proposed deviating from the dialogue policy when responding to a barge-in recognition. The barge-in recognition evaluation, independent from the dialogue manager, is performed on complete recognitions using confidence and interpretability measures. Instead of initiating a clarifying sub-dialogue, the system produced a filled-pause disfluency ('umm') and resumed the prompt at the phrase boundary closest to the prompt's suspension point. One challenge with this approach is that it only operated on the final speech recognition result so may lead to very long system silence before resumption in very challenging environments.

Chapter 7 describes and evaluates a method that the continuously predicts the understanding success of the barge-in recognition. This novel approach performs better than the standard model in a live spoken dialogue task.

2.7 Simulation for Speech Systems

2.7.1 Turn-Taking Simulation

Researchers often turn to simulation as developing a dialogue system with real users is expensive, time consuming, and sometimes impossible. Some turn-taking simulations have been highly stylized and only model utterance content, failing to give a realistic model of timing [61]. Others have modeled a content-free form of turn-taking and *only* attend to timing and prosodic information [30, 4, 42]. The former is insufficient for the training of deployable real-time systems, and the latter neglect an important aspect of turn-taking: semantic information [24]. We propose a method for simulating dialogue explicitly for improving turn-taking in Chapter 8.

2.7.2 Speech Recognition Simulation

Section 8.3 deals with the subject of synthesizing incremental speech recognition results. It is proposed work. There has been a few studies investigating synthesizing, or “hallucinating”, n-best lists (e.g. [51, 81]). The vast majority of these approaches use a confusion network (phone, word, or phrase) to simulate recognition errors and n-best lists. However, there has been surprisingly little work on simulating ISR with any eye towards reality. The work in Chapter 8 proposes a simple speech synthesis approach and provides a cursory evaluation.

Chapter 3

Introducing Importance-Driven Turn-Taking

3.1 Introduction

We now present the theoretical underpinnings of Importance-Driven Turn-Taking. In this, we seek to describe a conceptual view of how human–machine turn-taking *should* be. We find inspiration in studies of human turn-taking that suggest that the utterance importance plays a crucial role in turn-taking behavior (Section 2.3.2). By directing system behavior with motivation rather than rule, we keep it simple while providing the capacity for far greater behavioral sophistication than can be achieved by traditional top-down approaches.

3.2 Importance-Driven Turn-Taking

Our importance-driven approach has two primary components: a variable strength turn-taking signal and utterance importance. It relies on Reinforcement Learning to define the latter by the strength of the former. Importance may be broadly defined as how well the utterance leads to some predetermined conversational success, be it solely task completion or encompassing a myriad of social etiquette factors. This importance-driven concept springs directly from the contributional freedom of mixed-initiative interaction, in that the ideal system will reason over the utterance's importance to completing the task when determining whether to contribute or not.

3.2.1 A Variable Turn-Taking Signal

According to human turn-taking literature the turn-taking signal is variable, in that there are stronger and weaker signals. For example, a strong turn-cue would be speaking very loudly and a weak turn-cue would be saying nothing.

For purposes of creating a spoken dialogue system that can generate and monitor these signals, we find it useful to categorize them into three types: *timing*, *volume*, and *prosody*. The first involves the onset and cessation of system speech. In the case of system silence, the system is deciding whether to begin speaking now, later or somewhere in between. In the case of system speech, the system is deciding to stop speaking or not. The second, *volume*, is concerned with the loudness of the system prompt. In the case of turn-fights, where both the system and the user are trying to grab the turn the system *may* choose to increase its volume. Though, technically, volume falls under prosody, it is separated since (1) it is such a strong cue theoretically, and (2) it very simply realized practically. All other types of prosody, such as pitch and speaking rate, fall into the third category. These cues can be strong predictors of smooth turn-transitions [24]. In the case of system speech, the system may use them to a varying degree to encourage or dissuade the user from speaking. In the case of system silence, the system is using them to judge the turn-taking signal of the user. Note that the importance-driven model *does not* include the labeling of turn-cues (x is strong, y is weak), rather it assumes the behavior will *emerge* out of reinforcement learning.

3.2.2 An Importance-Driven Decision

The importance-driven framework expands the Keep-Or-Release method by creating a principled situation where the System and User can interactively determine the turn. As we have described previously, much research has shown that turn-cues are indicators of turn-transitions. Current systems are using turn-cues as features to make one decision: Is the User finished or not? Conversely, the importance-driven approach provides a means for the System and User to softly take and release the turn so as to interactively determine who *should* be speaking. If the utterance is very important than it will have very strong turn-cues. Alternatively, if the System's utterance is less important it will have weaker cues and be "out-bid" by the User. In this way, whoever has the most (perceived) important contribution, relative to the alternatives, wins the turn.

One critical component is determining the importance of the utterance. By using a reinforcement learning approach that seeks to optimize overall dialogue success, the importance of the utterance can be defined by the turn-taking signal which is optimal for it. One may use RL in this fashion without regard for the local turn dynamics (minimizing gaps and overlaps) as these will emerge (or not) from optimizing the dialogue overall. Whatever actions lead to a higher return may be thought of as more important than ones that do not. In this way, utterance importance is implicitly and principally determined. One other critical component is the real-time situational context that is required to make these decisions. Incremental Speech Recognition can provide this context. So, given an incremental recognizer and a reinforcement learning framework, the Importance-Driven Turn-Taking approach will learn the appropriate turn-taking signal. Since the utterance decision is *also* being determined by reinforcement learning, the optimal utterance and turn-taking signal for a given dialogue context will be learned simultaneously.

3.3 An Example

While a variety of turn-cues is important, undoubtedly the most critical is timing. Moreover, the timing of the utterance undoubtedly motivates the use of other turn-cues types such as volume in the case of overlap. Here we provide a illustrative example focusing solely on timing.

Previous approaches have shown that using both semantic and prosodic features, a transition point can be predicted. Unlike other approaches, we are not trying to minimize gaps and overlaps. Rather the system (and user) should speak at the optimal time that may *or may not* be immediately following a user utterance. A simple example of two dialogues (ASR and confidence in italics) is shown in Tables 3.1 and 3.2.

While this example is obviously contrived, it illustrates the benefits of variable onset. In the second dialogue, the System chose to delay its response since it expected that the

Table 3.1: Dialogue One

Speaker	Utterance
System	Where are you leaving from?
User	downtown [*'downtown',0.2*]
System	I'm sorry, I didn't get that. Where are you leaving from?
User	downtown [*'downtown',0.6*]
System	Ok, downtown

Table 3.2: Dialogue Two

Speaker	Utterance
System	Where are you leaving from?
User	downtown [*'downtown',0.2*]
-	[System does not respond immediately]
User	downtown [*'downtown',0.6*]
System	Ok, downtown

User will repeat or clarify if the system does not take the turn. Since the User will repeat themselves *anyways*, it increases task efficiency by not explicitly stating that it does not understand until it has given the User a chance to repeat. This entire example illustrates that there may be some utility to varying the response time following a User utterance, and this clearly extends on the functionality of the Keep-Or-Release approach. In other words, by violating the minimal onset assumption, the System can achieve gains in interaction success.

But what about violating the minimal overlap assumption? Obviously people interrupt each other and, as shown by Walker and Whittaker [76], there is often utility to these interruptions. However, as described in Section 2.6, system mid-utterance interruption has been met with questionable success. We argue that System mid-utterance interruption should be considered, yet only in the instances where it is critical to the success of the interaction. In other words, as suggested by the literature on turn-conflicts as described in Section 2.3.2, the decision to interrupt or not should motivated by the importance of the contribution to the dialogue as a whole.

Chapter 4

Evaluating Importance-Driven Turn-Taking in Simulation

4.1 Introduction

The primary goal of this chapter is demonstrate and evaluate a guiding principle — overall dialogue importance — to mediate the system's turn-taking signal. We begin by describing the simulation task and domain, followed by the system and user models. We then present and discuss performance differences between different turn-taking strategies. This chapter is based off Selfridge and Heeman (2010).

Background on human–human turn-taking can be found in Section 2.3. The critical element is that human turn assignment is interactively determined by the use of cues. Computational turn-taking approaches that emphasis smooth turn-taking are found in Section 2.5, and those that allow overlap are found in Section 2.6. A review of dialogue initiative is found in Section 2.2.

4.2 Simulation Domain

Here we show how the importance-driven approach can be implemented for a collaborative slot filling domain. We also describe the Single-Utterance and Keep-Or-Release domain implementations that we use for comparison.

4.2.1 Dialogue Task

We use a food ordering domain with two participants, the system and a user, and three slots: drink, burger, and side. The system's objective is to fill all three slots with the available fillers as quickly as possible. The user's role is to specify its desired filler for each slot, though that specific filler may not be available. The user simulation, while intended to be realistic, is not based on empirical data. Rather, it is designed to provide a rich turn-taking domain to evaluate the performance of different turn-taking designs. We consider this a collaborative slot-filling domain since both conversants must supply information to determine the intersection of available and desired fillers.

Users have two fillers for each slot.[1] A user's top choice is either available, in which case we say that the user has adequate filler knowledge, or their second choice will be available, in which we say it has inadequate filler knowledge. This assures that at least one of the user's fillers is available. Whether a user has adequate or inadequate filler knowledge is probabilistically determined based on user type, which we will describe in Section 4.2.5.

Table 4.1: System and User speech acts

Action	Gloss
System	
query slot	"what kind of burger do you want'?'
inform [yes—no]	"yes we have bacon burgers"
inform available slot fillers	"We have only have a veggie burger."
inform filler not available	"We do not have cheese burgers." bye
"Thank you. Come again."	
User	
inform slot filler	"I want water."
query slot filler	"Do you have milk?"

[1]The choice of two fillers is only to minimize the length of training. We found no substantive difference when the number was increased.

4.2.2 Simulated Speech Acts

The System and the User communicate via speech acts, shown with glosses in Table 4.1. The system has 5 speech actions: *inform [yes/no]*, to answer when the user has asked a filler availability question; *inform filler not available*, to inform the user when they specified an unavailable filler; three *query slot* actions (one for each slot), a query which asks the user for a filler and is proposed if that specific slot is unfilled; three *inform available slot fillers* actions, which lists the available fillers for that slot and is proposed if that specific slot is unfilled; and *bye*, which is always proposed.

The user has two actions. They can inform the system of a desired slot filler, *inform slot filler*; or query the availability of a slot's top filler, *query filler availability.* A user will always respond with the same slot as a system query, but may change slots entirely for all other situations. Additional details on user action selection are provided shortly.

Both agents use an Information State (IS) to indicate possible actions. The IS is made up of a number of variables that model the environment and is slightly different for the system and the user. Shared variables include *QUD*, a stack which manages the questions under discussion; *lastUtterance*, the previous utterance, and *slotList*, a list of the slot names. A few system specific IS variables are *availSlotFillers*, the available fillers for each slot; and three *slotFiller* variables that hold the fillers given by the user. Some user specific IS variables are three *desiredSlotFiller* variables that hold an ordered list of fillers, and *unvisitedSlots*, a list of slots that the user believes are unfilled.

A sample dialogue fragment using the Single-Utterance turn-taking approach is shown in Table 4.2. Notice that in Line 3 the system informs the user that their first filler, "coke", is unavailable. The user then asks asks about the availability of its second drink choice, "sprite" (Line 4), and upon receiving an affirmative action (Line 5), informs the system of that filler preference (Line 6).

Table 4.2: Single-Utterance dialogue

Spkr	Gloss	Speech Act
1 S:	"What kind of drink would you like?"	*query slot*
2 U:	"I want a coke"	*inform slot filler*
3 S:	"We do not have coke"	*inform filler not avail*
4 U:	"Do you have sprite?"	*query filler avail*
5 S:	"Yes we have sprite"	*inform slot*
6 U:	"I want a sprite."	*inform slot filler*
7 S:	"We have cheese burgers."	*inform available slot fillers*

4.2.3 Turn-Taking Approaches

This simulation models three different forms of turn-taking: Single-Utterance, Keep-Or-Release, and Importance-Driven. The Single-Utterance approach has the most rigid turn-taking mechanism. After a speaker makes a single speech act the turn transitions to the listener. Since the turn transitions after every utterance the system must only learn appropriate speech act, not turn-taking behavior. Similarly, the User does not have any turn-taking behavior. The less rigid Keep-Or-Release (KR) model allows the speaker to either keep the turn to make multiple utterances or release it. This allows the freedom of multiple contributions but, as it is the speaker who chooses when the turn should transition, there is no facility for variable strength cues and interactive determination of the turn.

In Importance-driven approach both agents "bid" for the turn after each speech act to determine who will speak next. The system and the user both have 5 different bids, and ties are randomly decided. The bids are *highest, high, middle, low, and lowest.* Each of these correspond to a range of 0.1, beginning at 0.3 to 0.4 for the *highest* bid and go to 0.7 to 0.8 for the *lowest* bid. The inverse relationship between bids and values is meant to be consistent with the conceptualization of utterance onset: *higher* bids should be spoken with less silence. Simply put, each agent generates a bidding action after each utterance and the agent with lowest value (quickest utterance) wins the turn. The distributions that govern the value generation are shown in Figure 4.1. Note that this is very much of an extension of the Keep-Or-Release approach. The main difference is the opportunity to "win" the turn through negotiation by utterance onset.

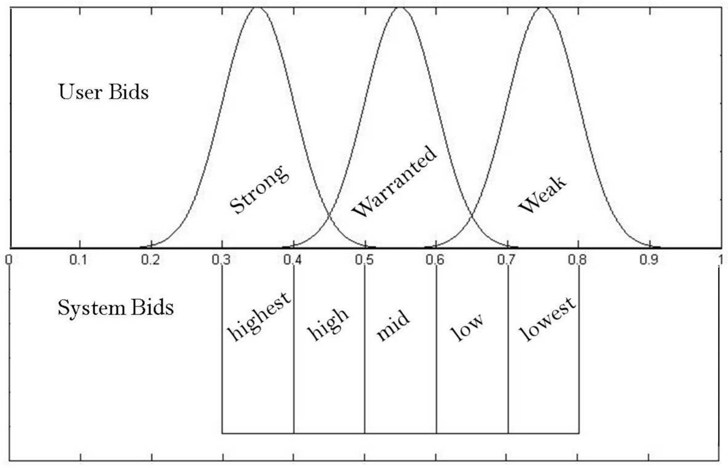


Figure 4.1: Bid Value Probability Distribution

4.2.4 System Behavior

Importance-driven turn-taking relies on reinforcement learning to choose both the utterance and the variable strength turn-taking signal. In order to learn the best action the System requires an objective function and a reinforcement learning state. We use a cost function, which the system attempts to minimize, based on dialogue length and the number of correct slots: $C = NumberofUtterances + 25 \cdot IncorrectSlots$. The RL state for the system has seven variables:[2] QUD-SPEAKER, the stack of speakers who have unresolved questions; INCORRECT-SLOT-FILLERS, a list of slots (ordered chronologically on when the user informed them) that are incorrect and have not been resolved; LAST-SYS-SPEECH-ACTION, the last speech action the system performed; GIVEN-SLOT-FILLERS, a list of slots that the system has performed the INFORM AVAILABLE SLOT FILLER action on; and three boolean variables, SLOT-RL, that specify whether a slot has been filled correctly or not (e.g. Drink-RL).

This RL state (with the requisite action indicator variables) is used for both speech acts and turn-taking. When using the Single-Utterance strategy, the System only learns a speech act policy. For the Keep-Or-Release approach, the System first chooses a speech act with state $s_{speechAct}$. Following its completion, the system then makes a its turn decision

[2]We experimented with a variety of RL States and this one proved to be both small and effective.

using s_{turn}. If it decides to keep the turn it once again selects a speech act. The transition between s_{turn} and $s_{speechAct}$ is zero cost, since cost is only accrued following a speech act. While the use of RL for KR imparts some importance onto the keep-or-release behavior, it does not influence whether the system gets the turn when it did not already have it. This is an critical distinction between KR and the importance-driven method. The latter allows the conversants to negotiate the turn using turn-bids motivated by importance, whereas in KR only the speaker determines when the turn can transition.

The process is slightly more complex for an importance-driven system. While not necessary in simulation, to maintain some semblance of reality both bid and utterance must be chosen before the system acts. The state s_{turn} first selects bid b and then transitions with zero cost to state $s_{speechAct}$ that selects the speech act. One difficulty is that, while the system will select a speech act, it may never occur as the system may not win the turn. In this situation, $s_{speechAct}$ is not reinforced for SA and s_{turn} transitions to s'_{turn}.

4.2.5 User Behavior

We define three different types of users — *Experts, Novices, and Intermediates.* User types differ probabilistically on two dimensions: slot knowledge, and slot belief strength. We define experts to have a 90 percent chance of having adequate domain knowledge, intermediates a 50 percent chance, and novices a 10 percent chance. These probabilities are independent between slots. Note that the dialogue system does *not* have knowledge of the user type besides implicit modeling within the state representation.

Initial slot belief strength is dependent on user type and whether their filler knowledge is adequate (their initial top choice is available). The intuition is that experts should tend to know when their top choice is available, and novices should tend to know that they do not know the domain well. Experts with adequate knowledge have a 70, 20, and 10 percent chance of having Strong, Warranted, and Weak beliefs respectfully. Similarly, intermediates with adequate knowledge have a 50, 25, and 25 percent chance of the respective belief strengths. When these user types have inadequate filler knowledge the probabilities are reversed to determine belief strength (e.g. Experts with inadequate domain knowledge for a slot have a 70% chance of having a weak belief). Novice users always have a 10, 10, and 80 percent chance of the respective belief strengths.

Speech Acts: The user simulation choses speech actions based on a slot belief strength. Slot belief strength represents the user's confidence that it has adequate domain knowledge for the slot (i.e. the top choice for that slot is available). It is either strong, warranted, or weak [9]. If the user is informed of the correct fillers by the system *inform*, that slot's belief strength is set to 3. If the user is informed that a filler is not available, than that filler is removed from the desired filler list and the belief remains the same.[3] Actions are chosen based on slot belief strength: A strong belief will always result in an *inform*, a warranted belief resulting in an *inform* with $p = 0.5$, and weak belief with $p = 0.25$.

Turn-Taking: Users in the KR environment choose whether to keep or release the turn similarly to bid decisions.[4] After a user performs an utterance, it chooses the slot that would be in the next utterance. A number, k, is generated from a Gaussian distribution in the same manner as the ID user's bids are chosen (e.g Strong belief implies a $\mu = 0.35$). If $k > 0.55$ then the user keeps the turn, otherwise it releases it. The users choose their bids based on slot belief strength. Before bidding, a user will determine the slot for the next utterance. The belief strength of that slot then specifies the mean of a normal distribution from which a random number will be generated. Strong belief strength corresponds with a $\mu = 0.35$, warranted with a $\mu = 0.55$ and weak with a $\mu = 0.75$; $\sigma^2 = 0.025$ for all beliefs. The bids are bound by 0 and 1. Computing bids in this fashion leads to, on average, users with strong beliefs bidding 'highest', warranted beliefs bidding in the 'middle', and weak beliefs bidding 'lowest'.

[3]In this simple domain the next filler is guaranteed to be available if the first is not. We do not model this with belief strength since it is probably not representative of reality.

[4]We experimented with a few different KR decision strategies, and chose the one that performed the best.

4.3 Evaluation and Discussion

We now evaluate Importance-Driven turn-taking (ID) in our simulated domain. The three turn-taking approaches are trained and tested in four user conditions: novice, intermediate, expert, and combined. In the combined condition, one of the three user types is randomly selected for each dialogue. We train 10 policies for each condition and turn-taking approach. Policies are trained using Q-learning, and $\epsilon-$*greedy* search with $\epsilon = 0.2$ for 10000 epochs (1 epoch = 100 dialogues, after which the Q-scores are updated). Also, we set the decay $\gamma = 1.0$ and the step size parameter $\alpha = 1/\sqrt{N(s,a)}$ where $N(s,a)$ is the number of times the s, a pair has been seen since the beginning of training. Each policy is then ran over 10000 test dialogues with $\epsilon = 0$, and the mean dialogue cost for that policy is determined. The 10 separate policy values are then averaged to create the Mean Policy Cost (MPC). The MPC between the turn-taking approaches and user conditions are shown in Table 4.3. Lower numbers are indicative of shorter dialogues, since the system learns to successfully complete the task in all cases.

Table 4.3: Mean Policy Cost for Model and User condition [5]

Model	Novice	Int.	Expert	Combined
SU	7.61	7.09	6.43	7.05
KR	6.00	6.35	4.46	6.01
ID	6.09	5.77	4.35	5.52

Single User Conditions: Single user conditions show how well each turn-taking approach can optimize its behavior for specific user populations and handle slight differences found in those populations. Table 4.3 shows that the MPC of the SU model is higher than the other two models, indicating longer dialogues on average. Since the SU system must respond to every user utterance and cannot learn a turn-taking strategy to utilize user knowledge, the dialogues are necessarily longer. For example, in the expert condition, the best possible dialogue for a SU interaction will have a cost of five (three user utterances for each slot, two system utterances in response). This cost is in contrast to the best expert dialogue cost of three (three user utterances) for KR and ID interactions.

The ID turn-taking approach outperforms the KR design in all single user conditions except for Novice (6.09 vs. 6.00). In the this condition, the KR system takes the turn first, informs the available fillers for each slot, and then releases the turn. The user can then inform its available filler easily. The ID system attempts a similar dialogue by using 'highest' bids but sometimes loses the turn when users also bid 'highest.' If the user informs an unavailable filler or queries the dialogue grows longer. This situation is quite rare, as shown by the *very* small difference in performance between both models. In all other single user conditions, the ID approach has shorter dialogues than the KR approach (5.77 and 4.35 vs. 6.35 and 4.46). A detailed explanation of ID's performance will be given in Section 4.3.1.

Combined User Condition: We next measure performance on the combined condition that mixes all three user types. This condition is more realistic than the other three, as it better mimics how a system will be used in actual practice. The system must learn to optimize its behavior for different types of users. The ID approach (MCP = 5.52) outperforms the KR (MCP = 6.01) and SU (MCP = 7.05) approaches. We also observe that the KR outperforms SU. These results suggest that the more a turn-taking design can be flexible and negotiative, the more efficient the dialogues can be.

4.3.1 Importance-Driven Performance:

It is not enough to state that the performance is better without considering *why* the performance is better, especially in simulation where we have absolute knowledge. In this domain, performance is measured by dialogue length and solution quality. However, dialogue length is the only component influencing the MPC since solution quality never affects the dialogue cost for a trained system. The primary cause of longer dialogues are unavailable filler inform and query (UFI–Q) speech acts by the user. This is because the system must inform the user of the available fillers and then the user must then inform the system of its second choice. These are easily identified and the mean number of UFI–Q speech acts for each dialogue are shown for all user conditions in Table 4.4. Notice that these numbers are inversely related to performance: the more UFI–Q speech acts, the worse the performance. For examples, in the combined condition the ID user's perform

0.38 UFI–Q speech acts per dialogue (u/d) compared to the 0.94 UFI–Q u/d for KR users. A lesser cause of longer dialogues is the system keeping the turn to inform the user of slot

Table 4.4: Mean number of UFI–Q speech acts between policies

Model	Novice	Int.	Expert	Combined
KR	0.0	1.15	0.53	0.94
ID	0.1	0.33	0.39	0.38

fillers when the user already has the available filler in the top position. In the combined condition, the KR system produces 0.06 unnecessary utterances per dialogue, whereas the ID system produces 0.045 of them per dialogue.

Exploiting User bidding differences: It follows that improved performance of the Importance-Driven approach stems from its negotiative turn transitions. These transitions are distinctly different than KR transitions in that there is information inherent in the users bids. A user that has a stronger belief strength is more likely to be have a higher bid and inform an available filler. Policy analysis shows that the ID system takes advantage of this information by using moderate bids —neither *highest* nor *lowest* bids— to filter user's based on their turn behavior. The distribution of bids used over the ten testing sessions is shown in Table 4.5. The initial position refers to the first bid of the dialogue; final position, the last bid of the dialogue; and medial position, all other bids. Notice that the system either uses the *low* or *mid* bids as its initial policy. Also, observe that 67.2% of dialogue medial bids are moderate.

Table 4.5: Bid percentages over all policies in the Combined User condition

Position	H-est	High	Mid	Low	L-est
Initial	0.0	0.0	70.0	30.0	0.0
Medial	20.5	19.4	24.5	23.3	12.3
Final	49.5	41.0	9.5	0.0	0.0

In some cases the KR system releases the turn when it should have kept it it. While a KR user will release the turn if its planned utterance has a weak belief, it may select that weak utterance when first getting the turn (either after a system utterance or at the start of the dialogue). This may lead to a UFI–Q utterance. The ID system, however, will

Table 4.6: Sample ID dialogue in Combined User condition; Cost=6

Sys	Usr	Spkr	Utt
1 low	mid	U:	inform burger b1
2 h-est	low	S:	inform burger have b3
3 mid	low	S:	inform side have s1
4 mid	h-est	U:	inform burger b3
5 mid	high	U:	inform drink d1
6 l-est	h-est	U:	inform side s1
7 high	mid	S:	bye

Table 4.7: Sample KR dialogue in Combined User condition; Cost=7

Spkr	Utt	Turn-Action
1 U:	inform burger b1	R
2 S:	inform burger have b3	R
3 U:	inform side s1	K
4 U:	inform drink d1	K
5 U:	inform burger b3	R
6 S:	inform side have s2	K
7 U:	inform side s2	K
8 S:	bye	

outbid the same user, resulting in a shorter dialogue. This situation is shown in Table 4.6 and Table 4.7. The dialogue is the same until utterance 3, where the ID system wins the turn with a *mid* bid over the user's *low* bid. In the KR environment however, the user gets the turn and performs an unavailable filler inform which the system must react too. The user has the same belief in both scenarios, but the negotiative nature of ID enables a shorter dialogues. In short, the ID system can win the turn when it should, but the KR system can not.

In other cases, though much fewer, the KR system keeps the turn when it should have released it. This situation presents itself when the user releases the turn and the system keeps the turn for more than one utterance, informing the available fillers for two slots. However, the user already had a strong belief and available top filler for one of those slots, and the system has increased the dialogue length needlessly. In the KR environment this is not unreasonable since the system knows that one of the user's beliefs was probably weak, otherwise the user would not have released the turn. The ID system can handle this by using a *high* bid to filter the user beliefs. In other words, the ID user can win the turn when it should have it, but the KR user can not.

4.4 Conclusion

These results definitively show that the ID model has the best overall performance because of it enables conversants to judge the importance of potential contributions. The KR model is not able to negotiate the turn assignment —its conversants cannot judge potential contributions— and so performance degrading utterances are far more common.

A discussion of initiative naturally arises from the importance-driven approach. As we mentioned before, a mixed-initiative system attempts to produce the ideal contribution at the ideal time. The data has shown that the importance-driven system does this far more effectively than the KR system, since the ID users produce far fewer extraneous speech acts. So, it seems that the importance-driven approach enables the system to not just reason over the transition of speaking roles, but reason over its initiative level as well. In the following chapter, we consider how turn-taking behavior is directly related to the initiative of the task as well as how the importance driven approach copes with more linguistically challenging domain.

Chapter 5

Exploring Importance-Driven Turn-Taking in Simulation

5.1 Introduction

The goal of this chapter it to explore the Importance-driven approach in a richer domain. This new domain models a number of discourse phenomena taken from the literature, expands the speech act set considerably, and increases the number of slots and values. We have three main questions. First, how does limiting the variability of turn-bids affect performance? Second, how does allowing the User to retract once preferred values change both the turn-taking and utterance policy? And third, how does turn-taking behavior relate to initiative behavior? Specifically, are there different turn-taking behaviors present when interaction initiative is more fixed rather than more mixed?

5.2 Negotiative Slot Filling

In this section, we introduce the domain used to evaluate different turn-taking strategies and the structure of the dialogue agents. In this domain we seek to model both the variability of user behavior, and psycholinguistic phenomena described in the literature. Even though this domain is not grounded by modeling real-world data, we feel that it is sufficiently sophisticated for comparing turn-taking frameworks.

We simulate a Negotiative Slot Filling (NSF) dialogue between two agents, the System and the User. The System takes the User's food order of a drink, burger, and side. The System knows which values are legal fillers for each of the three slots (sub-tasks) and the

User, with no initial knowledge of availability, has a number of desired items (sub-goals). Both agents must share information in order to fill all three slots with legal and preferred values, and so the dialogue is quite mixed-initiative. This domain, while based on the one used in Chapter 4, uses a much wider array of dialogue acts and has a much richer interaction between the System and the User.

Unlike Chapter 4, the User has a number of values sorted by preference for each subtask and operates on a goal stack, where each element is a combination of slot items, which can be dependent on each other. Dependence means that if one item in the combination is unavailable, the User no longer wants the other items in the combination. We address this because dependent sub-goals have long been the study of search techniques in Artificial Intelligence [49], but have received minimal attention in terms of user goals for a spoken dialogue system (cf. [44]). Items can appear on more than one combination and these combinations are randomized.

The NSF domain has more speech acts than the domain in Chapter 4. Table 5.1 gives the complete speech act list and their glosses. The System can give information about legal values (*inform*) and the User can *assert* which values it prefers. Both can *propose*, *reject* and *accept* values. In most situations, the System will choose between *inform*, purely information giving, and *propose*, which obligates the User to respond with a counter proposal, *reject*, or *accept*. When these are exhausted, the System will *query* to obligate the User to respond to a specific slot. The System can *reject* User *proposals* and *asserts* but will only explicitly *accept* the former. Both *inform* and *propose* speech acts have variants, *inform final* and *propose final*, that tell the user there is nothing else available resulting in the User preferences being constrained to what is available.

Three psycholinguistic phenomena are modeled in this simulation: implicit acceptance, implicit rejection, and retraction. We model implicit acceptance and rejection according to the rules in the list below. Rules 1 and 4 are based on the "Collaborative Principle" [74]: "Conversants must provide evidence of a detected discrepancy in belief as soon as possible [p.4]". Rules 2 and 3 model the counter-proposal approach to implicit rejection [74, 63].

Table 5.1: Agent Dialogue Moves and Glosses

Speech Act	Gloss
System	
inform	We have cola.
inform final	We have cola and thats it
propose	How about a cola?
propose final	How about a beer? Thats all we have.
accept	A sprite it is.
reject	We don't have grape juice.
User	
assert	I want a grape juice.
propose	I want a grape juice if you have it.
reject	No. I don't want water.
accept	Sure, a cola's great.
retract	I don't want that cola anymore.

1. The User considers any *assertion* to be accepted until another rule takes effect

2. If the *first* System speech act following a User's *assert* is *inform* on the same slot than it is considered rejected by both.

3. If the System *proposes* at any time following the User's *assert* or *propose*, it is considered rejected by both.

4. If the User *proposes* or *asserts* at any time following the System's *proposal*, it is considered rejected by both.

Among the collection of negotiation dialogue moves presented by Sidner [63], Retract-Proposal is of particular interest. This move enables an agent to change its mind about an item that was proposed. In NSF, we expand this so that the User can retract both proposed and accepted values, not just the former as in Sidner. *Retraction* occurs when preferences between slots are dependent and one item has been accepted (or proposed) before another item was rejected. Since the items are dependent, the User *retracts* the first item. In other words, User preferences change based on what is available.

5.2.1 Simulated System Design

The system simulation used here is similar to that used in Chapter 4. The most similar is the turn-taking behavior, the only difference being the addition of a *nearSilence* bid for the importance-driven approach. Keep-Or-Release remains the same.

The system design separates TURN, ATTENTION, and SPEECHACT actions. It uses TURN rules to determines the turn-taking signal, ATTENTION rules to decide what sub-task to focus on (i.e. a specific slot), and SPEECH ACT rules to decide what to say (e.q. "What kind of [attention] do you want?"). This structure enables us to hand-craft attention or speech act behavior individually, while optimizing the other.

We use one MDP for all three decisions and most the state variables are shared between the states for each particular action. The RL state consists of seven variables for each of the three sub-tasks, as well as binary variables indicating what type of the decision the state pertains too. The variables for one sub-task are shown in Table 5.2. In this set up, the ATTENTION decision is made first and the SPEECHACT and TURN states both have knowledge of the decision. However, neither of these states have knowledge of each other. Similar to Chapter 4, this design does not reinforce unspoken SPEECHACT decision, but it does reinforce ATTENTION decisions. This is because the TURN decision has knowledge of the sub-task selected.

However, one can theoretically tailor an MDP for each action type. This would enable state size reduction by maintaining the MDP while tailoring the state for each action. This delves into multi-agent RL, which is applied in Chaper 10.

In addition to the three action rule sets, the simulated system has an UNDERSTANDING rule set that updates the IS based on the environment, and a DELIBERATION rule set that updates the IS based on other IS variables.

5.2.2 Hand-crafted User Simulation

The goal here is to create a simulated user that will mimic the behavioral variability found in real-life and behave in commonsense fashion when it comes to turn-taking and slot-filling. The User's behavior is guided by its goal, agenda, and beliefs. Each User

Table 5.2: The Reinforcement Learning State is made of the same set of variables for each sub-task (slot) combined into one large state. All variables are sub-task specific. Here we show the RL state variables for one sub-task.

Variable	Type	Details
Attention	binary	Does the system currently have this sub-task in focus ?
QUD	binary	The System has queried the User. User has not yet responded.
Sys Proposed	tri-ary	N. of System proposals. 0, 1, or > 1
User ProposedNA	tri-ary	N. of User Not Available (NA) proposals. 0, 1, or >1
User ProposedAv	tri-ary	N. of User Available proposals. 0, 1, or >1
User Just ProposedNA	binary	In the last User turn, did they propose a NA value?
User AssertedNA	tri-ary	N. of User Not Available (NA) assertions. 0, 1, or >1
User Just AssertedNA	binary	In the last User turn, did they assert a NA value?
Avail Info	tri-ary	N. of available values *not* yet told to user. 0, 1, or >1
PLAN	binary	Has the system accepted a value

begins the dialogue with an arbitrary length Goal Stack where each element has a value for each slot (3 values for each element). The top of the Goal Stack is popped, and the User works to fill slots using that goal. If the System *proposes* a value not in that goal than the User is obligated to *reject* it. The types of turn-taking behavior of the user is identical to that described in Chapter 4, with the addition of a *silence* bid.

To produce a sophisticated simulated User population that will better mimic human users we use two different types of users: Novice and Expert. The main difference from Chapter 4 is that these Users have beliefs on *items* whereas the previous Users had beliefs on *slots*. Novice and Expert user's differ substantially over three different beliefs of item availability: Strong, Warranted, and Weak [9]. This results in different behavior of the User. For example, when working on a item with a Weak belief the User will *propose* instead of *assert*, and the converse is true for a Strong belief. A Warranted belief results in random action selection. Beliefs also change during the dialogue. If the System *proposes* or *givesInfo* on a value than the User's availability belief of that value becomes Strong. Beliefs are initialized so that the Expert is more likely to have a Strong belief of an available value, and a Weak belief of an unavailable one. Novice's are more likely to have Weak beliefs without regard to availability. The probabilities governing the values and beliefs are shown in Table 5.3 in Table 5.3. Only one combination of desired values is achievable, and Experts will have this achievable combinations within the first half of all its stack of possible combinations. The achievable combination is randomly placed in the Goal Stack

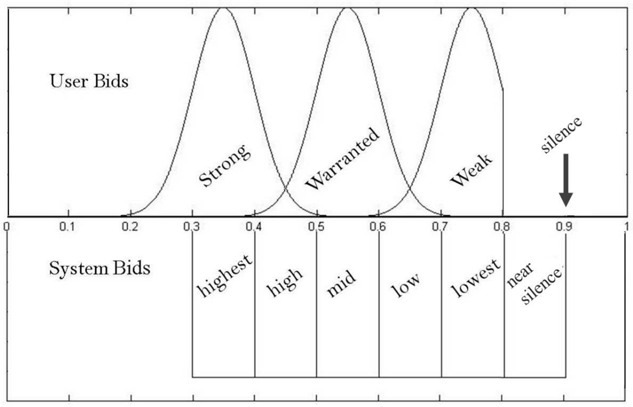


Figure 5.1: Probability Distributions of Turn-Bids

for the Novice.

When interacting with the importance-driven system, the User bids according to its confidence of the particular item: the lower the confidence, the slower the onset. The belief centers a Gaussian distribution (σ^2=0.05), and the resulting value becomes the onset. Figure 5.1 shows the onset distributions for both the System and the User. The User utterances *reject*, *accept*, and *retract* are treated as having high confidence for turn-taking purposes.

Table 5.3: User Belief Probabilities

	Value Available		Value Not Available	
Belief	Expert	Novice	Expert	Novice
Strong	0.7	0.1	0.1	0.1
Warranted	0.2	0.2	0.2	0.2
Weak	0.1	0.7	0.7	0.7

Sample Dialogue

A sample dialogue is shown in Table 5.4. The rules that govern implicit acceptance and rejection are noted in the final column. An Expert User *asserts* two values. This user is then forced to *retract* (line 4) the implicitly accepted drink after the System implicitly rejects the burger by a counter-proposal (line 3). The System then proposes a sprite (line 5). The User then rejects the System's proposal (since it is not the top value the User

Table 5.4: NSF dialogue fragment with Expert User

Line	Speaker	speech act	Gloss	Rule
1	User:	assert drink d1	*I want a coke*	
2	User:	assert burger b2	*I want a cheese burger*	
3	System:	inform burger b1	*We have a bacon burger*	1,2
4	User:	retract drink d1	*I don't want a coke anymore*	
5	System:	propose drink d2	*What about a sprite?*	
6	User:	propose drink d3	*How about milk?*	4
...				

wants), and proposes its most preferred one (line 6).

5.3 Importance-Driven Turn-taking in the Negotiative Slot Filling domain

We now present two sets of experiments that validate and explore the Importance-Driven framework.

We use four different NSF setups using two environments and two modes. One environment, 2-Av, has two available and three unavailable items for each of the three slots. Another environment, 3-Av, has three available and two unavailable items for each of the three slots. Both of these environments are used in two modes: item independence or item dependence. For these experiments, we fix the sub-task (attention) of the User and the System to be a strict order. However, when the agent believes the sub-task to be completed (or has nothing left to say) it does move onto the next one.

We expand on the findings of Chapter 4 in two ways. We first examine the granularity of the bids, comparing the performance of the full bid spectrum with two other bid structures: one that only uses the extremes (highest, near silence), and one that uses the extremes and a middle bid (highest, near silences, low). We then compare the importance-driven and KR approaches using the four setups described above. Since the NSF domain is far more richer and psycholinguistically grounded than the domain used in Chapter 4, we feel like is far better evaluation. However, while not shown here, we did recreate the results found in Chapter 4 using this domain by reducing the number of slots and items.

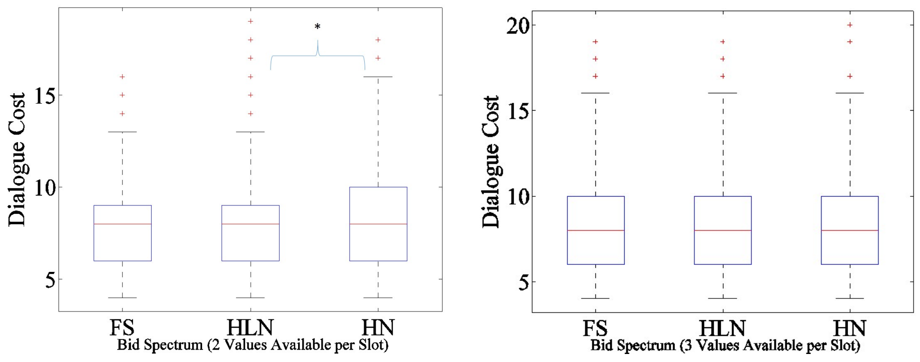


Figure 5.2: Comparing variation in the bid spectrum using Independent values. There is a significant ($p < 0.00001$) difference between HLN and HN in the 2-Av case but no where else.

For all experiments, policies are trained on 1 million simulated interactions and then evaluated using 100,000 simulated interactions. The Monte Carlo learning algorithm is used to minimize dialogue length, where each speech act has a cost of 1 and the system will not exit until the solution is found. The algorithm uses $\epsilon - greedy$ search, with ϵ decay at 0.99997 and ϵ initialization at 0.5. Policies are evaluated by comparing dialogue cost, and Wilcoxon Rank Sum tests are used for significance. Box and whisker plots are used as the dialogue costs of the test batch are substantially skewed.

5.3.1 The Bid Spectrum

We begin our discussion by investigating the bid spectrum. That is, does a system really need five or six bids to reach the same performance? Using the 4 environments described previously, we compared the three different system bidding schemes: FS, which has all six bids; HN, which only has Highest and the Near Silence bids, and HLN, which has Highest, Near Silence, and Low bids.

The results for Independent environments are shown in Figure 5.2, with 2-Av on the left and 3-Av on the right. We find that there is a significant ($p < 0.00001$) difference between the HLN and HN scheme in the 2-Av environment but no difference between schemes in 3-Av.

Since the likelihood that the User is wrong is greater in 2-Av, the System should be more cautious letting the User get the turn in 2-Av. This results in the System using a bid to filter the user, as described in Chapter 4. This can be confirmed by looking at number of User proposals and asserts. A relative increase in asserts indicates that System is giving more information to the User, and so increasing the confidence level of values. We find that in the HLN 2-Av case there is 71% asserts and 21% proposes (over all speech acts) but there is only 41 % asserts and 58% proposes in the 3-Av case. When we look at the HN 2-Av case we find the same pattern as seen in the 3-Av case with 42% asserts and 48% proposes. Since the 3-Av System can expect that the User is more likely to give an available value, and there is minimal impact from the User being wrong, it does not require user filtering for good performance.

In the Dependent environment, the 3-Av results are consistent with the those found in Independent environments but the 2-Av results do show a modest difference. In the 2-Av case, there is a small but significant performance gain for the FS scheme over the HLN scheme ($p < 0.001$, mean values are 8.8 vs. 8.89). The primary difference is best shown with comparing the beginning of two complementary dialogues, shown in Table 5.5. In the HLN fragment, the System uses a Low bid to filter the User, and then uses its Highest bid for the final inform. This choice, while necessarily optimal, results in the User attempting to get the turn with a highest bid but not getting it. In the FS fragment the System has much more flexibility with its bids and, choosing high for its second contribution, it leaves room for the User to out-bid it.

Table 5.5: Two IDTB dialogue fragments in the Dependent 2-Av NSF environment

HLN Bid Spectrum				
Line	System Bid	User Bid	Speaker	speech act
1	Low	Lowest	System	inform side s1
2	Highest	Highest	System	inform side all s2
3	Low	Highest	User	assert side s1
Full Spectrum				
1	Mid	Lowest	System	inform side s1
2	High	Highest	User	assert side s1

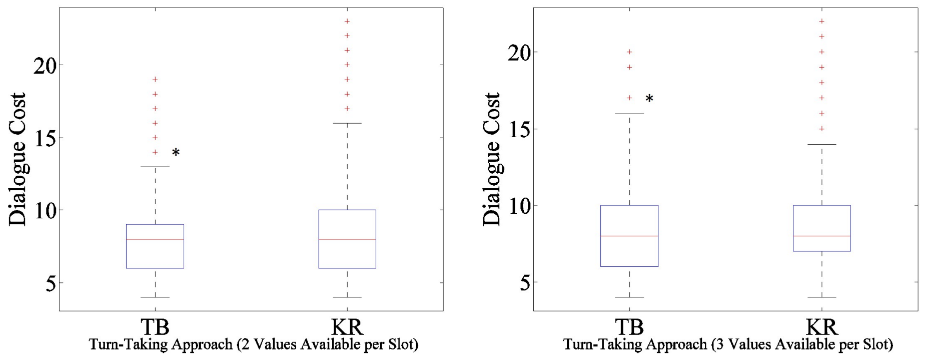


Figure 5.3: Comparing Importance-Driven and KR in Independent environments. The Importance-Driven system has lower dialogue cost than KR in both environments ($p < 0.00001$)

5.3.2 Fuller Evaluation of Importance-Driven Turn-Bidding

We now present a fuller comparison of the Importance-driven and KR approaches using the four NSF setups. We will sometimes use "TB" (turn-bidding) to indicate the importance-driven system or user.

The Independent Case

We first evaluated the ID framework on the independent 2-Av and 3-Av environments. Figure 5.3 shows the test batch results and we find that in both situations the ID framework performs significantly better than the KR approach ($p < 0.00001$).

In the 2-Av case (Figure 5.3, left), this is partly due to the ability of the system to filter users based on bids. However, there is more to the gain as the HN framework still performs better than the KR framework. We found that nearly all of the gain can be erased when accounting for "User Forced Release" (respective means of this condition and KR, 8.02 vs. 8.08). User Force Release refers to the KR User releasing the turn when it is *about* to speak on a value of low confidence. This does not necessarily happen in the Importance-driven framework, and so available values that have low confidence can be given without additional System speech acts. If we force the TB User to release the turn (i.e. say nothing) in the same manner as the KR user, most of the TB gains are erased. The normal TB user releases the turn interactively, allowing the System to out bid the

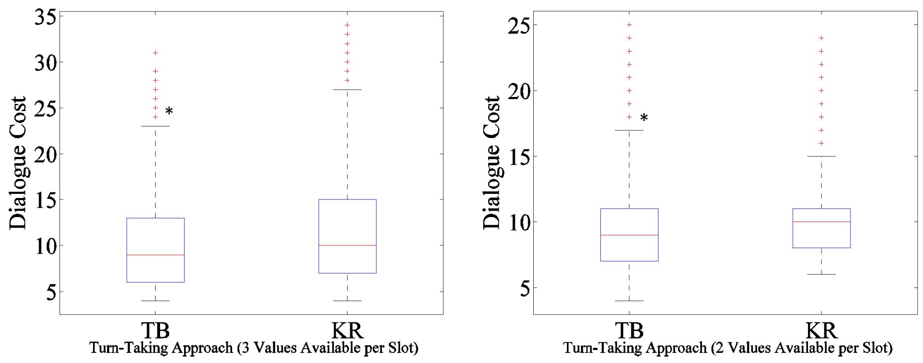


Figure 5.4: Comparing TB and KR in Dependent environments. TB has lower dialogue cost than KR in both environments ($p < 0.00001$)

User if they are about to propose an unavailable slot. However, even with the forced release there still is a significant difference. We found that the difference comes from the User bidding highest while the System attempts to keep the turn with its own highest bid. The can result in the User (randomly) getting the turn and doing something to advance the dialogue (e.g. accepting). In the KR framework, this would not be possible. If we eliminate the ability of the User to make a highest bid, this minor difference disappears.

The gains are simpler to explain in the 3-Av case (Figure 5.3, right), as ID performance meets that of KR when we account for User Forced Release ($p < 0.2$). This makes sense since the User is more often right than wrong, and so there is utility for the System to be more passive. The KR system does not have this choice, and so must speak when the User releases the turn (sometimes prematurely). This is supported by dialogue statistics that show the TB user proposes 2% more than the KR user, and the KR user asserts 2% more than the TB user. This suggests that the KR system is giving more information than the TB system, and indeed this is the case as the KR system informs nearly 11% more than the TB system while both rarely use proposals. However, this sometimes results in *over* informing, as the the KR system informs the availability of the top value on the value stack 3% more than the TB system does.

The Dependent Case

The results for the comparison in Dependent environments are consistent with the Independent case, and we report that in both situations the Importance-Driven approach performs significantly better than KR ($p < 0.00001$), shown in Figure 5.4.

The primary difference between independent and dependent sub-goals is that, in the dependent case, there is utility to rejecting and informing as quickly as possible. When we looked for behavior accounting for the performance differences we found similar explanations of both the 2-Av and the 3-Av case when we just looked at the HN bidding scheme (and so controlling for the user filtering in the 2-Av case).

We found that when we removed the ability of the System to reject using the highest bid, the bidding behavior changed to compensate for this. Table 5.6 shows the beginnings of an original ID dialogue and one using the constrained System. The constrained ID System performed worse than the unconstrained one, but did not completely lose all performance gains, only losing about 50% of the gain over KR. The bidding behavior changed considerably, with the System holding the turn for far longer than before in both environments resulting in a Maintain Turn increase of 10% for both 2-Av and 3-Av. Where the original systems had used the highest bid about 30% of the time, it now used it more than 40% of the time, and was doing so to hold the turn to give information. Though the KR system could also execute this behavior, it could not leave the option for a confident user to get the turn (albeit randomly) and so it learns to release the turn far more often. If we eliminate the User's use of the highest bid, performance reaches that of the KR system.

5.4 Comparing turn-taking and different initiative strategies

Using only importance-driven turn-taking, we compare different initiative strategies over three different NSF setups. We discuss the performance and turn-taking differences. The pertinent background is found in Section 2.2, which describes different initiative levels in human–machine dialogue. On one end of the spectrum are fixed strategies where either the

Table 5.6: Two ID dialogue fragments in the Dependent 3-Av NSF environment. Constrained system can not use highest bid to reject a value.

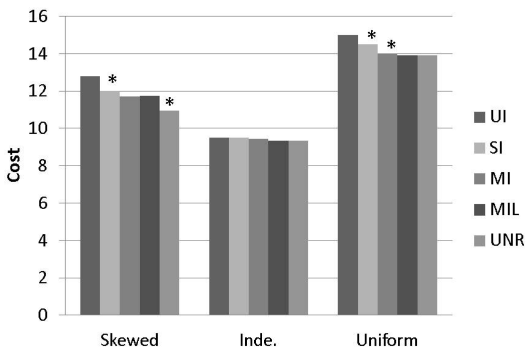
Original				
Line	Speaker	speech act	System Bid	User Bid
1	User	propose side s5	Near Silence	Lowest
2	System	reject side s5	Highest	High
3	User	assert side s4	Low	Highest
4	System	reject side s5	Highest	High
Constrained				
1	System	inform side s2	Highest	Low
2	System	inform side s3	Highest	Low
3	System	inform side s1	Highest	Low

system or the user directs the interaction. On the other end is mixed-initiative strategies, where the conversational roles are not fixed.

5.4.1 Experimental Design

We use three different versions of the simulation domain: Uniform-Dependent, Skewed-Dependent, and Skewed-Independent. All three slots have six potential items and the User has ten combinations of sub-goals. The first version, Uniform-Dependent, maintains the dependencies within sub-goal combinations and all slots have three available items and three unavailable items. In the Skewed-Dependent version, two of the three slots only have two items available, leaving four unavailable. The third slot has four items available and two items unavailable. This is meant to model situations where sub-tasks have differing complexities. In the Skewed-Independent version, skewed item availability remains but the dependencies between them are lifted. This makes the task much easier since the User is no longer constrained to the current combination.

We implemented five different attention strategies. The first three are hand-crafted. *User-Initiative*, where the User attempts to fill the slots based on the current item combination and the System will *not* try to initiate any new sub-task. *System-Initiative*, where the System attempts to fill the slots in the best order possible no matter what the User says. The best order is filling the slots with less items available first. *Mixed-Initiative*, where the System attempts to fill the slots in the best order but will change its sub-task



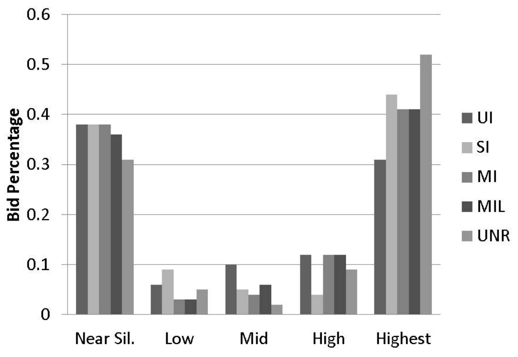


Figure 5.5: Mean dialogue cost and bid percentages for test batch. (*) indicates $p < 0.001$.

to match that of the User and will not deviate until that sub-task is finished (slot is filled) or the User changes sub-tasks. The last two use Reinforcement Learning. *Mixed-Initiative, Learning*, which is the same as the hand-crafted *Mixed-Initiative* above but the System learns the order; and *Unrestricted* (UNR), where the System can speak on any uncompleted sub-task at any point in the dialogue.

For all three conditions and all five attention management strategies we trained a System on one million dialogues. We then ran every System on 50,000 test dialogues. We evaluated different attention management strategies by comparing the number of speech acts it takes for all three slots (drink, burger, and side) to be filled. This number is the "cost" of the dialogue and is the reinforcement signal used to train turn-taking, speech act, and in some cases attention, behavior. Since the System will never exit until the slots have been filled appropriately this is the only reinforcement used: a "better" dialogue is a shorter one.

5.4.2 Results

Figure 5.5 (left) shows the mean dialogue cost of test dialogues for each attention management strategy and environmental condition. Here, since the data is roughly Gaussian, we use a t-test. For the Skewed-Independent condition (middle), we see virtually no difference between attention strategies. However, for the Uniform-Dependent condition (right), there is a significant reduction ($p < 0.001$) in dialogue cost between the System Initiative and User Initiative strategies. There is also a significant reduction between the Mixed-Initiative approaches (MI, MIL, and UNR) and the System Initiative one ($p < 0.001$). This pattern repeats itself for the Skewed-Dependent condition (left) with one exception. The UNR attention approach performs significantly better than the MI and MIL approaches with a cost reduction of nearly 8% ($p < 0.001$).

Figure 5.5 (right) shows the bid percentages of the test dialogues. Recall that these were also learned, though there were some restrictions in the User-Initiative strategy [1]. The UNR System has the highest percentage of *highest* bids, followed by the System-Initiative system. The User-Initiative System used far less *highest* bids than any other System. However, this is inverted with respect to the *high* bids. The three Systems that allowed the User more sub-task control (UI,MI,MIL) had a far larger percentage than the other two (SI,UNR). We also found that with respect to using the *Near Silence* bid, the UNR system uses far less than any other system.

5.4.3 Discussion

Taken as a whole, we found that as the conditions became more difficult, an attention strategy that gives the System more flexibility becomes more critical for better performance. In the Independent condition, there were no differences between attention strategies. However, these differences emerged as dependency was introduced (Uniform-Dependent) and then combined with uneven slot item availability (Skewed-Dependent). These emerging differences suggests that in the standard, simple dialogues most systems are currently used

[1]The System was forced to bid *Near Silence* if the slot was filled but the User had not offered a new sub-task yet. In this sense, the System is stalling

Table 5.7: Example UNR dialogue fragment with glosses. The bids pertain to the speech act on their line

Sys Bid	Usr Bid	Spkr	Gloss
Highest	Mid	Sys	We have milk
Highest	High	Sys	How about a water?
Highest	High	Sys	We have a regular burger
Highest	Highest	Sys	How about a veggie burger?
Near Sil.	Highest	User	I want a water
Near Sil.	Highest	User	I want a regular burger

for, sophisticated attention management is probably unnecessary. However, as spoken dialogue systems are used in more complex and difficult applications, effective attention management will be beneficial to the efficiency of the interaction.

The UNR approach had significantly shorter dialogues than any other approach in the Skewed-Independent condition. This condition, with uneven item availability and slot item dependence, is the most interesting. This System learned to give information on different slots without waiting for one slot to be filled, moving from the first low availability slot to the other a quickly as possible (Table 5.7). This behavior reduces retractions since the User preferences are more constrained. It is relatively intuitive: most previous systems require the sub-task to be complete before moving to another one. Since this behavior hinges on it winning the turn, it is unsurprising that this System learned to use *highest* bids more often than any other approach. We also found that there was little difference between the MI and MIL strategies. This is not unexpected since the MI system used a sub-task order that we believed to be optimal. The fact that they performed comparably is a nice sanity check in terms of attention learning.

When looking at the turn-taking differences we found that in general, our results suggest that the more the System can control what sub-task its speech act pertains to (the UNR and the System-Initiative approaches), the more *highest* bids it will use. This is because these two systems are trying to give information to the User as quickly as possible. However, while the UNR system was the most pro-active in terms of trying to win the turn, the more User-based strategies used more *high* bids. This suggests that there were a number of times when these systems "wanted" the turn but not enough to risk a confident

User *not* getting it. The turn-taking differences found between the attention strategies also underscore the importance of turn-taking for other dialogue behavior: if the System were unable to compete for the turn, then the UNR system would have been unable to learn the best attention management policy.

Chapter 6

Improving Incremental Speech Recognition for Spoken Dialogue Systems

6.1 Introduction

We introduced Importance-Driven Turn-Taking in Chapter 3, evaluated the approach in Chapter 4, and explored it further in Chapter 5. This chapter addresses one of the major challenges to importance-driven turn-taking in the real-world, that of incremental speech recognition. Importance-driven turn-taking is dependent on dialogue context, and Incremental Speech Recognition (ISR) provides this real-time situational knowledge that a fully importance-driven system requires. Section 2.4 provides a background on ISR.

Here, we address three challenges to using ISR in dialogue systems. First, we propose a new method of recognizing partial phrase results that increases their stability. Second, we illustrate the use of logistic regression to predict partial stability and accuracy. And third, we introduce a method to combine incremental results with a complete recognition-based dialogue system.

6.2 Lattice-Aware Incremental Speech Recognition

Incremental Speech Recognition (ISR) enables a spoken dialogue system (SDS) to react quicker than when using conventional speech recognition approaches. Where conventional methods only return a result after some indication of user completion (for example, a short period of silence), ISR returns partial phrase results while the user is still speaking. Having access to a real-time stream of user speech enables more natural behavior by a SDS, and is a foundation for creating systems which take a more active role in conversations such as Importance-Driven Turn-Taking.

Research by Fink et al.[18] and Skantze & Schlangen [64], among others, has demonstrated the efficacy of ISR but has also drawn attention to a significant obstacle to widespread use: partial phrase results are generally unstable and so, as more speech is decoded, are prone to revision. For example, the ISR component in a bus information SDS may return the partial "leaving from Hills", where "Hills" is a neighborhood name. It may then return the revision "leaving from Pittsburgh", which the system must handle gracefully. Recent approaches sought to minimize revisions by delaying the partial by some time. While this is effective, it works directly against the primary benefit of ISR: the short lag between speech and recognition. Here, we attempt to increase stability and accuracy without applying a uniform delay.

We first characterize three approaches to ISR that make different trade-offs between stability and the number of partials generated. We then present a novel hybrid approach — Lattice-Aware ISR — that combines their strengths to increase stability without adding latency. This section is taken from Selfridge, Arizmendi, Heeman, and Williams [58].

6.2.1 Three approaches to ISR

We now describe three approaches to ISR: Basic, Terminal, and Immortal. While we may be first to explicitly describe and compare these three methods we take no credit for their invention. Basic ISR simply returns the most likely word sequence observed after some number of speech frames has been decoded (in our case every 3 frames/30ms). This is the least restrictive approach, and we believe is the method used in recent ISR research.

Terminal ISR, a more restrictive approach, finds a partial result if the most likely path through the (partially-decoded) lattice ends at a *terminal* node in the language model. While we do not claim novelty here, we should note that this approach has never been explicitly described in the literature. The intuition is that if a partial result finishes a complete phrase expected by the language model, it is more likely to be stable. The meaning of *terminal* is slightly different for rule-based language models (RLMs) and statistical language models (SLMs). For a rule-based grammar, the terminal node is simply one that can end a valid phrase ('Pittsburgh' in 'leaving from Pittsburgh'). For an SLM, a terminal node indicates that the most likely successor state is the special end-of-sentence symbol. In other words, in an SLM Terminal partial result, the language model assigns the highest probability to ending the phrase.

A third method, Immortal ISR, is the most restrictive method [66]. If all paths of the lattice come together into a node — called an *immortal* node — then the lattice structure before that node will be unchanged by any subsequent decoding. This structure guarantees that the best word sequence prior to an immortal node is stable. Immortal ISR operates identically for both RLMs and SLMs. The LM size and choice of search beam size affects both accuracy and the number of immortal nodes produced: a smaller beams yields a sparser lattice with more immortal nodes and lower accuracy; a larger beam yields a richer lattice with fewer immortal nodes and higher accuracy. In this work we used the recognizer's default beam size, which allows recognition to run in less than real time and yields near-asymptotic accuracy for all experiments. We also restrict our attention to immortal nodes that do not occur *within* a word.

6.2.2 Comparing ISR Methods: Setup

To compare these approaches we evaluate their performance. Utterances were extracted from real calls to the Carnegie Mellon "Lets Go!" bus information system for Pittsburgh, USA [47, 43]. We chose this domain because this corpus is publicly available, and this domain has recently been used as a test bed for dialogue systems [6]. The AT&T WATSON speech recognition engine was used [23], modified to output each of the three types of partials. We then tested each approach on three different recognition tasks. The first two tasks used rule-based language models (RLM), and the third used a statistical language model (SLM).

The two rule-based language models were developed for AT&T "Let's Go" dialogue system, prior to its deployment [78]. The first RLM (RLM1) consisted of street and neighborhood names, built from the bus timetable database. The second RLM (RLM2) consisted of just neighborhood names. Utterances to test RLM1 and RLM2 were selected from the corpus provided by Carnegie Mellon to match the expected distribution of speech

Table 6.1: Statistics for Recognition Tasks. In all tables, *All* refers to all utterances in a test set, and *MW* refers to the subset of multi-word utterances in a test set.

	RLM1	RLM2	SLM
Num. Utts All	7722	5411	42620
Num. Utts MW	3213	1748	20396
Words/Utt All	1.7	1.5	2.3
Words/Utt MW	2.8	2.6	3.8
Utt. Acc. All.	50 %	60 %	62 %
Utt. Acc. MW	53 %	56 %	44 %

at the dialogue states where RLM1 and RLM2 would be used. RLM1 was evaluated on a set of 7722 utterances, and RLM2 on 5411 utterances. To simulate realistic use, both RLM test sets were built so that 80% of utterances are in-grammar, and 20% are out-of-grammar. The SLM was a 3-gram trained on a set of 140K in-domain utterances, and is tested on a set of 42620 utterances. In past work, Raux et al. (2005) report word error rates (WERs) of 60-68% on data from the same dialogue system, though on a different set of utterances. By comparison, our SLM yields a WER of 35%, which gives us some confidence that our overall recognition accuracy is competitive, and that our results are relevant.

Table 6.1 provides a few statistics of the LMs and test sets, including *whole-utterance accuracy*, computed using an exact string match. Results are analyzed in two groups: *All*, where all of the utterances are analyzed, and *Multi-Word (MW)*, where only utterances whose transcribed speech (what was actually said) has more than one word. Intuitively, the MW utterances are where ISR would be most effective. That said, ISR is beneficial for both short and long utterances — for example, ISR systems can react faster to users regardless of utterance length.

ISR was run using each of the three approaches (Basic, Terminal, Immortal) in each of the three configurations (RLM1, RLM2, SLM). The mean number of partials per utterance is shown in Table 6.2. For all ISR methods, the more flexible SLM produces more partials than the RLMs. Also as expected, multi-word utterances produce substantially more partials per utterance than when looking at the entire utterance set. The Basic approach produces nearly double the number of partials than Terminal ISR does, and Immortal ISR production highlights its primary weakness: in many utterances, no immortal nodes are found. Given this however, immortal node occurrence is directly related to the number of words in the utterance, as indicted by the greater number of immortal partials in multi-word utterances.

Stability is assessed by comparing the partial to the final recognition result. For simplicity, we restrict our analysis to the 1-Best hypothesis. If the *partial* 1-Best hypothesis is a prefix (or full exact match) of the *final* 1-Best hypothesis then it is considered stable. For instance, if the partial 1-Best hypothesis is "leaving from Forbes" then it would be stable if the final 1-Best is "leaving from Forbes" or "leaving from Forbes and Murray" but not if it is "from Forbes and Murray" or "leaving". Accuracy is assessed similarly except that the transcribed reference is used instead of the final recognition result.

6.2.3 Comparing ISR Methods: Results

We report stability and accuracy in Table 6.3. Immortal partials are excluded from stability since they are guaranteed to be stable. The first four rows report stability, and the second six report accuracy. The results show that Terminal Partials are relatively unstable, with 23%-37% of partials being stable, and that their stability drops off when looking at multi-word utterances. SLM stability seems to be somewhat higher than that of the RLM. Basic partials are even more unstable (about 10% of partials are stable), with extremely low stability for the SLM. Unlike Terminal ISR, their stability grows when multi-word utterances are analyzed, though the maximum is still quite low.

The results also show that partials are always less accurate than they are stable,

Table 6.2: Average Number of Partials per utterance

ISR	Group	RLM1	RLM2	SLM
Basic	All	12.0	9.9	11.6
	MW	14.6	12.3	29.7
Terminal	All	5.4	3.3	6.2
	MW	6.4	4.1	8.8
Immortal	All	0.22	0.32	0.55
	MW	0.42	0.67	0.63

Table 6.3: Stability and Accuracy Percentages

ISR	Group	RLM1	RLM2	SLM
Stability				
Basic	All	10 %	11 %	7 %
	MW	14 %	15 %	9 %
Terminal	All	23 %	31 %	37 %
	MW	20 %	28 %	36 %
Accuracy				
Basic	All	9 %	1 %	5 %
	MW	11 %	13 %	6 %
Terminal	All	13 %	21 %	24 %
	MW	12 %	17 %	21 %
Immortal	All	91 %	93 %	55 %
	MW	90 %	90 %	56 %

indicating that not all stable partials are accurate. Immortal partials are rare, but when they are found, they are much more accurate than Terminal or Basic partials. The RLM accuracy is very high, and we suspect that immortal nodes are correlated with utterances that are easier to recognize. Terminal ISR is far more accurate than Basic ISR for all of the utterances, but its improvement declines for multi-word RLMs.

We have shown three types of ISR: Basic, Terminal and Immortal ISR. While Basic and Terminal ISR are both highly productive, Terminal ISR is far more stable and accurate than Basic. Furthermore, there are far more Basic partials than Terminal partials, implying that the dialogue manager would have to handle more unstable and inaccurate partials more often. Given this, Terminal ISR is a far better “productive ISR” than the Basic method. Taking production and stability together, there is a double dissociation between Terminal and Immortal ISR. Terminal partials are over produced and relatively unstable. Furthermore, they are even less stable when the transcribed reference is greater than one word. On the other hand, Immortal partials are stable and quite accurate, but too rare for use alone. By integrating the Immortal Partials with the Terminal ones, we may be able to increase the stability and accuracy overall.

Table 6.4: Lattice-Aware ISR (LAISR) Example

1-best	Partial Type
yew	Terminal
sarah	Terminal
baum	Terminal
dallas	Terminal
downtown	Terminal
downtown	Immortal
downtown pittsburgh	Terminal
downtown pittsburgh	Immortal

Table 6.5: Lattice-Aware ISR Stats

Partials per Utterance			
	RLM1	RLM2	SLM
All	5.6	3.5	6.7
MW	6.7	4.5	9.6
Stability Percentage			
All	24 %	33 %	40 %
MW	24 %	35 %	41 %
Accuracy Percentage			
All	15 %	23 %	26 %
MW	16 %	22 %	24 %

6.2.4 Lattice-Aware ISR (LAISR)

We introduce *Lattice-Aware ISR* (LAISR — pronounced “laser”), that integrates Terminal and Immortal ISR by allowing both types of partials to be found. The selection procedure works by first checking for an Immortal partial. If a new immortal node is not found then it looks for a new Terminal node. Redundant partials, as defined by having the same 1-best string, are only returned when the partial type changes. An example recognition is shown in Table 6.4. Notice how the first four partials are completely unstable. This is very common, and suppressing this noise is one of the primary benefits of using more right context. Basic ISR has even more of this type of noise.

We evaluate LAISR on the three recognition tasks described above (see Table 6.5). The first two rows show the average number of partials per utterance for each task and utterance group. Unsurprisingly, these numbers are quite similar to Terminal ISR. The stability percentage of LAISR is shown in the second two rows. For all the utterances, there appears to be a very slight improvement when compared to Terminal ISR in Table 6.3. The improvement increases for MW utterances, with LAISR improving over Terminal ISR by 4–7 percentage points. This is primarily because there is a higher occurrence of Immortal partials as the utterance gets longer. Accuracy is reported in the final two rows. Like the previous ISR methods described, the accuracy percentage is lower than the stability percentage. When compared to Terminal ISR, LAISR accuracy is slightly higher, which confirms the benefit of incorporating immortal partials with their relatively high accuracy.

To be useful in practice, it is important to examine *when* in the utterance ISR results are being produced. For example, if most of the partials are returned towards the end of utterances, than ISR has less value over standard turn-based recognition because the difference in recognition time is minimal. Figure 6.1 shows the percent of partials returned from the start of speech to the final partial for MW utterances using the SLM. This figure shows that partials are returned rather evenly over the duration of utterances. For example, in the first 10% of duration of each utterance, about 10% of all partial results are returned. Figure 6.1 also reports the stability and accuracy of the partials returned.

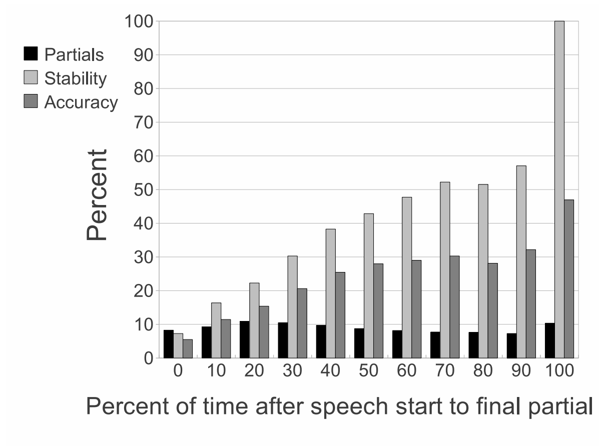


Figure 6.1: Percent of LAISR partials returned from the start of detected speech to the final partial using the SLM. The percentage of partials returned that are stable/accurate are also shown.

These numbers grow as decoding progresses, but shows that mid-utterance results do yield reasonable accuracy: partials returned in the middle of utterances (50%-60% duration) have an accuracy of near 30%, compared to final partials 47% percent.

For use in a real-time dialogue system, it is also important to assess *latency.* Here we define latency as the difference in (real-world) time between when the recognizer receives the last frame of audio for a segment of speech, and when the partial that covers that segment of speech is returned from the recognizer. Measuring latencies of LAISR on each task, we find that RLM1 has a median of 0.26 seconds and a mean of 0.41s; RLM2 has a median of 0.60s and a mean of 1.48s; and SLM has a median of 1.04s and a mean of 2.10s. Since reducing latency was not the focus of this work, no speed optimizations have been made, and we believe that straightforward optimization can reduce these latencies. For example, simply turning off N-Best processing with the SLM reduces the median latency to 0.55s and the mean to 0.79s. Human reaction time to speech is roughly 0.20 seconds [19], so even without optimization the RLM latencies are not far off human performance.

6.2.5 Conclusion

In sum, LAISR produces a steady stream of partials with relatively low latency over the course of recognition. It increases the stability and accuracy of partials substantially, making it easier for a dialogue system to operate. In terms of Importance-Driven Turn-Taking, LAISR can provide better real-time context during User speech than any capabilities developed previously, which will result in fewer turn-taking errors.

Though LAISR has higher stability and accuracy than Terminal ISR, its partials are still quite unstable and inaccurate. This means that in practice, dialogue systems will need to make important decisions about which partials to use, and which to discard. This need motivated us to devise techniques for predicting when a partial is stable, and when it is accurate, which we present in the next section.

6.3 Predicting the Stability and Accuracy of Partials

As seen in the previous section, partial speech recognition results are often revised and inaccurate. In order for a dialogue system to make use of partial results, measures of both stability and confidence are crucial. A Stability Measure (SM) predicts whether the current partial is a prefix or complete match of the final recognition result (regardless of whether the final result is accurate). A Confidence Measure (CM) predicts whether the current partial is a prefix or complete match of what the user actually said. Both are useful in real systems. For instance, these two measures would enable an SDS to identify inaccurate recognition results while the user is still speaking. The SDS could then interrupt and prompt the user to start again. On the other hand, ISR allows systems to handle pauses gracefully. If the SDS recognizes that an utterance is incomplete (though stable and accurate), it could give the user more time to speak before reacting. This section is also taken from Selfridge et al. [58].

6.3.1 Experimental Procedure and Results

We use logistic regression to learn separate classifiers for SM and CM. Logistic regression is appealing because it is well-calibrated, and has shown good performance for whole-utterance confidence measures [84]. For this, we use the BXR package with default settings [21]. For Terminal and Basic ISR we use 11 features: the raw WATSON confidence score, the individual features that affect the recognition confidence score, the normalized cost, the normalized speech likelihood, the likelihoods of competing models, the best path score of word confusion network (WCN), the length of WCN, the worst probability in the WCN, and the length of N-best list. For LAISR, four additional features are used: three binary indicators of whether the partial is Terminal, Immortal or a Terminal following an Immortal, and one which gives the percentage of words in the hypothesis that are immortal.

We built stability and confidence measures for Basic ISR, Terminal ISR, and LAISR. Each of the three corpora (RLM1, RLM2, SLM) was divided in half to form a train set and test set. Regression models were trained on *all* utterances in the train set. The

Table 6.6: Equal Error Rates: Significant improvements in bold. Basic at $p < 0.016$, Terminal at $p < 0.002$, and LAISR at $p < 0.00001$

		All			Multi-Word		
Stability Measure (SM) Equal Error Rate							
		RLM 1	RLM 2	SLM	RLM 1	RLM 2	SLM
Basic	WATSON Score	13.3	13.3	12.8	15.6	16.4	15.2
	Regression	**10.7**	**11.3**	**12.3**	**13.2**	**15.2**	15.1
Terminal	WATSON Score	24.3	29.1	34.4	26.6	26.0	34.1
	Regression	**19.7**	**26.5**	**26.5**	**23.0**	24.3	**24.7**
LAISR	WATSON Score	24.7	29.3	35.0	24.0	27.0	35.3
	Regression	**19.2**	**25.6**	**25.0**	**18.4**	**23.3**	**22.7**
Confidence Measure (CM) Equal Error Rate							
Basic	WATSON Score	11.3	11.7	9.9	14.1	14.0	11.6
	Regression	**9.8**	**9.8**	9.7	**12.3**	**12.9**	**11.0**
Terminal	WATSON Score	15.1	21.1	30.6	15.7	17.4	29.3
	Regression	**11.7**	**16.8**	**20.8**	**12.1**	**14.5**	**18.4**
LAISR	WATSON Score	15.8	21.8	32.3	18.4	19.5	31.8
	Regression	**11.6**	**16.6**	**21.0**	**11.6**	**14.2**	**18.7**

resulting models were then evaluated on both All and MW utterances. As a baseline for both measures, we compare to AT&T WATSON's existing confidence score, which has been developed for full utterances. This score is used in numerous deployed commercial applications, so we believe it is a fair baseline. Although the existing confidence score is designed to predict accuracy (*not stability*), there is no other existing mechanism for predicting stability.

We first report "equal error rate" for the measures (Table 6.6). Equal error rate (EER) is the sum of false accepts and false rejects at the rejection threshold for which false accepts and false rejects are equal. Equal error rate is a widely used metric to evaluate the quality of scoring models used for accept/reject decisions. A perfect scoring model would yield an EER of 0. For statistical significance we use χ^2 contingency tables with 1 degree of freedom. It is inappropriate to compare EER across ISR methods, since the total percentage of stable or accurate partials significantly effects the EER. For example, Basic ISR has relatively low EER, but this is because it also has a relatively low number of stable or accurate partials.

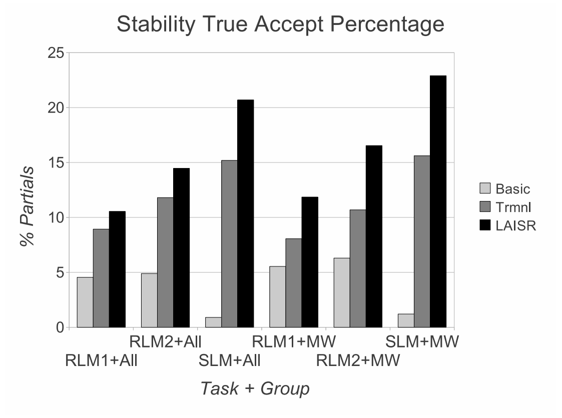
The top six rows of Table 6.6 show EER for the Stability Measure (SM). The left three columns show results on the entire test set (all utterances, of any length). On the whole, the SM outperforms the WATSON confidence scores, and the greatest improvement is a 10.0 point reduction in EER for LAISR on the SLM task. The right three columns show results on only multi-word (MW) utterances. Performance is similar to the entire test set, with a maximum EER reduction of 12.6%. The SLM MW performance is interesting, suggesting that it is easier to predict stability after at least one word has been decoded, possibly due to higher probability of immortal nodes occurring. This suggests there would be benefit in combining our method with past work that adds uniform delay to increase stability, perhaps using more context early in the utterance. This idea is left for future work.

The bottom six rows show results for the Confidence Measure (CM). We see that even when comparing our CM against the WATSON confidence scores, there is significant improvement, with a maximum of 13.1 for LAISR in the MW SLM task.

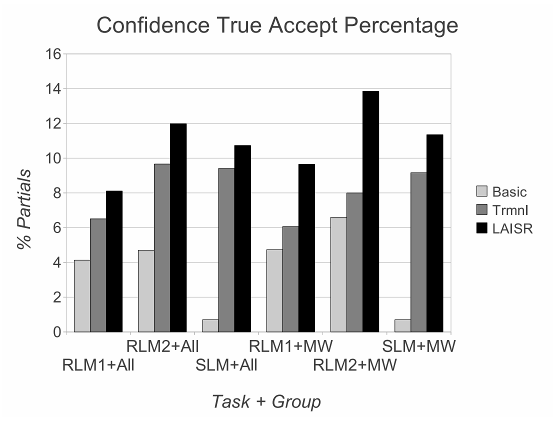
The consistent improvement shows that logistic regression is an effective technique for learning confidence and stability measures. It is most powerful when combined with LAISR, and only slightly less so with Terminal. Furthermore, though the gains are slight, it is also useful with Basic ISR, which speaks to the generality of the approach.

While equal error rate is useful for evaluating discriminative ability, when building an actual system a designer would be interested to know how often the correct partial is accepted. To evaluate this, we assumed a fixed false-accept rate of 5%, and report the resulting percentage of partials which are correctly accepted (true-accepts). Results are shown in Figure 6.2. LAISR accepts substantially more correct partials than other methods, indicating that LAISR would be more useful in practice. This result also shows a synergy between LAISR and our regression-based stability and confidence measures: not only does LAISR improve the fraction of stable and correct partials, but the regression is able to identify them better than for Terminal ISR. We believe this shows the usefulness of the additional lattice features used by the regression model built on LAISR results.

Figure 6.2: True accept percentages for stability measure (a) and confidence measure (b), using a fixed false accept rate of 5%. LAISR yields highest true accept rates, with $p < 0.0001$ in all cases.



(a) Stability measure



(b) Confidence measure

6.3.2 Conclusion

The Stability and Confidence measures both have lower Equal Error Rates than raw recognition scores when classifying partials. The improvement is greatest for LAISR, which benefits from additional features describing lattice structure. It also suggests that other incremental features such as the length of right context could be useful for predicting stability. The higher number of True Accept partials by LAISR indicates that this method is more useful to a dialogue manager than Basic or Terminal ISR. Even so, for all ISR methods there are still more useful stable partials than there are accurate ones. This suggests that both of these measures are important to the downstream dialogue manager. For example, if the partial is predicted to be stable but not correct, than the agent could possibly interrupt the user and ask them to begin again.

ISR completely restructures the conventional turn-based dialogue manager, giving the agent the opportunity to speak at any moment. The stability and accuracy measures provide the system with critical information to base its behavior on. This capability is necessary for Importance-Driven Turn-Taking since the decision to speak may often hinge on the potential accuracy of user speech.

6.4 The Incremental Interaction Manager

The previous two sections outlined methods to improve ISR results and prediction. Here, we address one aspect of applying ISR to dialogue, namely that conventional complete-phrase dialogue managers are unable to handle revision-prone partials. We introduce an *Incremental Interaction Manager* (IIM) that enables ISR to be used within the traditional turn-based dialogue management framework. The research community has spent a long time crafting the best methods to build dialogue managers, and the IIM enables ISR to be used very easily with these designs. We illustrate the IIM by pairing it with a POMDP-based dialogue manager yields a substantial improvement in accuracy for incremental recognition results at the dialogue level. This section is taken from Selfridge, Arizmendi, Heeman, and Williams [**?**].

The IIM confers many of the benefits of ISR without requiring modification to a traditional dialogue manager, the primary benefit being an improved turn-taking capability. Incremental Speech Recognition is critical to the Importance-Driven approach and the IIM provides a means of building Importance-Driven systems without completely restructuring the dialogue manager.

The chapter is organized as follows. Section 6.4.1 describes the IIM, Section 6.4.2 describes the POMDP integration, Sections 6.4.3 and 6.4.4 describe experiments and results, and Section 6.4.5 concludes.

Table 6.7: Example IIM operation. P = partial ISR result; A = dialogue action.

ISR	IIM	Original DM state	Copied DM state	DM Action
Prompt: “Where are you leaving from?”				
yew	Rej. P	0	0	-
ridge	Acc. P / Rej. A	0	0	“I’m sorry...”
mckee	Acc. P / Acc. A	0	1	“Ok, Mckee...”
mckeesport	Acc. P / Acc. A	0	2	“Ok, Mckeesport..”
mckeesport center	Acc. P / Rej. A	0	2	“Ok, Mckeesport..”
Prompt: “Ok, Mckeesport. Where are you going to?”				
pitt	Acc. P / Rej. A	2	4	“I’m sorry...”
pittsburgh	Acc. P / Acc. A	2	5	“Ok, Pittsburgh...”

6.4.1 Incremental Interaction Manager Operation

The Incremental Interaction Manager (IIM) mediates communication between the incremental speech recognizer and the DM. The key idea is that the IIM evaluates potential dialogue moves by applying ISR results to temporary instances of the DM. The IIM *copies* the current state of the DM, provides the copied DM with a recognition result, and inspects the action that the copied DM would take. If the DM design does *not* force a state transition following a result then the DM supplies the action without copying. If the action does not sufficiently advance the dialogue (such as re-asking the same question), the action is rejected and the copied DM is discarded. If the action advances the dialogue (such as asking for or providing new information), then that action is immediately executed.

The system should gracefully handle revisions following a premature action execution, and a copying procedure is a viable solution for any DM. When a revision is received, a *second* copy of the original DM is made and the new ISR result is passed to that second copy; if the second copy takes an action that advances the dialogue *and is different* from the action generated by the first copy, then the first action is terminated, the first copy of the DM is discarded, the second action is initiated, and the second copy assumes the position of the first copy. Additional revisions can be handled by following the same procedure. Terminating a speech action and immediately starting another can be jarring: “Say a city

/ Ok, Boston...". This unpleasantness can be mitigated by preceding actions with either a sound or simple silence, at the expense of some response delay. Once recognition is complete, the copied DM is installed as the new original DM.

Many ISR results can be discarded before passing them to the DM. First, only incremental results that could correspond to a complete user utterance are considered: incomplete results are discarded and never passed to the DM. In addition, ISR results are often unstable, and it is undesirable to proceed with an ISR result if it will very likely be revised. Thus each candidate ISR result is scored for stability using techniques presented in Section 6.3, and results with scores below a manually-set threshold are discarded.

Table 6.7 shows an example of the recognizer, the IIM, and the DM. For sake of clarity, stability scores are not shown. The system asks "Where are you leaving from?" and the user answers "Mckeesport Center." The IIM receives five partials, rejecting the first, *yew*, because its stability score is too low (not shown). With the second, *ridge*, it copies the DM, passes *ridge* to the copy, and discards the action of the copied DM (also discarded) because it does not advance the dialogue. It accepts and begins to execute the action generated by the third partial, *mckee*. The fourth partial revises the action, and the fifth action is rejected since it is the same. The original DM is then discarded and the copied DM state is installed in its place.

Overall, the IIM enables a turn-based DM to enjoy many of the benefits of ISR – in particular, the ability to make turn-taking decisions with a complete account of the dialogue history.

6.4.2 Integrating ISR with a POMDP-based dialogue manager

A dialogue manager based on a partially observable Markov decision process (POMDP DM) tracks a probability distribution over multiple hidden dialogue states called a *belief state* [82].[3] As such, POMDP DMs readily make use of the entire ASR N-Best list, even for low-confidence results — the confidence level of each N-Best list item contributes proportionally to the probability of its corresponding hidden state.

[3]It also uses reinforcement learning to choose actions, although in this paper we are not concerned with this aspect.

It is straightforward to integrate ISR and a POMDP DM using the IIM. Each item on the N-Best list of an incremental result is assigned a confidence score [80] and passed to the POMDP DM as if it were a complete result, triggering a belief state update. Note that this approach is not *predicting* future user speech from partial results [13, 37], but rather (tentatively) assuming that partial results are complete.

The key benefit is that a belief state generated from an incremental result incorporates all of the contextual information available to the system *from the start of the dialogue until the moment of that incremental result.* By comparison, an isolated incremental result includes only information from the current utterance. If the probability models in the POMDP are estimated properly, belief states should be more accurate than isolated incremental results.

6.4.3 Experimental design

For our experiments we used a corpus of 1037 calls from real users to a single dialogue system that provides bus timetable information for Pittsburgh, PA (a subsequent version of Williams williams2011empirical). This dialogue system opened by asking the caller to say a bus route number or "I don't know"; if the system had insufficient confidence following recognition, it repeated the question. We extracted the first 3 responses to the system's bus route question. Often the system did not need to ask 3 times; our experimental set contained 1037 calls with one or more attempts, 586 calls with two or more attempts, and 356 calls with three or more attempts. These utterances were all transcribed, and tagged for the bus route they contained, if any: 25% contained neither a route nor "I don't know".

We ran incremental speech recognition on each utterance using Lattice-Aware Incremental Speech Recognition [58] on the AT&T WATSON^{SM} speech recognizer [23] with the same rule-based language models used in the production system. On average, there were 5.78, 5.44, and 5.11 incremental results per utterance (plus an utterance-final result) for the first, second, and third attempts. For each incremental result, we noted its time stamp and interpretation: *correct*, if the interpretation was present and correct, otherwise *incorrect*. Each incremental result included an N-Best list, from which we determined oracle accuracy: *correct*, if the correct interpretation was present anywhere on the most recent ISR N-Best list, otherwise *incorrect*.

Each incremental result was then passed to the IIM and POMDP DM. The models in the POMDP DM were estimated using data collected from a different (earlier) time period. When an incremental result updated the belief state, the top hypothesis for the route was extracted from the belief state and scored for correctness. For utterances in the first attempt, the belief state was initialized to its prior; for subsequent attempts, it incorporated all of the prior (whole-turn) utterances. In other words, each attempt was begun assuming the belief state had been running up to that point.

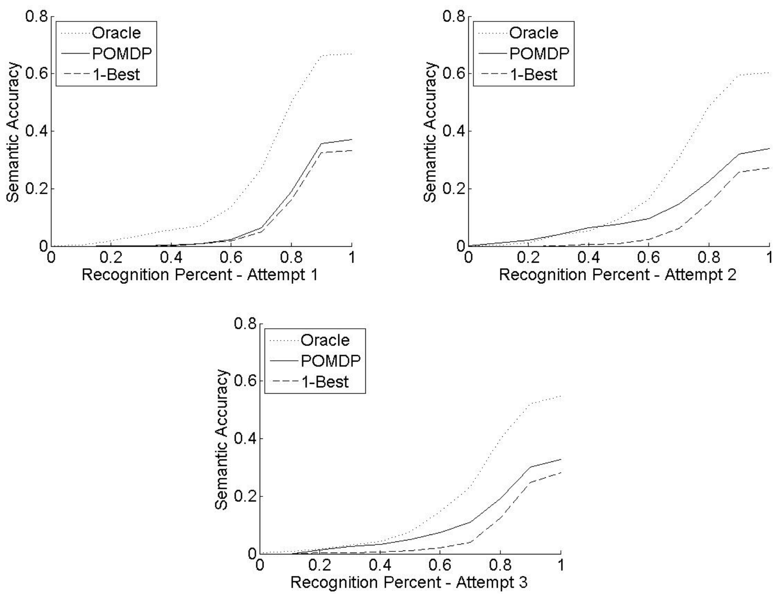


Figure 6.3: Instantaneous semantic accuracy of incremental results, excluding phrase-final results

6.4.4 Results and Discussion

We present results by showing instantaneous semantic accuracy for the raw incremental result (baseline), the top belief state, and oracle. Instantaneous semantic accuracy is shown with respect to the *percent* of the total recognition time the partial is recognized at. An utterance is incorrect if it has no incremental result before a certain percentage.

We show 2 sets of plots. Figure 6.3 shows only incremental recognition results and

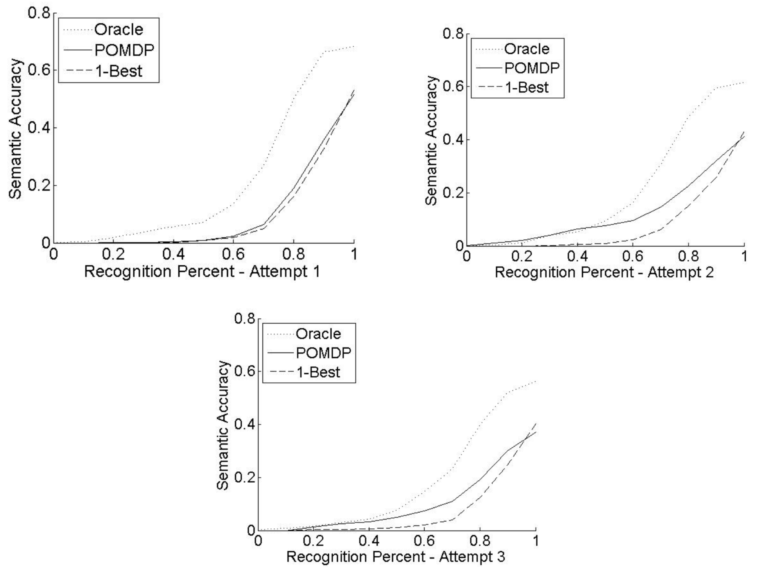


Figure 6.4: Instantaneous semantic accuracy of incremental and phrase-final results

excludes the end-of-utterance (*phrase*) results; Figure 6.4 shows incremental recognition results and includes phrase results. It is useful to view these separately since the phrase result, having access to all the speech, is substantially more accurate than the incremental results.

Figure 6.3 shows that the POMDP is more accurate than the raw incremental result (excluding end-of-phrase results). Its performance gain is minimal in attempt 1 because the belief is informed only by the prior. In attempt 2 and 3, the gain is larger since the belief also benefits from the previous attempts. Since the top POMDP result in subsequent attempts is sometimes already correct (because it incorporates past recognitions), the POMDP sometimes meets and occasionally exceeds the oracle during the early portions of attempts 2 and 3.

Figure 6.4 shows that when end-of-phrase recognition results are included, the benefit of the belief state is limited to the initial portions of the second and third turns. This is because the POMDP models are not fit well to the data: the models were estimated from an earlier version of the system, with a different user base and different functionality. Identifying and eliminating this type of mismatch is an important issue and has been studied before [79].

Taken as a whole, we find that using belief tracking increases the accuracy of partials by over 8% (absolute) in some cases. Even though the final phrase results of the 1-best list are more accurate than the belief state, the POMDP shows better accuracy on the volatile incremental results. As compared to the whole utterance results, incremental results have lower 1-best accuracy, yet high oracle accuracy. This combination is a natural fit with the POMDPs belief state, which considers the whole N-Best list, effectively re-ranking it by synthesizing information from dialogue history priors.

6.4.5 Conclusion

This work has taken a step toward integrating ISR and POMDP-based dialogue systems. The Incremental Interaction Manager (IIM) enables a traditional turn-based DM to make use of incremental results and enjoy many their benefits. When this IIM is paired with a POMDP DM, the interpretation accuracy of incremental results improves substantially. These results indicate that the IIM provides a stable platform to build Importance-Driven systems.

Chapter 7

Prediction-Based Barge-in Response

7.1 Introduction

Chapter 3 introduced the theory of Importance-Driven Turn-Taking and Chapter 6 outlined various approaches to improve the use of incremental speech recognition for spoken dialogue systems. The present chapter proposes and evaluates a barge-in processing approach whose behavior is based on importance-driven turn-taking and executed using incremental speech processing tools described in Chapter 6. This chapter is based on Selfridge et al. [60].

This chapter proposes and evaluates a barge-in processing method that focuses on handling Non-Understood Barge-Ins (NUBIs). The relevant related work is described in 2.6.1. The Prediction-based Barge-in Response (PBR) model continuously predicts interpretation success by applying adaptive thresholds to incremental recognition results. In our view, *predicting* whether the recognition will be understood has far more utility than detecting whether the barge-in is truly system directed speech as, for many domains, we feel only understandable input has more discourse importance than system speech. If the input is predicted to be understood, the prompt is paused. If it is predicted to be, or recognized as, NUBI, the prompt is resumed. Using this method, the system may resume speaking before recognition is complete and will never initiate a clarifying sub-dialogue in response to a NUBI. The PBR model was implemented in a public Lets Go! statistical dialogue system [47], and we compare it with a system using standard barge-in methods. We find the PBR model has a significantly better task success rate and efficiency.

Table 7.1: System response to Non-Understood Barge-In (NUBI)

Baseline	Ok, sixty one <*NUBI*> Sorry, say a bus route like twenty eight x
PBR	Ok, sixty one <*NUBI*> sixty one c. Where are you leaving from?

Table 7.1 illustrates the NUBI responses produced by the standard barge-in (Baseline) and PBR models. After both prompts are paused, the standard method initiates a clarifying sub-dialogue whereas PBR resumes the prompt.

The PBR model is predicated on the idea that, in some domains, the systems prompt is more important than any other input save successfully understood user speech. The PBR model uses adaptive thresholds to determine its turn-taking behavior and, while reinforcement learning is not used to determine the system's own "bid", the system's bid can be construed as the these adaptive thresholds. Furthermore, the system does to apply these adaptive bids to judge the importance of the user's speech but rather applies them to evaluate the likelihood that the input well-formed system-directed speech at all. So, while this chapter does not directly and explicitly evaluate importance-driven turn-taking, it demonstrates the value of applying the same theoretical components to a slightly different challenge; namely, effective and efficient barge-in handling.

7.2 Prediction-based Barge-in Response

The PBR model is characterized by three high-level states: State 1 (Speaking Prediction), whose goal is to pause the prompt if stability scores predict understanding; State 2 (Silent Prediction), whose goal is to resume the prompt if stability scores and the incremental recognition rate predict non-understanding; and State 3 (Completion), which operates on the final recognition result, and resumes the prompt unless the recognition is understood and the new speech act will advance the dialogue. Here, we define "advancing the dialogue" to be any speech act that does not start a clarifying sub-dialogue indicating a NUBI. Transitions between State 1 and 2 are governed by adaptive thresholds — repeated resumptions suggest the user is in a noisy environment, so each resumption increases the threshold required to advance from State 1 to State 2 and decreases the threshold required

Table 7.2: Background noise and User speech ISR

Background Noise		User Utterance	
Partial	Stab. Scr.	Partial	Stab. Scr.
one	0.134	six	0.396
two	0.193	sixty	0.542
six	0.127	fifty one	0.428
two	0.078	sixty one a	0.491

to advance from State 2 to State 1. A high-level comparison of the standard model and our approach is shown in Figure 7.1; a complete PBR state diagram is provided in the Appendix.

7.2.1 State 1: Speaking Prediction

In State 1, Speaking Prediction, the system is both speaking and performing ISR. The system scores each partial for stability, predicting the probability that it will remain "stable" – i.e., will not be later revised – using a logistic regression model [58]. This model uses a number features related to the recognizer's generic confidence score, the word confusion network, and lattice characteristics. Table 7.2 shows partial results and stability scores for two example inputs: background noise on the left, and the user saying "sixty one a" on the right.

State 1 relies on the internal threshold parameter, T_1. If a partial's stability score falls below T_1, control remains in State 1 and the partial result is discarded. If a stability score meets T_1, the prompt is paused and control transitions to State 2. T_1 is initially set to 0 and is adapted as the dialogue progresses. The adaptation procedure is described below in Section 7.2.4. If a *final* recognition result is received, control transitions directly to State 3. Transitioning from State 1 to State 2 is only allowed during the middle 80% of the prompt; otherwise only transitions to State 3 are allowed.[1]

[1]We hypothesized that people will rarely respond to the current prompt during the first 10% of prompt time as overlaps at the beginning of utterances are commonly initiative conflicts [85]. Users may produce early-onset utterances during the last 10% that should not stop the prompt as it is not an "intentional" barge-in.

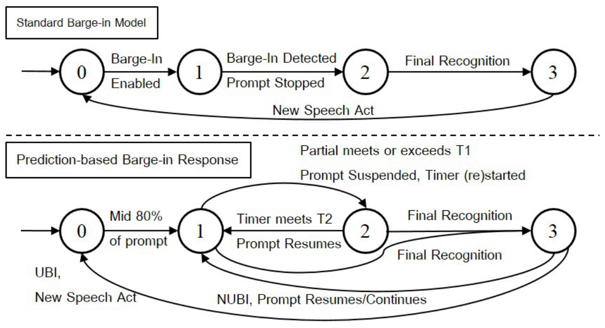


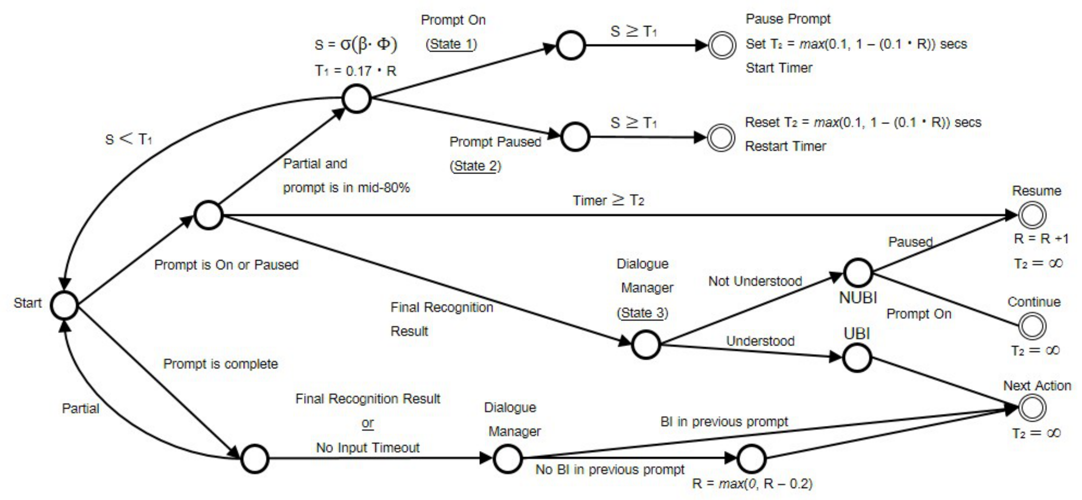
Figure 7.1: The Standard Barge-in and PBR Models

7.2.2 State 2: Silent Prediction

Upon entering State 2, Silent Prediction, the prompt is paused and a timer is started. State 2 requires continuous evidence (at least every T_2 ms) that the ISR is recognizing valid speech and each time a partial result that meets T_1 is received, the timer is reset. If the timer reaches the time threshold T_2, the prompt is resumed and control returns to State 1. T_2 is initially set at 1.0 seconds and is adapted as the dialogue progresses. Final recognition results trigger a transition to State 3.

The resumption prompt is constructed using the temporal position of the VAD specified speech start to find the percentage of the prompt that was played up to that point. This percentage is then reduced by 10% and used to create the resumption prompt by finding the word that is closest to, but not beyond, the modified percentage. White space characters and punctuation are used to determine word boundaries for text-to-speech prompts, whereas automatically generated word-alignments are used for pre-recorded prompts. This diagram represents the possible operating positions the Prediction-based Barge-in Response model can be in. If the prompt is complete, the PBR model applies the dialogue policy to the final recognition result and initiates the on-policy speech act. If the prompt was finished without being paused it decrements R. In the latter case (barge-in), it operates using the three states as described in Section 2. When a partial is recognized the Stability Score is computed and compared to the T_1 threshold parameter. If the score is below T_1 the partial is discarded. Otherwise, if the model is in State 1 (the prompt is on) the prompt is paused, a timer is started, and control transitions to State 2. If the

Figure 7.2: PBR State Diagram



model is in State 2 the timer is restarted. After transitioning to State 2, control only returns to State 1 if the timer exceeds T_2. At this time, the prompt is resumed and the resumption parameter R is incremented. Control immediately transitions to State 3 if a final recognition result is received. The result is evaluated by the dialogue manager, and the new speech act is returned. If the speech act indicates the recognition was not understood successfully, the system either resumes (if in State 1) or continues (if in State 2). In the case of resumption, R is incremented. If the new speech act indicates understanding success, the new speech is immediately produced.

7.2.3 State 3: Completion

State 3, Completion, is entered when a final recognition result has been received. The function of State 3 is to determine whether the dialogue policy for the final recognition result will advance the dialogue or not. Here, the PBR model relies on the ability of the dialogue manager (DM) to produce a speculative action without transitioning to the next dialogue state. If the new action will not advance the dialogue, it is discarded and the recognition is NUBI. However, if it *will* advance the dialogue then it is classified as an Understood Barge-In (UBI). In the case of NUBI, the system either continues speaking

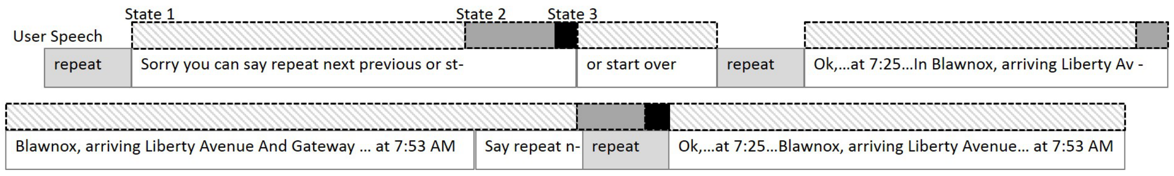

State 1
State 2
State 3
User Speech
repeat
Sorry you can say repeat next previous or st-
or start over
repeat
Ok,...at 7:25...In Blawnox, arriving Liberty Av -
Blawnox, arriving Liberty Avenue And Gateway ... at 7:53 AM
Say repeat n-
repeat
Ok,...at 7:25...Blawnox, arriving Liberty Avenue... at 7:53 AM

Figure 7.3: Example dialogue fragment of PBR Model

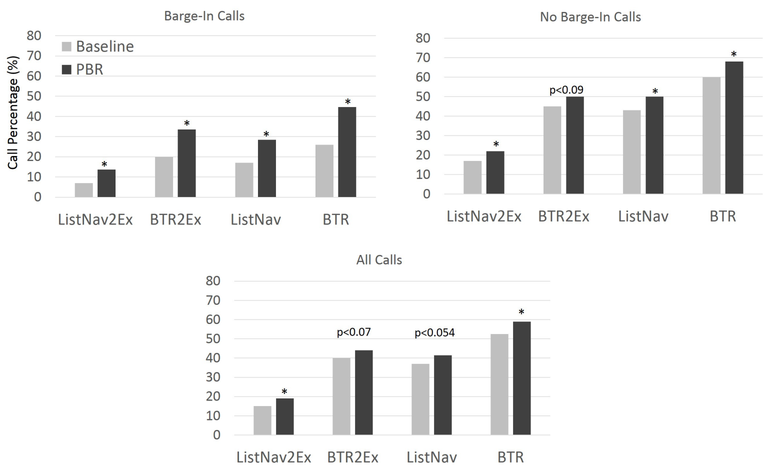
or resumes the current prompt (transitioning to State 1). In the case of UBI, the system initiates the new speech act after playing a short reaction sound and the DM transitions to the next dialogue state. This reaction sound precedes *all* speech acts outside the barge-in context but is *not* used for resumption or timeout prompts. Note that by depending on the new speech act, our model does not require access to the DM's internal understanding or confidence scoring components.

7.2.4 Threshold adjustments

States 1 and 2 contain parameters T_1 and T_2 that are adapted to the user's environment. T_1 is the stability threshold used in State 1 and State 2 that controls how stable an utterance must be before the prompt should be paused. In quiet environments — where only the user's speech produces partial results — a low threshold is desirable as it enables near-immediate pauses in the prompt. Conversely, noisy environments yield many spurious partials that (in general) have much lower stability scores, so a higher threshold is advantageous. T_2 is the timing threshold used to resume the prompt *during* recognition in State 2. In quiet environments, a higher threshold reduces the chance that the system will resume its prompt during a well-formed user speech. In noisy environments, a lower threshold allows the system to resume quickly as the NUBI likelihood is greater.

Both T_1 and T_2 are dependent on the number of system resumptions, as we view the action of resuming the prompt as an indication that the threshold is not correct. With every resumption, the parameter R is incremented by 1 and, to account for changing environments, R is decremented by 0.2 for every full prompt that is not *paused* until it reaches 0. Using R, T_1 is computed by $T_1 = 0.17 \cdot R$, and T_2 by $T_2 = argmax(0.1, 1 -$

Figure 7.4: Estimated success rate for the PBR and Baseline systems. Stars indicate p<0.018 with χ^2 test.



$(0.1 \cdot R))$.[2]

7.2.5 Method Discussion

The motivation behind the PBR model is both theoretical and practical. According to Selfridge and Heeman selfridge2010importance, turn-taking is best viewed as a collaborative process where the turn assignment should be determined by the importance of the utterance. During barge-in, the system is speaking and so should only yield the turn if the user's speech is more important than its own. For many domains, we view non-understood input as less important than the system's prompt and so, in this case, the system should not release the turn by stopping the prompt and initiating a clarifying sub-dialogue. On the practical side, there is a high likelihood that non-advancing input is not system directed, to which the system should neither consume, in terms of belief state updating, nor respond to, in terms of asking for clarification. In the rare case of non-understood system directed speech, the user can easily repeat their utterance.

The PBR approach differs from standard barge-in approaches in several respects. First, standard barge-in stops the prompt (i.e., transitions from State 1 to State 2) if either the

[2]The threshold update values were determined empirically by the authors.

VAD or the partial hypothesis suggests that there is speech; our approach — using acoustic, language model, and lattice features — predicts whether the input is likely to contain an interpretable recognition result. Second, standard barge-in uses a static threshold; our approach uses dynamic thresholds that adapt to the user's acoustic environment. Parameter adjustments are straightforward since our method automatically classifies each barge-in as NUBI or UBI. In practice, the prompt will be paused incorrectly only a few times in a noisy environment, after which the adaptive thresholds will prevent incorrect pauses at the expense of being less responsive to true user speech. If the noise level decreases, the thresholds will become more sensitive again, enabling swifter responses. Finally, with the exception of Strom and Seneff, standard approaches always discard the prompt; our approach can resume the prompt if recognition is not understood or is proceeding poorly, enabling the system to resume speaking before recognition is complete. Moreover, resumption yields a natural user experience as it often creates a repetition disfluency ("Ok, sixty - sixty one c"), which are rarely noticed by the listener [39].

An example dialogue fragment is shown in Figure 7.3, with the state transitions shown above. Note the transition from State 2 to State 1, which is the system resuming speech *during* recognition. This recognition stream, produced by non-system directed user speech, does not end until the user says "repeat" for the last time.

7.3 Evaluation Results

The PBR model was evaluated during the Spoken Dialog Challenge 2012-2013 in a live Lets Go! bus information task. In this task, the public can access bus schedule information during off hours in Pittsburgh, PA via a telephonic interaction with a dialogue system [47]. The task can be divided into five sub-tasks: route, origin, destination, date/time, and bus schedules. The last sub-task, bus schedules, provides information to the user whereas the first four gather information. We entered two systems using the same POMDP-based DM [83]. The first system, the "Baseline", used the standard barge-in model with VAD barge-in detection and barge-in disabled in a small number of dialogue states that appeared problematic during initial testing. The second system used the PBR model with an Incremental Interaction Manager [59] to produce speculative actions in State 3. The public called both systems during the final weeks of 2011 and the start of 2012. The DM applied a logistic regression based confidence measure to determine whether the recognition was understood. Both systems used the AT&T WATSON^{SM} speech recognizer [23] with the same sub-task specific rule-based language models. The beam width was set to maximize accuracy while still running faster than real-time. The PBR system used a WATSON modification to output lattice-aware partial results.

Call and barge-in statistics are shown in Table 7.3. Here, we define (potential) barge-in (somewhat imprecisely) as a full recognition that at some point overlaps with the system prompt, as determined by the call logs. We show the calls with barge-in *before* the bus schedule sub-task was reached (BI-BS) and the calls with barge-in during any point of the call (BI All). Since the Baseline system only enabled barge-in at specific points in the dialogue, it has fewer instances of barge-in (Total Barge-In) and fewer barge-in calls. Regretfully, due to logging issues with the PBR system, recognition specific metrics such as Word Error Rate and true/false barge-in rates are unavailable.

7.3.1 Estimated Success Rate

We begin by comparing the success rate and efficiency between the Baseline and PBR systems. Since task success can be quite difficult to measure, we use four increasingly stringent task success definitions: Bus Times Reached (BTR), where success is achieved if the call reaches the bus schedule sub-task; List Navigation (List Nav.), where success is achieved if the user says ‘’next”, “previous”, or “repeat” — the intuition being that if the user attempted to navigate the bus schedule sub-task they were somewhat satisfied with the system’s performance so far; and Immediate Exit (BTR2Ex and ListNav2Ex), which

Table 7.3: Baseline and PBR call/barge-in statistics.

	Baseline	PBR
Total Calls	1027	892
BI-BS	228 (23%)	345 (39%)
BI All	281 (27%)	483 (54%)
Total Barge-In	829	1388

further constrains both of the previous definitions to only calls that finish directly after the initial visit to the bus times sub-task. All definitions were automatically determined (not manually labeled). Figure 9.5 shows the success rate of the PBR and Baseline systems for all four definitions of success. It shows, from left to right, Barge-In, No Barge-In (NBI), and All calls. Here we restrict barge-in calls to those where barge-in occurred prior to the bus schedule task being reached.

For the calls with barge-in, a χ^2 test finds significant differences between the PBR and Baseline for all four task success definitions. However, we also found significant differences in the NBI calls. This was surprising since, when barge-in is not triggered, both systems are ostensibly the same. We speculate this could be due to the Baseline's barge-in enabling strategy: an environment that triggers barge-in in the Baseline would always trigger barge-in in the PBR model, whereas the converse is *not* true since the Baseline only enabled barge-in *intermittently.* This means that there is a potential mismatch when separating the calls based on barge-in, and so the fairest comparison is using *All* the calls. This is shown on the far right of Figure 9.5. We find that, while the effect is not as large, there are significant differences in the success rate for the PBR model for the most and least stringent success definition, and very strong trends for the middle two definitions ($p < 0.07$ for BTR2Ex and $p < 0.054$ for List Nav.). Taken as a whole, we feel this offers compelling evidence that the PBR method is more effective: i.e. yields higher task completion.

Next, we turn our attention to task efficiency. For this, we report the amount of clock time from the beginning of the call to when the Bus Schedule sub-task was reached. Calls that do not reach this sub-task are obviously excluded, and PBR times are adjusted for the reaction sound (explained in Section 7.2.3). Task efficiency is reported by cumulative percentage in Figure 7.5. We find that, while the NBI call times are nearly identical for both systems, the PBR barge-in calls are much faster than the Baseline calls. Here, we do not feel the previously described mismatch is particularly problematic as all the calls reached the goal state and the NBI are nearly identical. In fact, as more NUBI should actually *reduce* efficiency, the potential mismatch only strengthens the result.

Taken together, these results provide substantial evidence that the PBR model is more effective and more efficient than the Baseline. In order to explain PBR's performance we

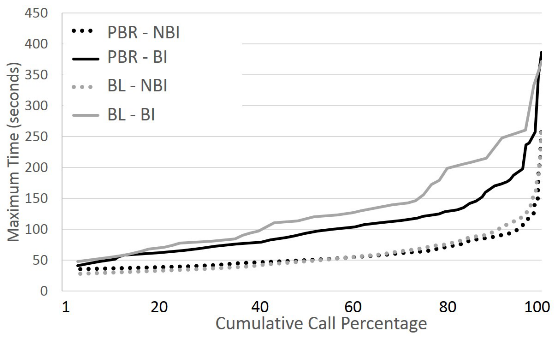


Figure 7.5: Seconds from beginning of dialogue to reaching the Bus Schedule Information sub-task

explore the effect of prediction and resumption in isolation.

7.3.2 State 1: Speaking Prediction

State 1 is responsible for pausing the prompt, the goal being to pause the prompt for UBI input and not to pause the prompt for NUBI input. The prompt is paused if a partial's stability score meets or exceeds the T_1 threshold. We evaluate the efficacy of State 1 and T_1 by analyzing the statistics of NUBI/UBI input and Paused/Not Paused (hereafter *Continued*) prompts. Since resuming the prompt during recognition affects the recognition outcome, we restrict our analysis to recognitions that do not transition from State 2 back to State 1. For comparison we show the overall UBI/NUBI percentages for the Baseline and PBR systems. This represents the recognition distribution for the live Baseline VAD detection and off-line speculation for the PBR model. Recall PBR *does* have VAD activation preceding partial results and so the off-line PBR VAD shows how the model *would* have behaved if it only used the VAD for detection, as the Baseline does.

Table 7.4 provides a number of percentages, with three micro-columns separated by dashes ('-') for T_1. The first micro-column shows the percentage of UBI/NUBI that either Paused or Continued the prompt (sums to 100 horizontally). The second micro-column shows the percentage of Paused/Continued that are UBI/NUBI (sums to 100 vertically). The third micro-column shows the percentage of each combination (e.g. UBI and Paused) over all the barge-in recognitions. The VAD columns show the percentage of UBI/NUBI

Table 7.4: Evaluation of T_1, off-line PBR, and Baseline VAD. For T_1 we respectively ('-' split) show the UBI/NUBI % that are Paused/Continued, the Paused/Continued % that are UBI/NUBI, and the percentage over all recognitions that (would) pause the prompt.

	T_1 (%)		VAD (%)	
	Paused	Continued	PBR	BL
UBI	72-40-26	28-29-10	36	54
NUBI	61-60-39	39-71-25	64	46

We first look at UBI/NUBI percentage that are Paused/Continued (first micro-column): We find that 72% of UBI are paused and 28% are Continued versus 61% of NUBI that are Paused with 39% Continued. We now look at the Paused/Continued percentage that are UBI/NUBI (second micro-column): We find that 40% of Paused are UBI and 60% are NUBI, whereas 29% of Continued are UBI and 71% are NUBI. So, while T_1 suspends the prompt for the majority of NUBI (not desirable, though expected since T_1 starts at 0), it has high precision when continuing the prompt. This reduces the number of times that the prompt is paused erroneously for NUBI while minimizing incorrect (UBI) continues. This is clearly shown by considering all of the recognitions (third micro-column). We find that PBR erroneously paused the prompt for 39% of recognitions, as opposed to 46% of recognitions for the Baseline and 64% for the off-line PBR. This came at the cost of reducing the number of correct (UBI) pauses to 26% from 36% (off-line PBR) and 54% (Baseline VAD).

The results show that the T_1 threshold had modest success at discriminating UBI and NUBI; while continuing the prompt had quite a high precision for NUBI, the recall was substantially lower. We note that, since erroneous pauses lead to resumptions and erroneous continues still lead to a new speech act, there is minimal cost to these errors. Furthermore, in our view, reducing the percentage of recognitions that pause and resume the prompt is more critical as these needlessly disrupt the prompt. In this, T_1 is clearly effective, reducing the percentage from 64% to 39%.

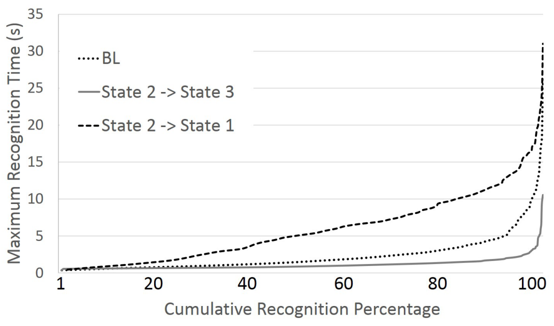


Figure 7.6: Secs from Speech Start to Final Result

7.3.3 State 2: Silent Prediction

State 2 governs whether the prompt will remain paused or be resumed *during* incremental recognition. This decision depends on the time parameter T_2, which should trigger resumptions for NUBIs. Since the act of resuming the prompt during recognition changes the outcome of the recognition, it is impossible to evaluate how well T_2 discriminated recognition results. However, we *can* evaluate the effect of that resumption by comparing UBI percentages between the PBR and Baseline systems. We first present evidence that T_2 is most active during longer recognitions, and then show that longer Baseline recognitions have a lower UBI percentage than longer PBR recognitions specifically because of T_2 resumptions. “Recognitions” refers speech recognition results, with “longer” or “shorter” referring to the clock time between speech detection and the final recognition result.

We first report the PBR and Baseline response and recognition time. We separate the PBR barge-in recognitions into two groups: State 2$\rightarrow$State 3, where the system *never* transitions from State 2 to State 1, and State 2$\rightarrow$State 1, where the system resumes the prompt *during* recognition, transitioning from State 2 to State 1. The cumulative percentages of the time from speech detection to final recognition are shown in Figure 7.6. We find that the State 2$\rightarrow$State 3 recognitions are far faster than the Baseline recognitions, which in turn are far faster than the State 2$\rightarrow$State 1 recognitions. State 2$\rightarrow$State 3 recognitions are much faster than the Baseline recognitions, implying that T_2 has greater activation during longer recognitions. Given this, the overall barge-in response time for PBR should be faster than the Baseline (as the PBR system is resuming where

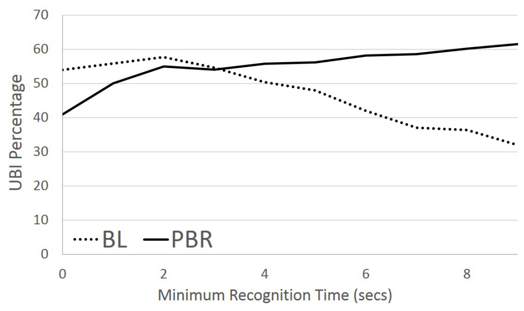


Figure 7.7: UBI % by minimum recognition time

the Baseline is silent). Indeed this is the case: the PBR system's overall mean/median response time is 1.58/1.53 seconds whereas Baseline has a mean/median response time of 2.61/1.8 seconds.

The goal of T_2 is for the system to resume when recognition is proceeding poorly, and we have shown that it is primarily being activated during longer recognitions. If T_2 is functioning properly, recognition length should be inversely related to recognition performance, and longer recognitions should be less likely to be understood. Furthermore, if T_2 resumption improves the user's experience then longer PBR recognitions should perform better than Baseline recognitions of comparable length. Figure 7.7 presents the UBI percentage by the minimum time for recognitions that reach State 2. We find that, when all recognitions are accounted for (0 second minimum), the Baseline has a higher rate of UBI. However, as recognition time increases the Baseline UBI percentage decreases (suggesting successful T_2 functioning) whereas the PBR UBI percentage actually increases. Since longer PBR recognitions are dominated by T_2 resumptions, we speculate this improvement is driven by users repeating or initiating new speech that leads to understanding success, as the PBR system is responding where the Baseline system is silent.

7.3.4 Resumption

The PBR model relies on resumption to recover from poor recognitions, either produced in State 2 or State 3. Instead of a resumption, the Baseline system initiates a clarifying subdialogue when a barge-in recognition is not understood. We compare these two behaviors

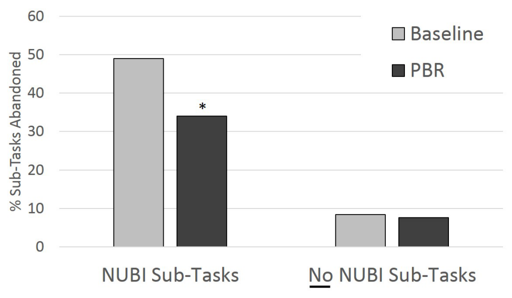


Figure 7.8: Sub-Task Abandonment Rate. NUBI is different at $p < 0.003$

using the call abandonment rate — the user hangs-up — of sub-tasks with and without NUBI. Here, we exclude the Bus Schedule sub-task as it is the goal state.

Figure 7.8 shows the call abandonment rate for sub-tasks that either have or do not have NUBI. We find that there is a significant difference in abandoned calls for NUBI sub-tasks between the two systems (33% vs 48%, $p < 0.003$ using a χ^2 test), but that there is no difference for the calls that do not have NUBI (7.6% vs 8.4%). This result shows that prompt resumption is viewed far more favorably by users than initiating a clarifying sub-dialogue.

7.4 Discussion and Conclusion

The above results offer strong evidence that the PBR model increases task success and efficiency, and we found that all three states contribute to the improved performance by creating a more robust, responsive, and natural interaction. T_1 prediction in State 1 reduced the number of spurious prompt suspensions, T_2 prediction in State 2 led to improved understanding performance, and prompt resumption (States 2 and 3) reduced the number of abandoned calls.

An important feature of the Prediction-based Barge-in Response model is that, while it leverages incremental speech processing for barge-in processing, it does not require an incremental dialogue manager to drive its behavior. Since the model is also domain independent and does not require access to internal dialogue manager components, it can easily be incorporated into any existing dialogue system. However, one limitation of the current model is that the prediction thresholds are hand-crafted. We also believe that substantial improvements can be made by explicitly attempting to predict eventual understanding instead of using the stability score and partial production rate as a proxy. Furthermore, the PBR model does not distinguish between the causes of the non-understanding, specifically whether the input contained in-domain user speech, out-of-domain user speech, or background noise. This case is specifically applicable in domains where system and user speech are in the same channel, such as interacting via speaker phone. In this context, the system *should* be able to initiate a clarifying sub-dialogue and release the turn, as the system must be more sensitive to the shared acoustic environment and so its current prompt may be less important than the user's non-understood utterance.

The results challenge a potential assumption regarding barge-in: that barge-in indicates greater user pro-activity and engagement with the task. One of the striking findings was that dialogues with barge-in are slower and less successful than dialogues without barge-in. This suggests that, for current systems, dialogues with barge-in are more indicative of environmental difficulty than user pro-activity. The superior performance of the PBR model, which is explicitly resistant to non-system directed speech, implies that dominant barge-in models will have increasingly limited utility as spoken dialogue systems become more prevalent and are used in increasingly difficult environments. Furthermore, within the context of overall dialogue systems, the PBR model's performance emphasizes the importance of continuous processing for future systems.

This chapter has proposed and evaluated the Prediction-based Barge-in Response model. This model's behavior is driven by continuously predicting whether a barge-in recognition will be understood successfully, and combines incremental speech processing techniques with a prompt resumption procedure. Using a live dialogue task with real users, we evaluated this model against the standard barge-in model and found that it led to improved performance in both task success and efficiency. While not directly governed by importance-driven turn-taking, the PBR model applies concepts rooted in the importance-driven approach to a different problem and provides further evidence of the efficacy and utility of importance-driven turn-taking.

Chapter 8

A Temporal Simulation for Advanced Turn-Taking

8.1 Introduction

In this chapter, we describe an approach to create a simulator that can be used to develop and evaluate an importance-driven turn-taking system. The simulator has two primary uses: (1) to provide a realistic environment to the craft the MDPs necessary for learning to occur and (2) to train a "seed" policy that the system uses as the foundation for training with real users. One requirement of the simulator is that *time* must be modeled correctly. The other requirement is that simulated incremental speech recognition, which provides partial results prior to user completion, must produce synthetic results that are characteristic of authentic partials. We first begin with introducing a simulation architecture that addresses the time requirement. We then propose a method for synthesizing and evaluating partial results, which combines modified speech synthesis with a true incremental speech recognizer.

Section 2.7 provides background on turn-taking simulation as well as the speech recognitions simulations. One key point is that, to our knowledge, there has been no attempt to combine speech recognition simulation with that of turn-taking.

8.2 Modeling Time

This section presents a temporal simulator for dialogue that models the timing and content of both user and system speech, as well as that of incremental speech recognition (ISR) and voice activity detection (VAD). The ISR and VAD are modeled quite simply and are probably not sufficient for training a live system. Our attempt to improve on this is the subject of Section 8.3. We detail the overall temporal simulator architecture, the design of the individual agents that simulate dialogue, and an instantiation of a simple domain. To demonstrate the utility of the simulator, we implement multiple turn-taking polices and use the instantiation to compare these policies under conditions of varying reaction time and speech recognition accuracy. This section is based on Selfridge and Heeman [62].

The temporal simulator, inspired by the Open Agent Architecture [38], it is composed of a number of *agents*. We first describe the time keeping procedure and agent communication structure. We then describe a method of simulating dialogue using this setup.

8.2.1 Time and Communication

To provide a useful training environment, the simulator must realistically model, and run much faster than, 'real-time'. To do this, the simulator keeps an internal clock that advances to the next time slice after all agents have been run for the current time slice. This structure allows the simulator to run far faster than 'real-time' while supporting realistic communication. This framework is similar to the clock cycle described by Padilha et al. [42]. Here, time slices are modeled as 10 millisecond (ms) increments, as this is the time scale that speech recognizers run at.

Agents use messages to communicate. Messages have three components: the addressee, the content and a time stamp. Time stamps dictate when the content is to be processed and must always be for a future, not the current, time slice, as the alternative would imply instantaneous communication and overly complicate the software architecture. A central hub receives all messages and passes them to the intended recipient agent at the appropriate time. At every slice, each agent runs two procedures: one that retrieves messages and one that can send messages. If there are multiple messages intended for the same time slice, the agent completely processes one before moving to the next.

8.2.2 Timing of Spoken Interaction

The above describes a method to model real-time interaction while running much faster than real-time. Here, we take this structure and describe the specific temporal parameters that occur within the interaction.

At present, we focus on dyadic interaction and have three agents that are run in a strict order at every time slice: the User, which encompasses the input understanding, internal reasoning, and output processing of some user model; ISR, which, given the User's output, models recognition results and Voice Activity Detection; and the System, which consists of the Natural Language Understanding, Dialogue Management, and Natural Language Generation components of a Spoken Dialogue System. The behavior of all three agents relies on parameters (Table 8.1) that may either be set by hand or estimated from data. This setup can be used with any user, speech recognition, or dialogue manager model. In general, the User agent sends messages to the ISR agent that sends messages to the System agent. The System agent generally sends messages to both the User agent and the ISR agent. This is shown schematically in Figure 8.1.

Figure 8.1 also emphasizes that the dialogue manager and natural language understanding components should be applicable in real-world systems as well, and so should have direct portability from the simulator. In many respects, this comes down to implementation decisions and details which we argue are, in general, too case-based to give a full treatment here. Rather, we offer this as a principle that should be used to guide simulation construction.

User and System Design: Agent speech is governed by a number of timing parameters. The *Take-Turn* parameter specifies when the agent will begin speaking the selected utterance. The agent gets the first word of the utterance and, using the *Word Length* parameter, "begins" to speak by sending a speech event message. The agent outputs the word after the delay specified by the *Word Length* parameter, and the *Inter-Word Pause* parameter determines when the next word will begin. When the agent completes the utterance, it waits until a future time slice to start another (as governed by the *Inter-Utterance Pause* parameter). However, if the listening agent interrupts mid-utterance, the speaking agent stops speaking and will not complete the utterance.

ISR Design: The ISR agent works as both an Incremental Speech Recognizer and a VAD. We currently model uncertainty in recognition but not in the VAD, though it certainly would be a plausible and worthwhile addition. When the ISR agent receives a speech event from the User, the VAD *Speech Start* parameter models lag in speech detection and the $Speech\ End_{no\ word}$ parameter models situations where the user starts speaking but stops mid-word and produces an unrecognizable sound. When the word is received from the User two things happen: (1) the $Speech\ End_{word}$ parameter is used model the lag in detecting silence and (2) an incremental recognition result (or partial) is generated with timing according to the *Recognition Lag* parameter based on the probability that the word will be correctly recognized. This probability is then used as the basis for a confidence score that is packaged with the partial phrase result. The form of ISR we model recognizes words cumulatively (see Figure 8.2 for an example) though the confidence score, at present, is only for the newly recognized word. The recognizer will continue to output incremental recognitions from User words until the User stops speaking or the System sends a message to stop recognizing. One critical aspect of ISR which we are *not* modeling is partial instability, where partials are revised as recognition progresses. While revisions may certainly be modeled in the future, we chose not to for simplicity's sake.

Figure 8.1: Diagram of the Temporal Dialogue Simulator

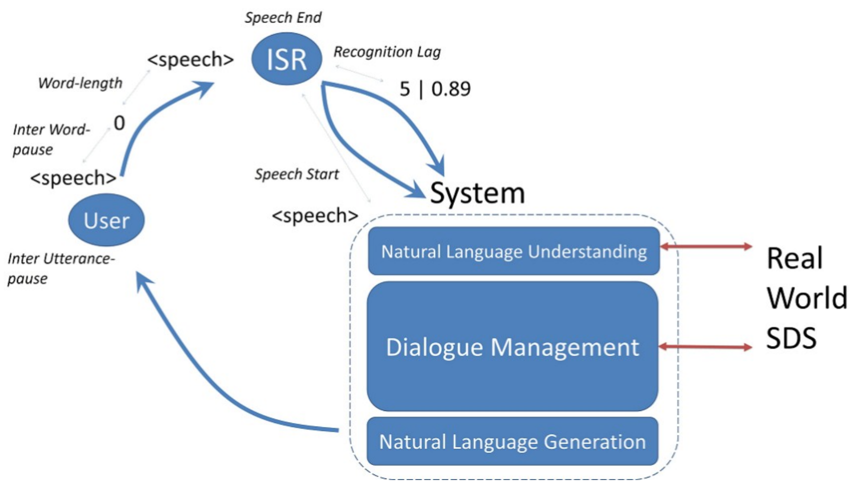


Table 8.1: Parameters and demonstration values (ms). Those with distributions are sampled for every instance.

Conversant Agents	
Inter-Word pause (Usr)	$\mu = 200, \sigma = 100$
Inter-Word pause (Sys)	100
Inter-Utt. pause	$\mu = 1000, \sigma = 500$
Word Length	400
Take-Turn (Usr)	500/200
Take-Turn (Sys)	750/100
ISR Agent	
Recog. Acc.	variable
Recog. Lag	300
VAD	
Speech Start	100
Speech End (word)	200
Speech End (no word)	600

We feel that, at present, the *Recognition Lag* parameter is sufficient to model the time for a partial to become stable.

8.2.3 Evaluating Turn-Taking Strategies with the Temporal Simulator

We now demonstrate the temporal simulator by showing that it can be used to evaluate different turn-taking strategies under conditions of varying ASR accuracy in a credit card domain similar to Skantze and Schlangen [64]. This is the first step before using it to train policies for use in a live dialogue system.

For this demonstration the two conversant agents, the System and User, are built according to the Information State Update approach [33], and perform an update for every message associated with the current time slice. In theory, any agent design can be used but here they are very much based on the the importance-driven approach though reinforcement learning is not used. Four types of rule sets are common across conversant agents: UNDERSTANDING rules, which update the IS using raw message content; DELIBERATION rules, which update the IS by comparing new information to old; UTTERANCE rules, which select the next utterance based on dialogue context; and TURN rules, which select the time to begin the new utterance by modifying the *Take Turn* parameter. This modification of the *Take Turn* parameter is similar to the "bid" used in Chapter 4 and Chapter 5. Rule sets are executed in this order with one exception. After the UNDERSTANDING rules, the System agent has ACCEPTANCE rules that use confidence scores to decide whether to understand the recognition or not.

Credit Card Domain: In the credit card domain the User says four utterances of four digits each. The System must implicitly confirm every number and if the System is correct, the User continues.[1] It can theoretically do this at any time: immediately after a word is recognized, after an utterance, or after multiple utterances. If the system says a wrong number the User interrupts the System with a "no" and begins the utterance again. The System has a Non-Understanding (NU) confidence score threshold set at 0.5. After an NU, the System will not understand any more words and will either confirm any digits recognized before the NU or, if there are no words to confirm, will say an NU utterance ("pardon?"). The User says "yes" to the final, correct confirmation. To maintain simplicity, "yes" and "no" are always accurate. If this were not the case, there would be a number of dialogues that were not successful. The User takes the turn in two ways. It either waits 500 ms after a System utterance to speak or interrupts 200 ms after the System confirms an misrecognized word, which is in line with human reaction time [20].

Turn-Taking Strategies: We implemented three different turn-taking strategies: two Fixed and one Context-based. Using a Fixed strategy, the System either uses a Slow policy, waiting 750 ms after no user speech is detected, or a Fast policy, waiting only 100 ms. The Fast reaction time results in the System interrupting the User during an utterance when the inter-word pause becomes longer than 200 ms. This is because the VAD *Speech End*$_{noword}$ parameter is 100 ms and the System is waiting for 100 ms of silence *after Speech End*. The Slow reaction time results in far less interruptions. The Context-based turn-taking strategy uses the recognition score to choose its turn-taking

[1]Unlike an explicit confirmation ("I heard five. Is that right?"), an implicit confirm ("Ok, five") does not necessitate a strict "yes" or "no" response.

behavior. The motivation is that one would want to confirm low-confidence recognitions sooner than those with high confidence.This designer specified importance, while having obvious utility, is inherently unprincipled. If any unconfirmed result has scores less than 0.8 then the System uses the Fast reaction time to try to confirm or reject as soon as possible. Alternatively, if the results all have high confidences, it can wait until a longer user pause (generally between utterances) by using the Slow reaction time. All parameter values are shown in Table 8.1.

Figure 8.2 shows a dialogue fragment of a System using the Context-based turn-taking policy. Numbers are used for the sake of brevity. The start of a box surrounding a word corresponds to when the Speech message was sent (from the User agent to the ISR agent) and the end of the box to when the word has been completed and recognition lag timer begins. The point of the ISR box refers to the time slice when the partial phrase result and score were sent to the System. After the third User word the System interrupts to confirm the utterance, since the confidence score of a previous word dropped below 0.8. Also, note that the User interrupts the System after it confirms a wrong number.

Results

We evaluated the three (two Fixed and one Context-based) turn-taking policies in two conditions of ASR accuracy: Low Error, where the probability of correctness was 95%; and High Error, where the probability of correctness was 75%. Using 1000 dialogues for each condition, we compared the mean dialogue time (top Figure 8.3) and the mean

Figure 8.2: Sample dialogue with timing information

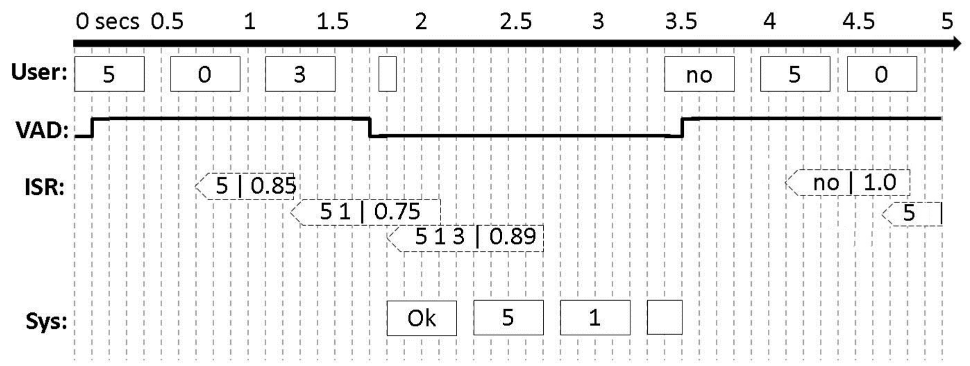
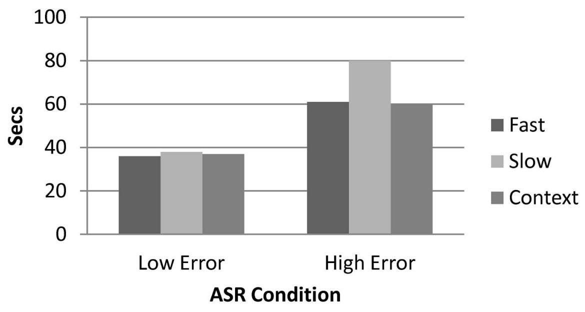
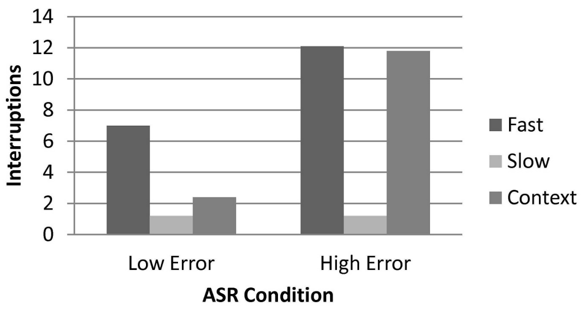


Figure 8.3: Mean Time and Interruption for different turn-taking polices and ASR accuracy conditions





number of interruptions per dialogue (bottom Figure 8.3). For dialogue time, we find that all turn-taking policies perform similarly in the Low Error condition. However, in the High Error condition the Slow reaction time performs much worse since it cannot address poor recognitions with the speed of the other two. For interruption, the Fast and Context-driven policies have *far* more than the Slow for the High Error condition. However, in the Low Error condition the Fast policy interrupts far more than the Context-driven. Given that natural behavior is one goal of turn-taking, interruption, while effective at handling High Error rates, should be minimized. The Context-based policy provides support for interruption when it is needed (High Error Condition) and reduces it when it is not (Low Error Condition). The other policies are either unable to interrupt at all (Slow), increasing the dialogue time, or due to a lack the flexibility (Fast), interrupt constantly.

8.2.4 Discussion and Conclusion

Here, we took the first steps towards a simulation approach that characterizes both the content of conversant speech as well as its timing. The temporal simulator models conversant utterances, ISR, and the VAD. The simulator runs quickly (100 times faster than real-time), and is simple and highly flexible. Using an example, we demonstrated that the simulator can help understand the ramifications of different turn-taking policies. This example highlighted both the temporal nature of turn-taking — interruptions, reaction time, recognition lag...etc. — and the content of utterances — speech recognition errors, confidence scores, and wrong confirmations. The success of the demonstration supports the use of the temporal simulator to train importance-driven spoken dialogue systems.

The obvious next step is to implement the temporal simulator with an operational dialogue system that can have real interaction with real people. Above, we mentioned that both the NLU and DM components should be portable to real-world systems and that this principle should guide simulation construction. This requires that the NLU/DM be "framework agnostic" and thus not care whether the recognition results (or VAD events) be from a simulated or real user. For now, we restrict our attention to speech recognition as we believe that any approach that works for recognition could also work for the VAD. One method to maintain input agnosticism is to build an interpreter (of sorts) that takes the string and the score, and packages it in the same format as a true recognition result that the NLU is expecting. However, this approach would only allow for modeling superficial elements of recognition and necessitate of modeling partials in some kind of generative fashion. While a generative model of this sort may be the right approach in principle, it could be somewhat tangential to our main goal: developing advanced turn-taking capabilities. Another approach is to simulate user speech instead and use an actual recognizer. This allows for very simple portability, as well as full and complete recognition results. It also enables real user audio to be used in the same framework as synthetic user audio, which is nice. This is the approach we take, and is the subject of next section.

8.3 Modeling Partials

Our general architecture to synthesize partial results is quite straight-forward. The synthetic recognizer has two components: (1) the synthesizer and (2) the recognizer. The synthesizer takes the user's reference as input (with possible prosodic markups), creates a synthetic speech audio file, and then modifies the audio file according to the design specification or training data. This audio file is then passed to the recognizer which outputs complete recognition results with time stamps. These time stamps can then be used by the temporal simulator to pass the result to the System at the appropriate time. This can all be done in the midst of simulated dialogue or off-line to generate a large population of complete recognitions that can be sampled during simulation. The latter may be the superior method since synthesis and recognition, while running much faster than real-time, does slow the simulation down considerably. This method of synthetic recognition results in partials that are complete recognition results, and not just incrementally produced surface text with a score as done above, and previously [37]. Note that by using a real recognizer, one can potentially get VAD events as well, since many recognizers already have this capability.

8.3.1 Modifying synthetic speech

Our approach is to add noise a random points to the synthetic speech, and the algorithm is quite straightforwards. Once the audio file is created by the speech synthesizer we use random crowd noise to create a file thats duration is equal to the duration of the synthesis file plus an some additional time (here we use 3 seconds). This file is then split into 13 different segments. The volume of each individual part is then modified based on the desired noise level, with the final part always being silent unless one wishes to model end-point failures. The variable segmental volume is then combined with the speech synthesis to create a new audio file to pass to the recognizer.

We model three different noise conditions: clean, low, and high. The condition indicate to the probability that there will be some noise in a segment. For clean, there will never be noise. For low and high, there is a 0.33 probability of 0.66 probability of noise, respectively.

For the noise, its volume is modeled as Gaussian function: N(0.2,0.1). A noise volume of 0.5 renders the speech indiscernible.

8.3.2 Creating synthetic partials

The modified audio file is then passed to a batch speech recognizer. Since the synthetic recognizer takes a reference string and produces partial results, it can easily be incorporated into our temporal simulation framework. Here, the user's temporal parameters would be used to construct the synthetic speech. In the case of system barge-in, what the user said so far can easily be calculated by finding the percentage of speech using the timing information as done in Chapter 7 for system resumption.

8.3.3 Cursory Evaluation of Synthetic Partials

We compare authentic and synthetic recognitions using stability, accuracy, and partials per word. The former two give an idea as to how characteristic the actual results are to authentic partials, and the latter shows how the production rates compare. Authentic recognitions came from 6773 recognitions in a Hands-Free Email task (described in Chapter 10). This task involved saying experimentally provided lines and so we automatically determined correctness by comparing the final recognition hypothesis with those lines; if it is present than the recognition is considered correct. For the synthetic recognitions, we used lines taken from the Hands Free Email Task to create 1 clean audio file and 3 audio files for the low and high noisy conditions each. Correctness was determined with by comparing the reference used to create the audio with the final recognition hypothesis. Partials Per Word (PPW) is only computed for correct recognitions.

Table 8.2 shows the mean and median PPW and stability and accuracy percentage of partials for correctly and incorrectly recognized utterances. We find that the PPW is quite similar between the synthetic and authentic recognitions, though authentic recognitions do have slightly more. When looking at Correct Clean synthetic speech we find that as a whole synthetic partials are less stable/accurate than authentic partials. Incorrect and clean synthetic speech (which only occurs when the reference is out of the language model) is somewhat more stable and a bit more accurate. When taken All together,

Table 8.2: Paritials Per Word (PPW), Stability, and Accuracy comparison between Authentic and Synthetic Partials for Correct(CFR) and Incorrect Final Recognitions (IFR)

	CFR		IFR		All	
	PPW Mn(Mdn)	Acc. %	Stab. %	Acc. %	Stab. %	Acc.
Auth	2.2 (3.0)	73	72	2	73	43
Clear	1.98 (2.0)	66	77	20	70	50
Low	2.1 (2.0)	42	56	14	52	24
High	2.0 (2.0)	50	45	10	46	14

Synthetic Clear recognitions are quite similar to the entire authentic recognition corpus. We note that this recognition corpus was collected using mechanical turk and the acoustic environment is probably quite good. We also find that as noise is added to the synthetic speech recognition performance degrades as expected with high noise generally yielding less accurate and stable partials than the low noise.

Taken as a whole, it seems that synthetic recognitions provide a relatively representative method of simulating incremental speech recognition results, particularly in terms of recognition rate. We feel this makes them appropriate to developing spoken dialogue systems in a new domain prior to the collection of authentic user data.

Chapter 9

Importance-Driven Turn-Taking in a live Hands-Free Email Task

9.1 Introduction

The goals of this chapter are two-fold: to describe a multi-agent architecture for building importance-driven systems and provide a live evaluation of the importance-driven approach. In sections 9.2 and 9.3 we describe the *Tempest* dialogue system that supports both utterance and turn-taking behavior. We detail how a multi-agent interaction management framework can be used with reinforcement learning to control and coordinate these various behaviors. In sections 9.3–9.7, we evaluate importance-driven turn-taking in a live hands-free email task. We find that under certain conditions the Importance-Driven (ID) approach is more efficient than the Keep-or-Release (KR) method, as its inherent flexibility accommodates and allows for both "expert" and "novice" users. We highlight the benefit of adapting to the user and underscore the difficulties of poorly-matched user simulations. We also evaluate the use of system-initiated mid-utterance interruptions, where the system starts to speak before recognition is complete. Here, the system explicitly attends to incremental input that it believes to being mis-recognized and we find that, in these situations, dialogues with mid-utterance interruption are significantly shorter than those without. Note, this application of ISR is somewhat different than the interruption described in Chapter 7, where the goal was to resume speaking during poorly formed input, disregarding the input entirely.

9.2 The Tempest Dialogue System

We now describe a dialogue system which we refer to as *Tempest.* The Tempest dialogue system is a multi-agent interaction framework that, leveraging both information state and reinforcement learning techniques, supports both utterance and turn-taking decisions. Here, "agent" refers to different *decision* agents. As in traditional AI, where multiple agents can coordinate their actions to clean a room, we use multiple agents to coordinate SDS behavior using multi-agent learning. We stress that this approach is specifically meant to partition the state space and enable distinct reinforcement learning agents for each behavior. While compatible with, it is *not* an Open Agent Architecture [38].

In general, Tempest will consume raw recognition results from some speech recognizer and this information will be processed by its understanding and deliberation rule sets to update its information state. Where a standard dialogue system's action will be some speech act, Tempest is designed to coordinate multiple actions types such as utterance and onset. Each high-level action type is called a *decision*; e.g. the utterance decision could be the request action. While we are primarily concerned with utterance and turn-taking decisions, the structure could easily be extended to multi-modal decisions. Tempest uses its decision agents to produce at least an utterance decision and turn-taking decision. The turn-taking decision determines *when* the system should produce the utterance decision speech act. These are determined by applying multi-agent reinforcement learning, which we discuss in Section 9.3. We first describe the overall Tempest functionality and then describe the reinforcement learning procedure used to learn a policy using this functionality.

9.2.1 Dialogue Management

Tempest consumes input using understanding and deliberation rule sets and the decision agents (implemented as rule sets) that apply their respective policies to select the optimal action for each decision. Here, we limit ourselves to just two decisions: utterance and turn-taking. However, other decisions such as volume or prosodic emphasis or multi-modal display can easily be incorporated into this structure to modify the utterance and are within the theoretical confines of the Importance-Driven approach.

Rule Sets: As described in Chapter 2, information state update rule sets are handcrafted logic rules that typically have two steps. First, a precondition, represented as a boolean expression, is evaluated to determine whether the rule is valid. If the precondition evaluates to true, then the the rule is "fired" and the effects component updates the information state. Currently, Tempest has four rule sets: understanding, that updates the information state based on the applying Natural Language Understanding to input; deliberation, that updates the information state based on the *current* information state; utterance, that produces a list of potential speech acts; and turn-taking, that produces a list of potential onsets. For the utterance and turn-taking rule sets, preconditions are used to constrain the search space so learning agents do not need to learn polices such as "Don't exit the conversation immediately." [25]. The actual utterance or turn-taking action is then chosen by the decision agent.

The **utterance decision agent** selects which of the potential speech acts to use for a given dialogue context. Once selected, the utterance action has two stages. The first stage updates the IS at the time of the action's selection, and is commonly used to communicate that the system has the intention of speaking a particular utterance. The second stage is activated after the utterance has been said. This allows for the system to have so-called "output awareness", in that its second stage updates the state based on what was actually said and not just the intended speech act. This is applicable in two different scenarios. First, since Tempest engages in turn-bidding the utterance may never actually be said. Second, as user barge-in may truncate the utterance, the second stage synchronizes the Tempest IS with the environment. While some may argue that output awareness should be handled by understanding rules, we caution that since this second component is generally specific to each utterance it increases the complexity of the understanding rule set. Moreover, structuring the speech act as such increases the modularity and mobility of the utterance action set.

The **turn-taking decision agent** controls when the system does and does not speak. We define 3 different interaction contexts, from which different system turn-taking actions will be applicable: *onset*, when both the system and user are silent; *continue/pause*, when the system is speaking and the user is speaking; and *react/ignore*, when the system is silent and the user is speaking. We note that there is another dimension, where the recognizer is active but the user is *not* speaking but , while we devoted considerable attention to this scenario in Chapter 7, we feel that it can overcomplicate the interaction and should be treated as a special case. As such we disregard it here.

The onset case is the general interaction context where an utterance has just finished and enables the system to choose the utterance's onset based on its importance and so bid for the turn. This is the turn-bidding behavior described at length in Chapter 4. The continue/pause case is the user barge-in context where the system needs to determine how to respond to the environment initiated overlapping input. The react/ignore case applies to active incremental speech recognition during system silence where "react" means to start speaking and "ignore" to remain silent. The PBR model described and evaluated in Chapter 7 is a hand-crafted policy of the latter two cases.

State Management: The IIM updates the Information State and requests new decisions from the DM at certain decision points. These decision points are dependent on whether Tempest is operating incrementally or not. In the general case, where the system is not reacting to incremental results, the state is updated when user speech starts, after every final speech recognition result, and after every system utterance.

When operating incrementally, Tempest employs a similar Incremental Interaction Manager (IIM) to the one presented in 6. The Tempest IIM has two primary roles: state management, for handling incremental results; and state updating, for applying the interaction policy. Since the IIM sits between the "production" side (speech recognition and synthesis) and the dialogue manager, it allows the interaction manager to be agnostic to its environment and allow DM decisions to be learnt and executed irrespective of a live or simulated environment. While capable of handling revisions by employing techniques described in Chapter 6, we chose to not enable this feature. Thus, the IIM only manages two version of the state (or DM, depending on implemention): the initial state and the new state. For each partial, the system copies the initial state and creates a new state. If that state produces a React action, the system produces the next utterance and overwrites the (now historic) initial state with the new state.

In summary, Tempest allows for pauses and breaks between system utterances, the ability to explicit reason on barge-in, and the ability to react to or ignore incremental results. However, since it is event driven, Tempest does not need to explicitly incorporate time into its information state.

9.3 Learning Utterance and Turn-Taking Policies

We now describe the reinforcement learning approach for finding optimal utterance and turn-taking policies. An utterance policy is a mapping from context to speech act, and a turn-taking policy is a mapping from context to turn-taking action.

Here, we take a multi-agent view to learning these decisions, using a separate MDP for utterance and turn-taking decision instead of a joint MDP as done in 4 and 5. There, we used one MDP, forcing the (joint) state space to be much larger and minimizing generalization capabilities. The multi-agent approach is appealing since it supports a specific set of state features and learning procedure for each decision type and so drastically reduces training time and increases generalizability. One issue with multi-agent systems is that the non-stationarity of the each agent can lead to training difficulties unless specifically accounted for. Here, we enforce coordination by using a cascade approach [8]: Each decision is performed in the same order at every decision point and the results of that decision are a state feature for the next decision. Since the outcomes of previous decisions constrain the potential actions of the next, the learning structure itself is semi-hierarchical though rewards are not propagated between decision state-action pairs as done in Chapters 4 and 5, since each decision type has its own MDP. After a dialogue, each decision agent receives the same fixed reward that represents a dialogue time that will never be reached during training. This unattainable reward is then modified by decision agent specific reward functions.

9.3.1 Utterance Decisions

In learning utterance decisions, the learner should only reinforce the subset of utterances that *were* spoken, as opposed to those that were only *intended* to be spoken. The second stage of the utterance action, which is only active after the utterance is used, facilitates this conditional reinforcement. Indeed, the Q-Score is only updated if a sufficient percentage of the utterance is said, the percentage being defined by the second stage. For instance, if user barge-in occurs within 300 milliseconds of starting the prompt than one can assume it is an "initiative conflict" situation and so the user is *not* reacting to the prompt. In this case, the Q-Score would not be updated. In the other case, where there is no temporal evidence of initiative conflicts, the Q-Score *would* be updated and (currently) the system assumes that the utterance has been grounded and understood.

Utterance Q-scores are updated using a Q-Learning variant, specifically the reward function shown in Equation 9.1. The reward at each state is the combination of the (negative) amount of utterance time and the discounted estimate of the next on-policy Q-score ($Q(s', \pi(s'))$). Our variant of Q-Learning modifies the discount by τ, which represents the number of utterance decisions that were *not* spoken since the last system utterance and was inspired by hierarchical RL approaches for dialogue systems that use semi-Markov Decision Processes [11]. Our experimentation has found it critical to learning a proper utterance policy. This updating scheme seeks to limit the amount of actual speech the system says. The α value used here is proportional to the number of times the state–action pair has been seen: $1/\sqrt(SA_{count})$

$$Q(s,a) = (-1 \cdot utterancetime) + \alpha \cdot \gamma^{\tau} Q(s', \pi(s')) \tag{9.1}$$

9.3.2 Turn-Taking Decisions

Unlike utterance decision, turn decision Q-Scores are always updated using Q-Learning since turn-taking decisions, which determine onset, always influence the dialogue outcome. The turn-taking Q-Scores (Equation 9.2) are updated based on the time between when the decision was made and when the next utterance (either user or system) occurs.

This updating scheme directs the system to always choose higher (faster) bids, as these have less time to speech. However, it also reinforces the use of lower bids if it expects the user to say something that will advance the dialogue. The alpha value used here is the same as for the utterance decisions. Here, as we now take the bid time itself into account, we improve over our previous work which did not [61] .

$$Q(s,a) = (-1 \cdot \mathit{timetospeech}) + \alpha \cdot \gamma \cdot Q(s', \pi(s')) \tag{9.2}$$

9.4 Importance-Driven Turn-Taking in a Live Domain

Recall from Chapter 3 that the importance-driven approach has two fundamental components: a variable strength turn-taking signal that is determined by utterance importance and the use of reinforcement learning to estimate this importance. We evaluate the efficacy of this turn-taking model in a hands-free email task with live users. In this section, we describe the domain, the system architecture, the user simulation, and the reinforcement learning training procedure. The work in this chapter builds on the work of Chapters 4 and 5, as we (a) use a more sophisticated RL model that uses separate agents to learn each decision, (b) describe a model with fuller range of turn-taking actions, including user barge-ins, system interruptions, and turn fights, (c) incorporate this work in a full system, and thus deal with such issues as speech recognition, and (d) conduct experiements with actual users.

9.4.1 Hands-Free Email Domain Description

A hands-free email domain is used for evaluation purposes. We chose the domain because authoring an email with just voice is quite simple, accessible, and is currently a task being used by the public. The goal of the system is to determine the recipient, subject, and body of the an email that the user wants to send. The user's role is to supply that information. Since the goal of this work is to evaluate turn-taking strategies rather than dialogue systems or speech recognition, the specific email was held constant. We further exploited the email's static nature for confidence scoring and phrase segmentation. The email is shown in Table 9.1. The system can query ("what is the body?"), confirm ("i think the subject is read report."), confirm prompt ("Is this correct?"), ask if there is any more body (" Is there any more body?"), and acknowledge (User: "the recipient is steve smith", System: "ok."). The system can also reject the speech recognition hypothesis ("I'm sorry, what was that?"), though we chose to only activate thie rejection behavior for

Table 9.1: Hands-Free Email

Recipient	steve smith
Subject	read report
Body	i read through the report and it looks good. i have a few questions so lets chat about it next week. lets plan to meet first thing monday. have a good weekend.

Table 9.2: Hands-Free Email

Speaker	Utterance
System	who is the recipient?
User	steve smith
System	what is the subject?
User	read report
System	i think the subject is red rabbit
System	is this correct?
User	no
System	what is the subject?
User	read report
System	what is the body?
User	i read through the report and it looks good
System	i thought you said i read through the report and it looks good
User	yes
System	ok
System	is there any more body?
User	yes
System	what is the body?
User	i have a few questions so lets talk about it next week.
...	...

the incremental case. The user can give information (“subject is read report”), respond to a system query (System: “What is the subject?”, User:“read report”), accept (System’ “I thought you said Steve Smith. Is this correct?”, User: “yes”), and reject (System’ “I thought you said Frank.”, User: “no”). After the system knows the recepient, subject, and user has indicated there is no more body the system ends the interaction. An example dialogue is provided in Table 9.2.

9.4.2 The Hands-Free Email Dialogue System Architecture

Here, we describe the domain *dependent* components of the hands-free email system built using the Tempest dialogue architecture. As we mentioned in the previous section, the goal of the system is to gather recipient, subject, and body information. As the goal of the work is to focus on turn-taking and initiative, we simulated state-of-the-art confidence scoring and parsing components by leveraging the tight-constraints of our experimental domain.

Input and Output Processing: Speech recognition is performed using the AT&T WATSON^{SM} speech recognizer [23] with the beam width set to run faster than real time. The language model is a tri-gram SLM that was trained on 80k utterances taken from the EnronSent Corpus [68] plus the subject, recipient, and body of the email shown above. No adjustments were made to favor the HFE utterances. The body was included as four separate utterances. In practice, this means that multiple lines of body have not been explicitly seen by the recognition model and are likely to result in a recognition error. The recognizer uses a generic English telephone acoustic model and a text-to-speech based pronunciation model. Incremental Speech Recognition was performed using Lattice-Aware Incremental Speech Recognition set to check for new partials every 20 frames (roughly 200 ms of clock time), as described in Chapter 6. System output was entirely synthetic, and for which we used we used AT&T Natural Voices speech synthesis.

As we mentioned previously, we simulate two state-of-the-art system components: *oracle confidence* and *oracle parsing*. *Oracle confidence* uses the pre-defined task utterances to give the recognition hypothesis one of three possible "scores": If the hypothesis is an exact match to one of the email task utterances it recieves an score of 1.0, if the hypothesis is an exact prefix (to handle the incremental case) it recieves a 0.5, and if it is a no match than it recieves a score of 0. As the goal of this work is not to evaluate confidence scoring techniques we see *oracle confidence* as a way to evaluate the best case scenario for detecting recognition errors. Along those same lines, the system has knowledge of certain words that end phrases ("good", "week", "monday", and "weekend"), providing *oracle parsing* and that it is only expecting 4 body utterances. It uses this knowledge to partition the

Table 9.3: Speech Acts and Phrase Templates

intent	phrases
User	
accept	yes, yeah, yup, correct
reject	no, nope, wrong, incorrect
giveSubject	subject [is] <subjec>
giveRecipient	recipient [is] <recipient>
giveBody	body [is] <body> backchannel
uh huh	
System	
Explicit Confirm Recipient	confirming recipient
Explicit Confirm Subject	confirming subject
Explicit Confirm Body	confirming body
moreBody	any more body
querySubject	what is the subject
queryRecipient	who is the recipient
queryBody	what is the body
confirm-prompt	is this correct, am I correct so far

input so that it confirms only one phrase at a time.

Natural Language Understanding is applied very simply in the current work. The goal is determine an interpretation for each utterance. The interpretation has two components: intent, which specifies the type of information being spoken (e.g. giveSubject); and the value, which specifies the actual information (e.g. “steve smith”):

The interpretation is determined by first associating a number of prototypical phrase templates with intents. Table 9.3 provides both a list of intents and the phrases. The bracketed words are optional and the words surrounded by angle brackets indicates the value position. We include both the System and User as this NLU approach is used in both simulation and live experimentation.

These phrase templates are then matched against the utterance, and the intepretation is extracted. For example, the utterance “subject steve smith” matches the intent giveSubject with a value of “steve smith” as the actual subject.

Deliberation rules manage both a QUD and the three slots. The QUD handles two types of questions, which only orginiate from the System: queries regarding collecting email values (e.g. querySubject, “what is the subject?”) and confirmations regarding already collected values (e.g.confirmSubject, “I heard Steve Smith. Is that correct?”). The QUD behaves as a queue, associating potential responses to the QUD items in chronological order. For example, if the System says ‘’I head Steve Smith. Is this correct? I heard ready. Is that correct?” and the User responds “Yes. No” than the System will believe “Steave Smith” is correct and “ready” is not.

While every line of the body must be confirmed, the recipient and subject slots do not need to be. The *oracle confidence* score and the number of slot attempts determinees whether the system will confirm a recipient/subject hypothesis. If the hypothesis has a score of 1.0 or the System has made two previous attempts to fill the slot than the slot if filled without confirmatin and System moves on. For body, the system must confirm every utterance irrespective of the oracle confidence score. Since the system only confirms one body phrase at a time the confirmed body list should be 4 items long.

The system will exit on two conditions. The user responds with a negative response to the system’s “any more body?” utternace and the confirmed body list has at least 4 elements or the user rejects the 5th contnet confirmation attempt.

Turn-Taking decisions are constrained based on what turn-taking framework is being used, with three possible actions during user and system silence: bid high (speak in 0.1 seconds), bid mid (speak in 2 seconds, only available to Importance-Driven systems), and bid low (speak in 10 seconds). These times were chosen based on informal experimentation with the system

Obviously, more fine grained onsets are valuable (as explored in Chapter 5) but we felt that these three would be sufficient. When the user is speaking the system decides whether to react or ignore the input and it does not handle revisions, meaning that once it decides to react it will disrgard all subsequent recognitions from the particular recognition stream. Barge-in is disabled for all system prompts.

Table 9.4: RL State Features

State Variable	Type	Utterance State	Turn State
Potential Recipient	binary	x	x
Potential Subject	binary	x	x
$\|PotentialBody\|$	integer	x	x
Recipient	binary	x	x
Subject	binary	x	x
$\|Body\|$	integer	x	x
Quantized(QUD)	triary	x	x
Confidence of 1st Potential Body element	triary	x	
Confidence of 2nd Potential Body element	triary	x	
Number of elements in Potential Body List	binary	x	x
last system speech act	string	x	
last user speech act	string	x	
last system turn-taking action	triary	x	
last speaker	binary	x	
current prompt	string		x
$\|Potentialontent\|$¿0	binary		x
$\|PotentialBody\|$¿1	binary		x
last system speech act is acknowledge	binary		x
last system speech act is confirm-prompt	binary		x[1]

Reinforcement Learning is applied to both utterance and turn-taking decisions. These decisions apply different MDPs and the state variables for each are domain dependent. They are shown in Table 9.4.

9.4.3 Simulated User

Learning an interaction policy, which is the combination of the utterance and turn-taking policy, can take a large number of iterations and so it is not practical to train an importance-driven system with live users. Hence, we apply a user simulation for training. The user simulation incrementally understands the system speech and is capable of producing barge-in utterances. The user can be interrupted by the system and the recognition is stopped short (with respect to the user's intended utterance) as it would in a live system.

The user simulation also relies on the Tempest architecture to make utterance and turn-taking decisions, though without reinforcement learning. Instead, the user's behavior

Table 9.5: User Attributes. Each one is either activated or not. Only the first three attributes were varied in the experiments to be described.

Attribute	Description
Acknowledge Reaction	Provide the next line of body if the system acknowledges that previous line rather than wait until prompted
Body Confirm Reaction	Respond immediately to a body confirmation without the system having to say “is this correct?”
Multi-Body Utt	Group body utterances together into a single utterance
Backchannel Understand	The User assumes any information prior to the backchannel does not need to be confirmed
System Initiative	Do not provide information unless prompted by the system
Proactivity	Requires a system acknowledgement before continuing on
Non-Confirm Reaction	Respond to a non-body confirmation without the system having to say “is this correct?”
More Body Reaction	Provide the next line of body when the system says “any more body”, rather than just say “yes”
True Rejection	Only reject body confirmations that are actually incorrect
No Barge-in Accept	Do not barge-in during system confirm prompts to confirm the confirmation
No Barge-in Reject	Do not barge-in during system confirm prompts to reject the confirmation
No Restart on Reject	Do not start the entire body slot over if the system gets anything wrong irrespective of previous confirmations.

is dependent on a set of attributes. These attributes are applied in the IS preconditions and, while not exhaustive, can model a wide variety of user behavior. Each combination of attributes specifies a particular user type with a static behavioral pattern. User variability can be introduced by varying the attribute combination. The full attribute list is provided in Table 9.5. The attributes generally vary the initiative pattern of the users between being an “expert” or being an “novice”. In short, an “expert” user does not require explicit system prompts whereas a novice does.

The user is capable of taking three different turn-taking actions: speak now, wait 2 seconds and then speak, or be silent. While variability can easily be applied to the first two turn-taking actions by applying some random noise to when the user actually speaks (as done in Chapters 4 and 5), it does nothing but increase training time for the system. We keep them constant to reduce training time.

We constructed this user simulation using the temporal simulation framework described in Chapter 8. One primary differenc is the use of synthetic speech recognition results as described in Section 8.3. In this sense, the framework described in Chapter 8 is a precursor to the work described here. The user simulation relies on speech synthesis to provide realistic input to the system. For each possible user utterance, an audio file is created. This audio file is then used to produce synthetic speech recognition results as described in Section 8.3. If noise is not added, we found that every possible user utterance with the exception of “weekend” is recognized correctly and a rule can be applied to restrict the use of this singleton. Synthetic speech is also used to determine the timing of the users speech. If the system interrupts the user, this timing is used to find out how much user speech was said so far and the current recognition stream is clipped so that the next partial recognition becomes the final recognition result. Here, we use the actual ISR result to support the clipping of the recognition stream which is somewhat different than our approach in Chapter 8.

9.5 Characterizing the Interaction Policies

This section describes and compares the interaction policies learnt when the environment does not explicitly introduce speech recognition erros. We feel this provides the best insight into the policies prior to our description of the experiments.

9.5.1 Interaction Policy Training

The Importance-Driven systems are trained using reinforcement learning methods described in Section 9.3. The **KR System** uses a hand-crafted interaction policy. We also trained a KR interaction policy and found it it was identical to our hand-crafted one. We decided to forgo using the trained policy to eliminate any state space issues that would cause the KR system to perform superficially poorly.

The importance-driven interaction policies were trained by varying the first three user attributes shown in Table 9.5, yielding 8 different users, and the probability of recognizing noisy audio. These attributes were chosen based on pre-testing with small number of live users (described in Section 9.7), and represent three hypothesized differences between the user populations. This pre-testing is more fully described in Section 9.7. All other attributes are present. Each user is coded by the presence or absence of three different attributes (respectively): acknowledge reaction, confirm body reaction, and multi-body turn. Note that the last attribute, multi-body turn, will have speech recognition errors at the beginning of body delivery and is the more "difficult" attribute. For simplicty we refer to any user that has either of the first two attributes as an "expert" and any user that lacks both attributes as a "novice". The test set consists of one dialogue for each user type (here 8), and the training Epoch consists of each user type being presented 100 times to the system.

To speed training, we start with the first user type and train until convergence. We then serially introduce the next two user types and train until convergence for each one. The policy for the current state space is then "frozen" (the Q-Scores cannot be modified), and speech recognition errors are introduced using the synthetic speech recognition approach outlined in Chapter 8; each user utterance is randomly determined to be either clean, low, or high. The introduction of recognition errors increases the state space dramactically, while keeping the "clean" policy intact and unaltered. The serial training of policies circumvents the possibility that Markov violations in the state space would degrade the "clean" policies.

9.5.2 The Keep-and-Release System

The KR system first fills the recipient and subject slots and then moves onto body. It will ask for the body and when wait until the user has stopped speaking. If there is a phrase boundary within the Potential Body List then it will confirm up to that phrase boundary, use the confirm-prompt utterance, and wait for the user to respond. If the user responds in the affirmative then the system will either (1) confirm the next part of the potential body up to the next phrase boundary using oracle parsing if the potential body list is non-empty or (2) use the anymore-body utterance and wait for a response. If the user says "yes", then it will ask for the body. If the user says "no", it will end the interaction. Table 9.6 shows a fragment from the body sub-task for the KR system.

Table 9.6: KR Body sub-task Example

Speaker	Utterance
User	i have a few questions so lets talk about it next week
KR	i thought you said i have a few questions so lets talk about it next week
KR	is this correct?
User	yes
KR	is there any more body?
User	yes
KR	what is the body?
User	lets plan to meet first thing Monday have a good weekend
KR	I thought you said lets plan to meet first thing Monday
KR	Am I correct so far?
User	yes
...	...

9.5.3 The Importance-Driven System

There are two **importance-driven systems**: ID1 and ID2. They differ only on their confirmation capabilities. While ID1 must always say "I thought you said" confirmation lead-in when confirming body hypotheses, ID2 is able to learn a shortened form: if the user accepts a confirmation without requring a confirm-prompt than the system has the option to not use the confirmation lead-in for the next item on the Potential Body List. This behavior is obviously only applicable when there are multiple entries into the Potential Body List. As Table 9.7 illustrates, this single difference causes characteristics and performance of the trained policies to differ substantially. The motivation for separating ID1 and ID2 is that ID1 is quite similar to KR whereas ID2 exhibits greater distance. We were curious as to whether only a small shift is turn-taking was required for performance improvement (ID1) or a much larger adjustment was necessary (ID2). Note we also trained a KR system that had the same confirmational flexibility as ID2; the resulting interaction policy did not deviate from the original KR. This underscores the primary difference between KR and ID approaches. KR mirrors the monolithic system utterances that most dialogue system use today, whereas ID allows the systems to have a larger set of smaller utterance it can use to produce more dynamic and flexible behaviors. ID1's policy is quite close to KR with two exceptions, both related to the first two attributes. One

Table 9.7: Two Importance-Driven Systems. The first fragment is ID1 and the second is ID2

Speaker	Utterance
ID1	I thought you said, I read through the report and it looks good.
User	yes
ID1	I thought you said, I have a few questions so lets talk about it next week.
User	yes
ID2	I thought you said, I read through the report and it looks good.
User	yes
ID2	I have a few questions so lets talk about it next week.
User	yes

difference is that ID1 learns to confirm after every utterance and use a middle bid before saying the confirm-prompt utterance. This allows Expert users to provide a confirmation response without the system explicitly prompting them while being able to accomodate Novice users that require the confirm-prompt utterance. Another difference is that ID1 also learns to acknowledge with a high bid after the confirmation response and then use a middle bid before asking for any more body. This behavior gives the user the opporunity to take the initiative and so increase the efficiency of the interaction. An example of ID1 with three different users is shown in Table 9.8.

ID2 learns a much different policy. Instead of confirming after every utterance, ID2 attempts to accumulate potential body without confirming. It can then confirm all of them at once, and only use one confirmation "lead in" phrase ("I thought you said..."). It does this accumulation by acknowledging with a high bid after every successful body utterance and then prompts the user for more body using a middle bid rather than a high bid. Table 9.9 provides an example.

We point out that both importance-driven systems adapt to the user behavior, as the reinforcement learning state implicitly models the user. For ID1, if the initial line of body indicates that the user is a novice (the user does not offer the confirmation response without the confirm-prompt), then the system will upgrade any subsequent middle bids to high. This reduces the amount of silence between system utterances. As ID2 does not attempt to confirm after every line of body, it learns a different form of adaptation. If

Table 9.8: ID1 Example Dialogue

Speaker	Utterance
ID1	I thought you said, I read through the report and it looks good.
ID1	am I correct so far?
User 000	yes
ID1	ok
ID1	Is there any more body?
User 000	yes
ID1	What is the body?
ID1	I thought you said, I read through the report and it looks good.
User 010	yes
ID1	ok
ID1	Is there any more body?
User 010	yes
ID1	What is the body?
ID1	I thought you said, I read through the report and it looks good.
User 110	yes
ID1	ok
User 110	i have a few questions so lets talk about it next week

the first acknowledgement does not produce a user action than ID2 no longer uses the acknowledge utterance and goes directly to the anymore body utterance. This is shown for user 010 in Table 9.9. ID2 will also exhibit the same bid adaptation that ID1 does, in terms of upgrading middle bids to high. We revisit adaptation when we evaluate the system with live users.

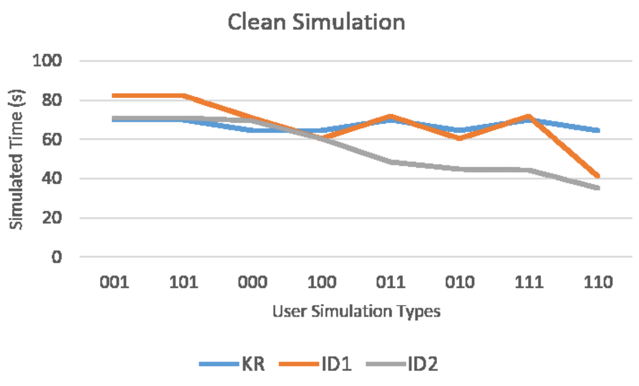
Mid-Utterance Interruption relies on the *oracle confidence* score and the *type* of partial being returned. We built a handcrafted interrruption policy that is very simple and only applies to the body sub-task, as this is the only sub-task with longer utterances. Here, the goal of interruption is minimize the amount of user speech that may be misrecognized as it increases the dialogue time. The interruption policy directs the system to react if an *immortal* partial has an *oracle confidence* score of 0 and will use one of two behaviors depending on the context. In one case, if phrase boundaries are present in the potential body list, the System will interrupt to confirm the elements on the the list including the one it deems misrecognized. However, if there are no phrase boundaries present, the system will interrupt to reject the recognition saying that it cannot understand the speech.

Table 9.9: ID2 Example. Here, we show just two lines of body though in reality the system will go through all 4

Speaker	Utterance
User 000	i read through the report and it looks good
ID2	ok
ID2	Is there any more body?
User 000	yes
ID2	What is the body?
User 000	i have a few questions so lets chat about it next week
ID2	i thought you said i read through the report and it looks good
ID2	am I correct so far?
User 000	yes
ID2	i thought you said I have a few questions so lets chat about it next week
ID2	is this correct?
User 000	yes
User 010	i read through the report and it looks good
ID2	ok
ID2	Is there any more body?
User 010	yes
ID2	What is the body?
User 010	i have a few questions so lets talk about it next week
ID2	Is there any more body?
User 010	no
ID2	i thought you said i read through the report and it looks good
User 000	yes
ID2	I have a few questions so lets chat about it next week
User 010	yes
User 110	i read through the report and it looks good
ID2	ok
User 110	i have a few questions so lets chat about it next week
ID2	i thought you said i read through the report and it looks good
User 000	yes
ID2	I have a few questions so lets chat about it next week
User 010	yes

ID2 learns the same interruption turn-taking policy as this simlpe handcrafted one.

Figure 9.1: Clean Simulation Results



9.6 Evaluation with the Simulated Users

9.6.1 Turn-Bidding

Here, we present the results of comparing KR, ID1, and ID2 with simulated users without system interruption.

Clean Environment: We compared the systems in simulation for each of the 8 user types described in Section 9.4.3 with no additive noise, so recognition error is non-existent except for multi-utterance body. Since each user simulation behaves exactly the same, we can calculate the exact time it takes each agent to interact with each user type. This time is shown in Figure 9.1.

We find that there are substantial differences between all three systems, though most notably between ID2 and the other two. While the performance of the KR system remains relatively steady, ID2 can exploit the differences between the user types. ID1 only performs better for experts that have both indicative attributes (code = 110) and so require very little prompting, as any turn-bidding gains are erased by stretches of silence from more novice-like users (code = 100 or 010).

Noisy Environment: Here, we compare the effect of frequent speech recognitions errors on the three systems. To compute the expected performance of the systems, we ran each of the 8 user types 100 times. For each user utterance, we either added no noise (50% of the time), low noise (25% of the time), or high noise (50% of the time

Figure 9.2: Noisy Simulation Results

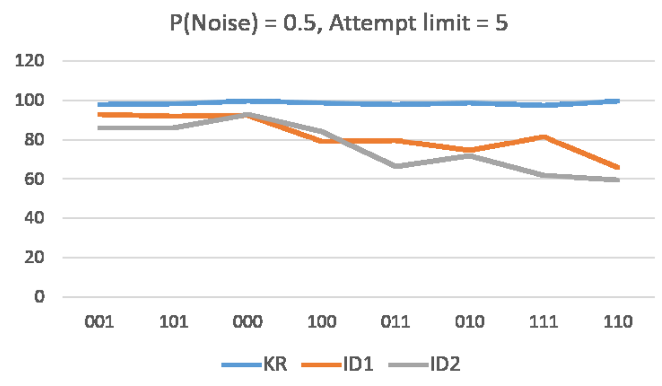


Table 9.10: ID2 dialogue fragement from Noisy Environment. Notice that instead of acknowledging the User's last body utterance to get more body before confirming (as in the Clean Environment policy), ID2 confirms immediately.

User 010	i read through the report and it looks good
ID2	ok.
User 010	i have a few questions so lets chat about it next week
ID2	i thought you said i read through the report and it looks good
User 010	yes
ID2	i have so little about it next week
User 010	no
ID2	What is the body?
User 010	i have a few questions so lets chat about it next week
ID2	i thought you said i have a few questions so lets chat about it next week

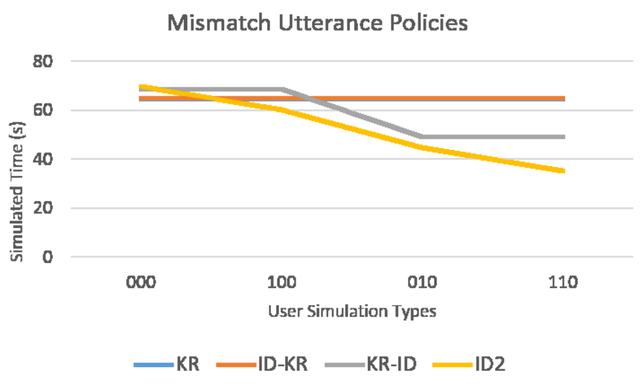
We find that, as expected, the differences in the noisy environment mirror those found in the clean environment. One exception is that the performance gap is greater between ID1 and KR than in the clean environment. This is due to the greater number of confirmations and so greater opportunity for user initiative gains. We note that after encountering recognition errors ID2 changes its usual body collection behavior and starts to confirm every line. This is based on learning an expectation that it will encounter difficulties later and will no longer get any gain from grouping confirmation phrases. Table 9.10 provides an example of this.

Policy Mismatches: One question that arises is the optimality of the utterance policy with the change of turn-taking behavior. In other words, is the utterance optimal policy maintained with more degrees of turn-taking freedom, as this determines whether importance-driven turn-taking can be applied existing systems while preserving the optimal behavior of this existing system. To explore this we took the utterance policy components of the KR and ID2 interaction policies and trained a new turn-taking policy to create two mismatched interaction policies: the KR utterance policy is frozen and used while training a new importance-driven turn-taking policy (KR-ID), and an importance-driven utterance policy is used while a KR turn-taking policy is learnt (ID-KR). Note that since the utterance states contain turn-taking variables a certain amount of exploration for the utterance policy will occur in the KR-ID setup. While this will not impact existing utterance state-action pairs it does mean that the utterance policy may change.

By comparing the original interaction policy with the mismatched one we can determine whether the utterance policy remains optimal. If the mismatched policy performs worse, it implies that utterance policy optimality is dependent on being learnt in coordination with the turn-taking framework and that interaction policy components are not necessarily interchangable.

The results are shown in Figure 9.3. We find that the mismatched interaction policy ID-KR is virtually identical to the completely KR approach: the importance-driven utterance policy is still optimal when used with a KR framework. However, we also find that the KR-ID policy is substantially different from the ID2 policy: the KR utterance policy is *not* optimal when used in importance-driven framework. Looking at the policy we find that there are two main differences. First, since KR never acknowledges, KR-ID system is forced to confirm after every line of body. This constraint forces the system to ask for "any more body" after the user confirmation responses. Furthermore, since the policy is somewhat fixed the KR-ID system cannot adapt to users since ID2 adapts both its turn-taking *and* utterance decisions. While these are minor difference, it provides solid evidence that utterance and turn-taking policies must be trained simultaneously if optimal interaction behavior is going to be achieved. In order words, utterance optimality is not preserved between turn-taking approaches. So, in order to assure and achieve the highest

Figure 9.3: Mismatches between initial utterance and turn-taking policies



possible performance utterance policies must be determined with the turn-taking policy; i.e. the interaction policy is a single entity and should be treated as such.

9.6.2 Mid-Utterance Evaluation

Here, we evaluate the effect of allowing system-initiated mid-utterance interruption using a hand-crafted interruption policy with both KR and ID2 systems. This policy is only active during speech recognition errors since the goal is to minimize user speech that is being misrecognized and so here we only show simulation data for the noisy condition. Figure 9.4 shows the difference in mean task time between the interruption systems and systems that do not interrupt. We see that, as expected, the interruption systems are generally more efficient, particularly when the multi-body attribute is present. This difference is driven by the ability of the system to interrupt the user when the 1-best hypothesis is incorrect, and so limiting the time the user spends saying an poorly recognized utterance and the amount of time the system takes confirming it. An example, meant to compare with that in Table 9.10 is shown in Table 9.11.

9.7 Live Evaluation

We now compare turn-taking approaches with live users of the Hands-Free Email task. Specifically, we compare the time it takes for users to complete the body sub-task. We

Figure 9.4: Differences between mid-utterance interruption (MUI) dialogues and dialogues without interruption (non-MUI). We find that interruption provides substantial efficiency gains.

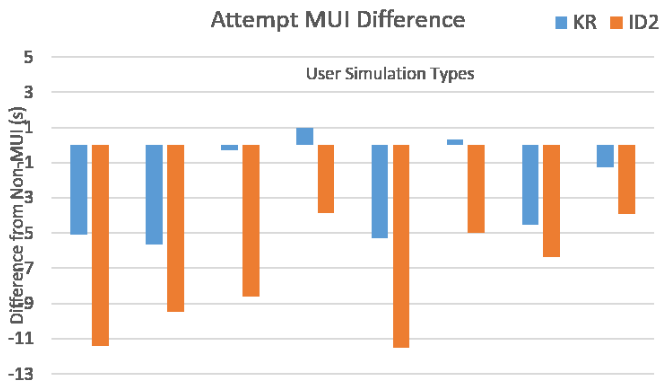


Table 9.11: ID2 MUI Example for ID2.

User 010	i read through the report and it looks good
ID2	ok.
User 010	i have so —
ID2	i thought you said i read through the report and it looks good
User 010	yes
ID2	i have so
User 010	no
ID2	What is the body?
User 010	i have a few questions so lets chat about it next week
ID2	i thought you said i have a few questions so lets chat about it next week

find that, under certain conditions, the user's of ID2 complete the body sub-task in less time than those using the KR system. As expected, we find this difference because ID2 can accomodate both Expert and Novice users, where the KR system cannot. We also find that the use of mid-utterance interruption leads to more efficient interactions.

We were uncertain whether adaptation would be advantageous with live users for the ID2 system. An ID2 system that adapts will use a high bid instead of a middle bid with the confirm-prompt following a confirmation. We call this *Novice Adaptation*. On the one hand, *Novice Adaptation* eliminates the opportunity for a Novice user to adapt to the system and behave more like an expert. On the other hand, if the system does not apply *Novice Adaptation* and Novice user's do not adapt themselvers than there is a large amount of unnecessary silence which impacts the efficiency of the dialogue. We investigate this design decision experimentally with live users by using a non-adaptive system, as we can artificially determine the efficiency of adaptation if the users *do not* adapt after the first body confirmation. We were unable to model this in simulation since we did not know what percentage of real users would adapt to the system.

A development system was used to tune both the simulation, system behaviors, instructions, language model, and pay schedule. A total of 50 development calls were collected and multiple development KR and ID systems were used in this step. Our experimental goal is to compare turn-taking approaches and we strove to create an environment to facilitate this and so, in addition to selecting the three attributes, we made a number of decisions based on these calls: **(1)** To explicitly instruct the users to speak each line of body separately. As the data shows, every user did not follow these instructions, but without them *every* user spoke the body as one long utterance; **(2)** To create oracle parsing and force the system to confirm each line of body separately; **(3)** To craft the language model so that recognition would be quite accurate when each line was spoken individually; **(4)** To group HITS in batches of 3/5 instead of just 1. This increased our per day call rate from roughly 10/day to 34/day. While this encourages repeat users we feel that (a) any live dialogue system will be used by the same customer population and (b) keeping the batch to 1 HIT would not dissuade repeat users either; **(5)** To use "priming" examples for importance-driven systems similar to those shown in the above tables. We found that Turkers were unlikely to modify their behavior unless explicitly told that they could, and so would basically continue to exhibit extreme system initiative behavior. This could be because they were understandably unaware of the system capabilities, were worried about doing the task wrong and not getting paid, or simple preferred that mode of interaction. We wished to eliminate the former two reasons, and so provided some examples at the beginning of the HIT that was specific to the system they would interact with. However, as the data shows the importance-driven system still received a wide variety of interaction patterns so, even though HITs displayed an example Expert interaction, the users did not follow this example a large percentage of the time.

9.7.1 The Data

A total of 356 verified calls were received by the 5 versions of the system. Of these, 296 calls had all four lines of body from 68 different users. Of the remaining 60 calls: 32 calls reached the attempt limit, 18 calls were ended prematurely by the user (saying no more body, when there was), and 10 had system errors. We focus our analysis on the 296 calls that have all four lines of body.

Recruiting Live Users: The users were recruited on Mechanical Turk and paid $0.25 per verified call. The task was either presented (randomly) as a set of 3 HITS (Human Intelligence Tasks) or a set of 5. The "Turkers" were instructed to author the email using the system via a toll-free number and, upon completion, enter a 3-digit verification code and fill out a short survey. The survey asked questions regarding the interaction and past experience with SDS: did the users have experience with virtual agents, whether the interaction was natural, whether they would use it again, was the interaction was successful...etc. We found no difference between survey responses for any system; indeed the same user would have wildly different survey responses for the same system and comparable interactions. Each HIT accesses the same system and were made available in the following order: KR, KR-MUI, ID1, ID2, ID2-MUI. We selecting this ordering, from simple to complex, so that our turkers would not attempt to use more sophisticated turn-taking with the simpler systems as we felt this to give a potentially unfair advantage to the importance-driven systems. Following this initial ordering the systems were presented in no particular order.

The Calls: We break up the calls into two different conditions. Those without failed body attempts — BF=0 — as determined by the user rejecting a system confirmation, and those with failed body attempts — BF>0. We also analyze the calls for the presence of each of the three attributes discussed earlier in Section 9.5. Note that even with specific instructions (Section 9.7) users still speak the lines of body as one utterance (last code element = 1) and even with priming examples a substantial number of users still behaved as Novices (first two code elements = 0). As mentioned in Section 9.5, the code elements indicate which of the three attributes are present in the user's behavior.

Table 9.12: Mean(median) seconds for filling the recipient/subject slots

	KR	ID1	ID2	KR-MUI	ID2-MUI
BF 0 C. Fails	20.7 (15.9)	19.84 (15.6)	17.8 (14.973)	0	17.57 (14.92)
BF $>$ 0 C. Fails	22.0 (17.7)	23.1 (16.7)	21.5 (17.6)	22.1 (15.911)	23.8 (16.7)

We evaluate the different turn-taking approaches by comparing the time it takes the system to collect the email body, as it is this sub-task that the interaction policies differ. As there are no interaction policy differences the recipient and subject sub-task, we expect there to be minimal performance differences. Indeed this is the case and mean/median times are shown in Table 9.12.

Evaluation Method: Our dependent variable is the seconds that it takes to complete the body sub-task and our independent variable is the system version. Recall that we only consider successful calls. As many users make multiple calls, our analysis focuses on mean durations of unique users within the system version and body attempt condition (BF=0 or BF>0). For all comparison, significance is determined by applying Wilcoxan Rank sum analyses to the mean call duration for each unique worker in each condition. While some users call multiple systems, there are insufficient instances of this to perform within-user analyses.

We divide the calls into two groups based on three user attributes. We define "Novice" calls as those that require extreme system initiative (first two attributes are always 0) and "Expert' calls as those that at some point do not require extreme system initiative behavior: at some point the user will provide information without an explicit prompt.

9.7.2 Results: System and User Silence

Here, we compare the efficiency of completing the body sub-task between the KR, ID, and ID2 system without mid-utterance interruption (MUI). This provides a live evaluation of the turn-bidding behavior we have described throughout this thesis. The worker and call statistics are provided in Table 9.13: the outer number is the worker count and the inner number is the call count. As the numbers of workers are similar we are not concerned with a misapplication of null hypothesis testing due to skewed population counts. As our

Table 9.13: Workers(Calls) without MUI

	KR	ID1	ID2
All	30 (81)	18 (53)	31 (126)
BF 0	18 (36)	10 (21)	15 (71)
BF $>$ 0	19 (45)	16 (32)	23 (55)

primary comparison is KR and ID2, it was critical to get a good estimate for each. We collected more ID2 data than KR since ID2 variability is substantially greater than KR: the between-worker standard deviation being 37.59 seconds for ID and 21.507 seconds for KR. This variance is expected since the nature of ID2 accommodates different types of users whereas KR only accounts for one. Since the recognizer, language model, and task are static between system versions we do not feel that the body attempt rate to be a meaningful evaluation metric and instead focus on how the system operates with and without failed confirmation attempts by separating the calls into two groups.[2] We first look at the BF=0 calls, directly comparing turn-taking approaches in ideal conditions where there is no failed confirmation attempts for the email body. We then explore differences in the BF $>$ 1 calls, where the user indicates at least one recognition error.

Body Attempt Fails = 0 (BF=0)

This condition directly compares the turn-taking approaches in an ideal environment where there is no (user-indicated) recognition error. We note that this is the more important condition because it highlights differences when recognition is perfect. Efficiency differences in this condition are entirely due to turn-taking approach, indicating that this work will continue to have meaningful impact as recognition performance improves. If differences were only found in the BF ¿ 0 condition, then future impact will be minimal since efficiency gains would be reduced as recognition improves.

The worker and call statistics for BF=0 are shown in the top portion of Table 9.14. Notice that even with priming examples (Section 9.7, a substantial number of ID calls are classified novice. Also note that a number of workers exhibit both expert and novice

[2]Indeed, when efficiency is normalized by body attempt, the results and conclusions do not change

Table 9.14: BF 0 Worker (Calls) by attributes and constraints

	KR	ID1	ID2
All	18 (36)	10 (21)	15 (71)
Novice	18 (36)	4 (6)	7 (24)
Expert	0 (0)	7 (15)	11 (47)
	All Adapt Constrained		
	18 (36)	-	13 (59)
	All OPA Constrained		
	18 (36)	6(7)	14 (56)
	All Adapt+OPA Constrained		
	18 (36)	6 (7)	12(47)

attributes (the sum of the Novice and Expert workers exceed All workers). This suggests that attempts to classify users as experts or novices to determine the appropriate dialogue policy based on previous interactions may not be entirely successful. We first compare the efficiency of KR, ID1, and ID2 (Table 9.14) and find that there is no significant difference in the time it takes to complete the body sub-task between KR, ID1, and ID2. This might be somewhat surprising especially for ID2 versus the other ID1 and KR, as ID2 is 10% faster on average. However, the high variance of ID2 might require more data to show significance.

We then separate the ID calls by Expert and Novice attributes. We find that there is a strong trend that ID novices are worse than KR users ($p<0.1$) and ID experts are significantly better than KR users, particularly ID2 experts ($p=0.00046$, $r=0.52$). In other words, while ID Expert performance is much more efficient than KR users (who are all treated as novices), the ID behavior with Novice user's is so poor that the performance of the Expert calls cannot compensate. It is from this difference that the larger standard deviation arises.

We artificially modified the calls and call counts to understand and further explore our results (Table 9.16). This artificial manipulation took two forms. First, we investigated the application of *Novice Adaptation* discussed in Section 9.5. Second, we determined to what extent unexplored regions of the interaction policy space affected the results.

Table 9.15: BF=O: Mean (median) seconds to complete body sub-task, where $^+$ indicates $p<0.1$ and * indicates $p=0.00046$

	KR	ID1	ID2
All	63.6 (61.1)	65.7 (63.0)	57.6 (57.8)
Novice	63.6 (61.1)	78.55 (69.0)$^+$	68.8 (67.2)$^+$
Expert	0	55.2 (53.2)$^+$	50.6 (49.0)*

In Section 9.5 we described how Novice Adaptation could dissuade users from exhibiting expert behavior later in the call (*user* adaptation). To investigate this we applied a version of the Interaction Policy that does not perform this Novice adaptation. We found that only 17% of ID2 calls exhibit *user* adaptation (All Adapt Constrained in Table 9.14) and so, after removing these calls we can artificially apply Novice Adaptation. This is done by modifying the body sub-task time by substituting high bids (0.1 seconds) for the Novice Adaptation middle bids (2 seconds). The results of this modification are shown in Table 9.16 (Adaptation), where ID2 now achieves a trend towards significant efficiency improvements over both user types.

We were concerned that our user simulation was deficient and so the ID2 system would encounter numerous states that had not been seen in training. The simulation modeled each user to either have or not have an attribute. However, in reality, many users did not exhibit the attribute consistently throughout a dialogue and so the learning agents entered unfamiliar areas of the state space and was forced to choose actions randomly. However, if we only consider calls where both the turn and utterance policy are followed than nearly every call is eliminated. We compromise and only constrain by excluding calls with turn taking policy deviation (OPA; Only Policy Actions), which accounts for 15 calls. We rush to point out that this limited number is testament to the generalizibility delivered by our multi-agent approach and would be impossible using a single MDP.

We find that, as Table 9.16 shows, when we apply artificial Novice adaptation to only calls without unfamiliar turn-taking states (Adaptation + OPA), ID2 performance approaches significance and has a medium effect size (p = 0.06, r = 0.34). Combined with the strong effect for the experts, we believe there is strong evidence in support of the importance-driven approach when recognition errors are not present with the caveat that

Table 9.16: BF O: Modified mean (median) seconds to complete body sub-task, where $^+$ indicates p<0.1 and * indicates p = 0.06

KR	ID1	ID2
Adaptation		
63.6 (61.1)	-	54.98 (57.0)$^+$
Only Policy Actions(OPA)		
63.6 (61.1)	56.55 (58.0)	55.7 (63.0)
Adaptation + OPA		
63.6 (61.1)	56.55 (58.0)	52.53 (57.0)*

the system should adapt to users instead of expecting the user's adapt to the system.

Body Attempt Fails > 0

We next examine BF>0 calls (worker and call statistics in Table 9.17, AC = All Constraints).

We find that ID1(91.02) and ID2 (107.85) are performing far worse than the KR approach (86.93), as shown in Table 9.18. While breaking the calls into Novice and Expert does indeed show differences between two groups, we were surprised to find that Expert ID2 users (96.85) were still slower than KR users on average.

We apply the same manipulations as we did for BF 0 calls, artificially adding Novice Adaptation and constraining the calls to be only be on-policy for turn-taking. We find that, as before, the performance of ID1 and ID2 increases considerably (Table 9.19) but does not reach a statistical significance difference.

In examining the call logs, we found that a non-trivial percentage of calls included an instance where the system succeed in speaking an utterance with a low bid which is a 10 second pause following a question. In these situations, the user is obviously confused or somehow incapacitated. Under this bid constraint (BC in Table 9.19), there are no differences between the systems. When adjusted for on-policy turn-taking actions and Novice Adaptation, leaving 18 workers and 42 calls for KR, and 7 workers and 9 calls for ID2 , ID2 performs 12% better than the KR system (though statistical significance is not achieved). However, again the balance of workers and calls is large so the comparison is questionable. Although it was hard to see a difference in time-to-completion, perhaps

Table 9.17: BF>0 Worker (Calls) by attributes. AC = All Constraints

	KR	ID1	ID2
All	19 (45)	16 (32)	23 (55)
Novice	19 (45)	8 (13)	12 (23)
Expert	19 (45)	12 (19)	13 (32)
All - AC	18 (42)	-	7 (9)

Table 9.18: BF > 0: Mean (median) seconds to complete the body sub-task, where * indicates p<0.033

	KR	ID1	ID2
All	86.93 (84.0)	91.02 (91.0)	107.85 (109.0)*
Novice	86.93 (84.0)	114.88 (111.0)	116.79 (117.0)
Expert	0 (0)	81.95 (83.0)	96.85 (87.0)

Table 9.19: BF > 0: Modified mean (median) seconds to complete the body sub-task, where * indicates p<=0.05

KR	ID1	ID2
BC		
85.33 (82.0)	81.54 (76.0)	86.33 (87.0)
Adaptation		
86.93 (84.0)	91.83 (92.0)	100.86 (101.0)
Adaptation + OPA		
86.93 (84.0)	66.63 (70.0)	91.69 (94.0)
Adaptation + OPA + BC		
85.33 (82.0)	-	75.96 (69.0)

due to a number of issues (simulated users, classification of users, bid time selections), there are some very interesting behaviors that can be done and elicited by using the importance-driven approach.

Behavioral Differences

Here, we use heat graphs to represent the behavioral differences between the systems when used with live users. This emphasizes the greater flexibility and breadth of behavior that importance-driven system have. The vertical represents the speech act at time T (A) and the horizontal the speech at time T+1 (A-prime) The lightness of the cell indicates the

Table 9.20: Workers(calls) with ASR errors

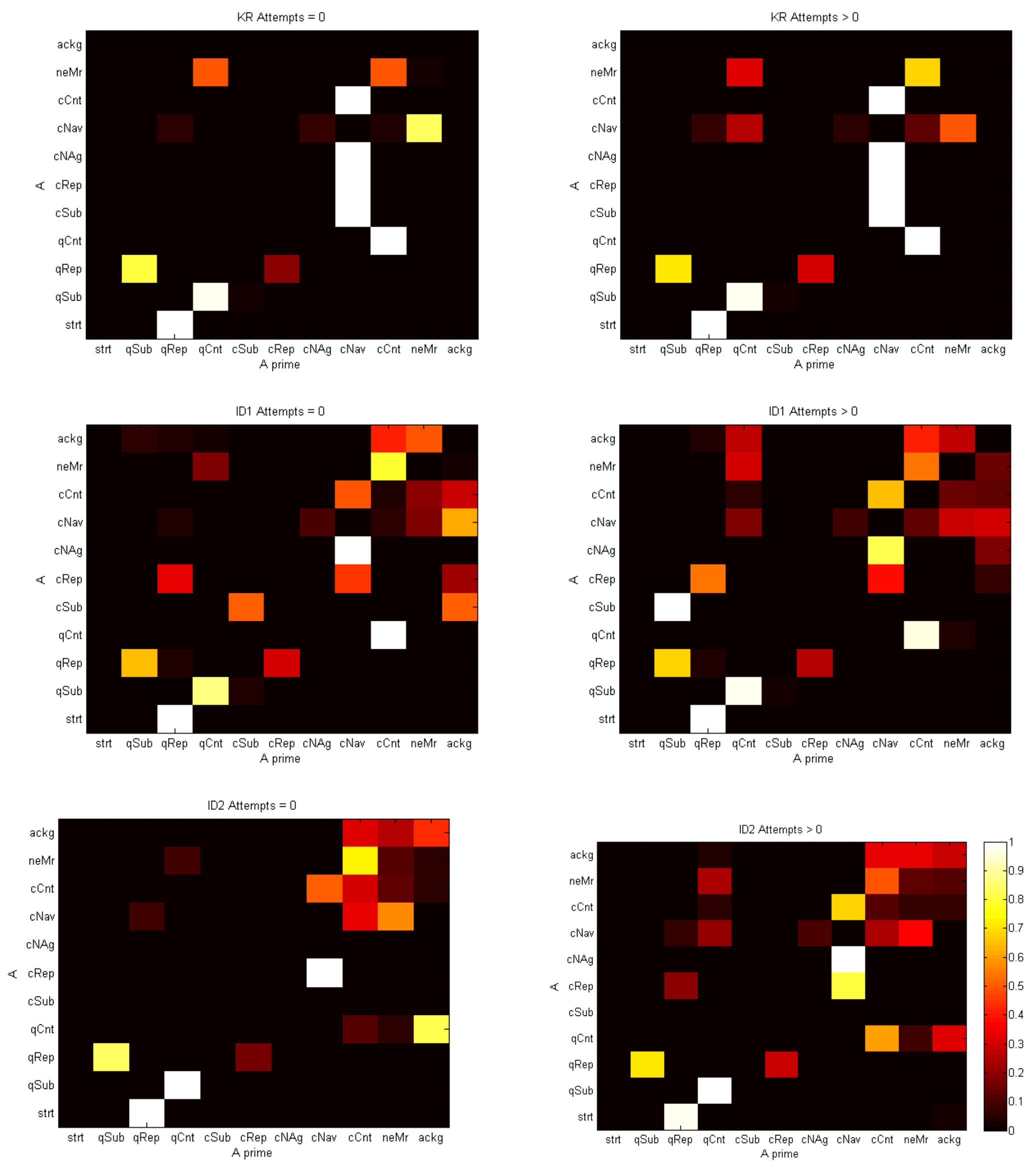
	KR	MUI-KR	ID2	MUI-ID2
All	21 (47)	12 (17)	23 (52)	8 (16))
Novice	-	-	13 (25)	3 (6))
Expert	-	-	12 (27)	6 (10))

probability of transitioning from action A to action A-prime where closer to white indicates a higher probability. We first note that there is far more variability in the ID versions than the KR version. Furthermore, as expected, where ID1 never transitions from acknowledge to acknowledge, ID2 does quite often. This is due to its greater confirmational freedom. Also, while the behavioral patterns are similar from BF=0 (left column) to BF $>$ 0 (right column) for ID1, ID2 exhibits greater behavioral differences between BF=0 and BF>0 is far more variable and has more transitions. This

9.7.3 Results: Mid-Utterance Interruption

We now evaluate the utility of mid-utterance interruption, where the system can interrupt the user if it believes a mis-recognition to be occurring based on the *oracle confidence* measure described in Section 9.4.2. The potential benefit of this behavior is that the system can minimize amount of mis-recognized speech. For this we compare calls with MUI with calls that have ASR errors that *would* have triggered an MUI, based on the *oracle confidence*. The call statistics are shown in Table 9.20. Here, since we are comparing within turn-taking strategies, we forgo adaptation or policy constraints (which is why ID2 has worse efficiency than KR). The number of MUI calls are substantially smaller since one cannot force user's to make ASR errors so a large number of MUI-enabled calls had no instance of MUI and were used in the previous section. Significant differences are not found between the two individual systems when evaluating MUI (Table 9.21), though the difference approaches significance for ID2. However, in general, the MUI calls are far more efficient (up to 25% for ID2), so we suspect our low N is a major hurdle for reaching statistically significant results. To wit, as there are also no differences within the Non-MUI and MUI systems we can directly evaluate the effect of MUI over both systems

Figure 9.5: Action Transition Map of Calls



by combining the non-MUI calls and combining the MUI calls. The results are shown in Table 9.22. We find strong statistically significant gains for the MUI system and a medium effect size for those gains (note this difference persists when we apply any and all of the constraints shown above). This is compelling evidence for the use of incremental speech recognition to enable the system to interrupt the user to remedy misunderstandings. However, this result only holds when confidence scoring is quite reliable, as here we use

Table 9.21: Mean (median) seconds to complete body sub-task for calls with ASR errors, where * indicates p<0.058

	KR	MUI-KR	ID2	MUI-ID2
All	86.05 (82.0)	74.61 (68.0)	105.56 (109.0)	79.04 (86.0)*
Novice	86.05 (82.0)	74.61 (68.0)	109.78 (117.0)	96.56 (90.0)
Expert	0 (0)	0 (0)	97.22 (91.0)	70.26 (76.0)

Table 9.22: Comparing No MUI and MUI Calls. p<0.0074, r=0.33

No MUI	MUI
96.2 (91.0)	76.4 (73.6)

oracle confidence.

9.7.4 Importance-Driven Evaluation Conclusion

Simulation experiments illustrated differences between the systems. The Importance-Driven systems are more flexible, learning synergistic behaviors between utterance and turn-taking behaviors. Simulation experiments also found that these behaviors must be learnt together, as utterance policies are no longer optimal when applied to different turn-taking capabilities.

We applied policies learnt in simulation to live users and initially found that, in the BF=0 condition, while there was no difference between ID and KR systems, ID2 Expert users were significantly faster than KR users. Moreover, we found that the system should adapt to the user, as it is very unlikely for the user to adapt to it, and that user simulation mismatch was a substantial hindrance to high performance (BF. [53]). We then found that, when these things are controlled for, ID2 is more efficient to KR. Specifically, it allows expert users to interact far more efficiently than with the KR system which is unable to handle user turn-taking and initiative differences. In the BF>0 condition we found mixed and confusing results. In general, ID2 performed worse than KR until we controlled for executing speech acts with low bids. Further work is required to fully evaluate importance-driven turn-taking in domains where speech recognition errors are likely.

In addition to evaluating importance-driven turn-taking in situations where both the system and user are silent, we evaluated the use of mid-utterance interruption. We found that the ability to interrupt the user to remedy some misunderstanding was extremely effective at increasing dialogue efficiency and, while this finding has been reported for a reading task [1], it is the first to show it for an interactive dialogue system.

9.8 Conclusion

This chapter has provided a live evaluation of Importance-Driven Turn-Taking and outlined a the Tempest architecture to build an importance-driven dialogue system. We have described a novel semi-hierarchical multi-agent reinforcement learning paradigm for spoken dialogue systems and demonstrated how the flexible generalizability of that architecture is critical for performance with live users. We have also shown how utterance policy optimality is dependent on turn-taking capabilities. Finally, a live evaluation of the importance-driven approach showcased its flexibility in handling multiple user types and we found a strong result for using mid-utterance interruption.

Chapter 10

Conclusion

10.1 Summary

Spoken dialogue systems are quickly becoming a part of everyday life, and will soon play an integral role in the modern world. One critical component of such systems is turn-taking and initiative: knowing what to say, how to say it, and when to speak. The majority of current approaches have taken the narrow view that the system should only seek to minimize the gaps and overlaps between speaking turns. While superficially valid, this approach disregards the negotiative nature of turn-taking and the critical role it plays at effective and efficient interactions, resulting in highly-constrained dialogues. This thesis takes an alternate view to turn-taking and, seeking to underscore the interaction between turn-taking and initiative, views speaking turns as units that emerge from a continuous interaction.

In this thesis we have proposed Importance-Driven Turn-Taking, a behavioral approach that combines reinforcement learning with a variable strength turn-taking signal. This importance-driven approach was inspired by psycholinguistic research that finds the importance of speaking is a fundamental motivation behind the timing of the utterance. Reinforcement Learning plays a central role in importance-driven turn-taking, as it enables the system to implicitly determine the importance of the utterance. We have found that the reinforcement learning can successfully learn utterance and turn-taking policies, using either a single MDP for both decisions or a single MDO for *each* decision. While the latter approach is slightly more complex, it enables the state features of each MDP to be decision specific which decreases training time and increases generalizibility. This is ideal for RL where the user simulation may not match the live user population, as occurred in the hand-free email evaluation. We have also found that the utterance and turn-taking policy must be learned simultaneously, as utterance policy optimality can be assured between turn-taking policies. In practice, importance-driven turn-taking allows the system to handle multiple types of users, specifically expert and novice users. Where Keep-or-Release systems do not have the behavioral flexibility to cater to both types of users, the importance-driven system has greater turn-taking capabilities and so can adapt to differing user behaviors during the dialogue.

We have also described methods to implement and develop systems that operate outside the traditional turn boundaries. This work revolves around incremental speech processing and temporal simulation. We have proposed novel methods for leveraging incremental results for dialogue systems, both for the actual recognition algorithm and for downstream dialogue-centric components. The temporal simulation we have described allows systems with advanced turn-taking to be trained off-line and then seamlessly integrated into a live dialogue system. Both of these contributions were used in our live evaluations.

Our live evaluations cover all three of the decision points that we have described in this thesis: system silence/inactive recognizer, system silence/active recognizer, and system speech/active recognizer. We first evaluated the Prediction-Based Barge-in Response (PBR) model, which operates in the latter two contexts, in a live spoken dialogue task with real users. While not using RL, this model was based off the importance-driven theory, and we view it as a partial validation of the method. The PBR model views the system's turn as more important than any input that will not be understood by the system and uses prompt suspension and resumption to actualize this guiding principle. We then evaluated a true importance-driven system, that covered the first and and second contexts, in a live hands-free email task with mechanical turk users. We found that when the system was operating in a known state space and allowed to adapt to the user, the importance-driven approach was more efficient than the keep-or-release approach. This was precisely because of its ability to manage expert and novice users. We also note that both of these evaluations provide strong evidence for the initiation of system speech when the recognizer is active.

10.2 Future Directions

There are a number of fruitful research directions, in both broadening the application of the importance-driven approach and in refining its components.

In this thesis, we have shown results for task-driven spoken dialogue systems. However, we feel that the importance-driven approach could be applied to other domains where task success is not particularly clear such as speech to speech machine translation or web browsing. While the challenge of constructing the appropriate reward function is non-trivial, we feel the interactive flexibility provided by our approach is desirable as the system must handle users that are, by definition in the case of speech to speech translation, different. We also feel that it would be interesting to apply our multi-agent architecture to coordinating multi-modal behavior such as gesture or information display actions.

Our reinforcement learning and temporal simulation were directed towards providing a live evaluation of the importance-driven approach, but are far from perfect. We are particularly interested in applying abstraction techniques to differentiate interaction policies that are domain independent from those that are unique to a single domain. This work will be instrumental in efficiently applying sophisticated yet generic dialogue systems to new domains, and enabled non-expert authoring of next-generation speech systems.

Bibliography

[1] Aist, G. Expanding a time-sensitive conversational architecture for turn-taking to handle content-driven interruption. In *Fifth International Conference on Spoken Language Processing* (1998).

[2] Aist, G., Allen, J., Campana, E., Gallo, C. G., Stoness, S., Swift, M., and Tanenhaus, M. K. Incremental understanding in human-computer dialogue and experimental evidence for advantages over nonincremental methods. In *In Proc. DECALOG* (2007), pp. 149–154.

[3] Allen, J., Guinn, C., and E., H. Mixed-initiative interaction. *IEEE Intelligent Systems 14*, 5 (1999), 14–23.

[4] Baumann, T. Simulating spoken dialogue with a focus on realistic turn-taking. *Proceedings of the 13th ESSLLI Student Session* (2008).

[5] Baumann, T., Atterer, M., and Schlangen, D. Assessing and improving the performance of speech recognition for incremental systems. In *Proc. NAACL: HLT* (2009), Association for Computational Linguistics, pp. 380–388.

[6] Black, A., Burger, S., Langner, B., Parent, G., and Eskenazi, M. Spoken dialog challenge 2010. In *Spoken Language Technology Workshop (SLT), 2010 IEEE* (2010), IEEE, pp. 448–453.

[7] Bohus, D., and Horvitz, E. Decisions about turns in multiparty conversation: From perception to action. In *Proceedings of ICMI* (2011).

[8] Busoniu, L., Babuska, R., and De Schutter, B. A comprehensive survey of multiagent reinforcement learning. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on 38*, 2 (2008), 156–172.

[9] Chu-Carroll, J., and Carberry, S. Response generation in collaborative negotiation. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics* (Morristown, NJ, USA, 1995), Association for Computational Linguistics, pp. 136–143.

[10] Cuayáhuitl, H. Hierarchical reinforcement learning for spoken dialogue systems.

[11] Cuayáhuitl, H., Renals, S., Lemon, O., and Shimodaira, H. Evaluation of a hierarchical reinforcement learning spoken dialogue system. *Computer Speech & Language 24*, 2 (2010), 395–429.

[12] Dethlefs, N., Hastie, H., Rieser, V., and Lemon, O. Optimising incremental dialogue decisions using information density for interactive systems. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)* (2012).

[13] DeVault, D., Sagae, K., and Traum, D. Can i finish? learning when to respond to incremental interpretation results in interactive dialogue. In *Proceedings of the SIGDIAL 2009 Conference* (London, UK, September 2009), Association for Computational Linguistics, pp. 11–20.

[14] Duncan, S. Some signals and rules for taking speaking turns in conversations. *Journal of Personality and Social Psychology 23* (1972), 283–292.

[15] Duncan, S., and Niederehe, G. On signalling that it's your turn to speak. *Journal of Experimental Social Psychology 10* (1974), 234–247.

[16] English, M., and Heeman, P. A. Learning mixed initiative dialog strategies by using reinforcement learning on both conversants. In *Proceedings of HLT/EMNLP* (2005), pp. 1011–1018.

[17] Ferrer, L., Shriberg, E., and Stolcke, A. Is the speaker done yet? faster and more accurate end-of-utterance detection using prosody. In *Seventh International Conference on Spoken Language Processing* (2002).

[18] Fink, G., Schillo, C., Kummert, F., and Sagerer, G. Incremental speech recognition for multimodal interfaces. In *Industrial Electronics Society, 1998. IECON'98. Proceedings of the 24th Annual Conference of the IEEE* (1998), vol. 4, IEEE, pp. 2012–2017.

[19] Fry, D. Simple reaction-times to speech and non-speech stimuli. *Cortex 11*, 4 (1975), 355–360.

[20] Fry, D. B. Simple reaction-times to speech and non-speech stimuli. *Cortex 11*, 4 (1975), 355–360.

[21] Genkin, A., Shenzhi, L., Madigan, D., and Lewis, D. Bayesian logistic regression. *http://www.bayesianregression.org* (2011).

[22] Georgila, K., Wolters, M., and Moore, J. Learning dialogue strategies from older and younger simulated users. In *Proceedings of the 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue* (2010), Association for Computational Linguistics, pp. 103–106.

[23] Goffin, V., Allauzen, C., Bocchieri, E., Hakkani-Tur, D., Ljolje, A., Parthasarathy, S., Rahim, M., Riccardi, G., and Saraclar, M. The AT&T WATSON speech recognizer. In *Proceedings of ICASSP* (2005), pp. 1033–1036.

[24] Gravano, A., and Hirschberg, J. Turn-taking cues in task-oriented dialogue. *Computer Speech & Language 25*, 3 (2011), 601–634.

[25] Heeman, P. Combining reinforcement learning with information-state update rules. In *Proceedings of the Annual Conference of the North American Association for Computational Linguistics* (Rochester, NY, 2007), pp. 268–275.

[26] Hirasawa, J., Nakano, M., Kawabata, T., and Aikawa, K. Effects of system barge-in responses on user impressions. In *Sixth European Conference on Speech Communication and Technology* (1999), ISCA.

[27] Hulstijn, J., and Vreeswijk, G. Turntaking: a case for agent-based programming. Tech. rep., Institute of Information and Computing Sciences, Utrecht University, 2003.

[28] Jaffe, J., and Feldstien, S. *Rhythms of Dialogue.* Academic Press, 1970.

[29] Janarthanam, S., and Lemon, O. Learning to adapt to unknown users: referring expression generation in spoken dialogue systems. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics* (2010), Association for Computational Linguistics, pp. 69–78.

[30] Jonsdottir, G. R., Thorisson, K. R., and Nivel, E. Learning smooth, human-like turntaking in realtime dialogue. In *IVA '08: Proceedings of the 8th international conference on Intelligent Virtual Agents* (Berlin, Heidelberg, 2008), Springer-Verlag, pp. 162–175.

[31] Kallirroi, G., Wolters, M., and Moore, J. Learning dialogue strategies from older and younger simulated users.

[32] Kronlid, F. Turn taking for artificial conversational agents. In *Cooperative Information Agents*. Springer-Verlag Berlin Heidelberg, 2006, pp. 81–95.

[33] Larsson, S., and Traum, D. Information state and dialogue managment in the trindi dialogue move engine toolkit. *Natural Language Engineering 6* (2000), 323–340.

[34] Lemon, O., Georgila, K., Henderson, J., and Stuttle, M. An ISU dialogue system exhibiting reinforcement learning of dialogue policies: generic slot-filling in the TALK in-car system. In *Proceedings of the Eleventh Conference of the European Chapter of the Association for Computational Linguistics: Posters & Demonstrations* (2006), Association for Computational Linguistics, pp. 119–122.

[35] Levin, E., Pieraccini, R., and Eckert, W. A stochastic model of human-machine interaction for learning dialog strategies. *IEEE Transactions on Speech and Audio Processing 8*, 1 (2000), 11 – 23.

[36] Litman, D., Kearns, M., Singh, S., and Walker, M. Automatic optimization of dialogue management. In *Proceedings of the 18th conference on Computational linguistics-Volume 1* (2000), Association for Computational Linguistics, pp. 502–508.

[37] Lu, D., Nishimoto, T., and Minematsu, N. Decision of response timing for incremental speech recognition with reinforcement learning. In *Automatic Speech Recognition and Understanding (ASRU), 2011 IEEE Workshop on* (2011), IEEE, pp. 467–472.

[38] Martin, D. L., Cheyer, A. J., and Moran, D. B. The open agent architecture: A framework for building distributed software systems. *Applied Artificial Intelligence: An International Journal 13*, 1-2 (Jan. 1999), 91–128.

[39] Martin, J. G., and Strange, W. The perception of hesitation in spontaneous speech. *Perception & Psychophysics 3*, 6 (1968), 427–438.

[40] Nishimura, R., and Nakagawa, S. A spoken dialog system for spontaneous conversations considering response timing and response type. *IEEJ Transactions on Electrical and Electronic Engineering 6*, S1 (2011), S17–S26.

[41] Ono, N., and Fukumoto, K. Multi-agent reinforcement learning: A modular approach. In *Proceedings of the Second International Conference on Multi-Agent Systems* (1996), pp. 252–258.

[42] Padilha, E., and Carletta, J. A simulation of small group discussion. In *Proceedings of EDILOG* (2002), pp. 117–124.

[43] Parent, G., and Eskenazi, M. Toward Better Crowdsourced Transcription: Transcription of a year of the Let's Go Bus Information System Data. *Proceedings of Interspeech 2005, Lisbon, Portugal* (2009).

[44] Ramshaw, L. A three-level model for plan exploration. In *Proceedings of the 29th annual meeting on Association for Computational Linguistics* (1991), Association for Computational Linguistics, pp. 39–46.

[45] Raux, A. *Flexible Turn-Taking for Spoken Dialog Systems.* PhD thesis, CMU, 2008.

[46] Raux, A., and Eskenazi, M. Optimizing the turn-taking behavior of task-oriented spoken dialog systems. *ACM Transactions on Speech and Language Processing (TSLP) 9*, 1 (2012), 1.

[47] Raux, A., Langner, B., Bohus, D., Black, A., and Eskenazi, M. Lets go public! taking a spoken dialog system to the real world. In *in Proc. of Interspeech 2005* (2005).

[48] Rose, R. C., and Kim, H. K. A hybrid barge-in procedure for more reliable turn-taking in human-machine dialog systems. In *Automatic Speech Recognition and Understanding, 2003. ASRU'03. 2003 IEEE Workshop on* (2003), IEEE, pp. 198–203.

[49] Russell, S., and Norvig, P. *Artificial intelligence: a modern approach.* Prentice hall, 2010.

[50] Sacks, H., Schegloff, E., and Jefferson, G. A simplest systematics for the organization of turn-taking for conversation. *Language 50*, 4 (1974), 696–735.

[51] Sagae, K., Lehr, M., Prud'hommeaux, E., Xu, P., Glenn, N., Karakos, D., Khudanpur, S., Roark, B., Saraçlar, M., Shafran, I., et al. Hallucinated n-best lists for discriminative language modeling. In *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on* (2012), IEEE, pp. 5001–5004.

[52] Sato, R., Higashinaka, R., Tamoto, M., Nakano, M., and Aikawa, K. Learning decision trees to determine turn-taking by spoken dialogue systems. In *ICSLP* (Denver, CO., 2002), pp. 861–864.

[53] Schatzmann, J., Weilhammer, K., Stuttle, M., and Young, S. A survey of statistical user simulation techniques for reinforcement-learning of dialogue management strategies. *The Knowledge Engineering Review 21*, 2 (2006), 97–126.

[54] Scheffler, K., and Young, S. Automatic learning of dialogue strategy using dialogue simulation and reinforcement learning. In *Proceedings of the second international conference on Human Language Technology Research* (2002), pp. 12–19.

[55] Schegloff, E. Overlapping talk and the organization of turn-taking for conversation. *Language in Society 29* (2000)), 1 – 63.

[56] Schlangen, D. From reaction to prediction: Experiments with computational models of turn-taking. In *Ninth International Conference on Spoken Language Processing* (2006).

[57] Schlangen, D., Baumann, T., and Atterer, M. Incremental reference resolution: The task, metrics for evaluation, and a bayesian filtering model that is sensitive to disfluencies. In *Proceedings of the SIGDIAL 2009 Conference: The 10th Annual Meeting of the Special Interest Group on Discourse and Dialogue* (2009), Association for Computational Linguistics, pp. 30–37.

[58] Selfridge, E., Arizmendi, I., Heeman, P., and Williams, J. Stability and accuracy in incremental speech recognition. In *Proceedings of the SIGdial 2011* (2011).

[59] Selfridge, E., Arizmendi, I., Heeman, P., and Williams, J. Integrating incremental speech recognition and pomdp-based dialogue systems. In *Proceedings of the SIGdial 2012* (2012).

[60] Selfridge, E., Arizmendi, I., Heeman, P., and Williams, J. Continuously predicting and processing barge-in during a live spoken dialogue task. In *Proceedings of the SIGdial 2013* (2013).

[61] Selfridge, E., and Heeman, P. Importance-Driven Turn-Bidding for spoken dialogue systems. In *Proc. of ACL 2010* (2010), Association for Computational Linguistics, pp. 177–185.

[62] Selfridge, E., and Heeman, P. A temporal simulator for developing turn-taking methods for spoken dialogue systems. In *Proceedings of SIGdial* (2012).

[63] Sidner, C. An artificial discourse language for collaborative negotiation. In *Proceedings of the National Conference on Artificial Intelligence* (1994), JOHN WILEY & SONS LTD, pp. 814–814.

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

[65] Smith, R., and Gordon, S. Effects of variable initiative on linguistic behavior in human-computer spoken natural language dialogue. *Computational Linguistics 23*, 1 (1997), 141–168.

[66] Spohrer, J., Brown, P., Hochschild, P., and Baker, J. Partial traceback in continuous speech recognition. In *Proceedings of the IEEE International Conference on* (1980).

[67] Ström, N., and Seneff, S. Intelligent barge-in in conversational systems. *Proceedings of ICSLP* (2000).

[68] Styler, W. The enronsent corpus. *Boulder: University of Colorado at Boulder Institute of Cognitive Science* (2011).

[69] Sutton, R., and Barto, A. *Reinforcement Learning.* MIT Press, 1998.

[70] Sutton, S., Novick, D., Cole, R., Vermeulen, P., de Villiers, J., Schalkwyk, J., and Fanty, M. Building 10,000 spoken-dialogue systems. In *ICSLP* (Philadelphia, Oct 1996).

[71] Thórisson, K. Natural turn-taking needs no manual: Computational theory and model, from perception to action. *Multimodality in language and speech systems 19* (2002).

[72] Traum, D., and Hinkelman, E. Conversation acts in task-oriented spoken dialogue. *Computational Intelligence 8*, 2 (1992).

[73] Traum, D., and Rickel, J. Embodied agents for multi-party dialogue in immersive virtual worlds. In *Proceedings of Autonomous Agents and Multi-Agent Systems* (2002), pp. 766 – 773.

[74] Walker, M. Inferring acceptance and rejection in dialog by default rules of inference. *Language and Speech 39*, 2-3 (1996), 265.

[75] Walker, M., Hindle, D., Fromer, J., Di Fabbrizio, G., and Mestel, C. Evaluating competing agent strategies for a voice email agent. *EUROSPEECH97* (1997).

[76] Walker, M., and Whittaker, S. Mixed initiative in dialogue: an investigation into discourse segmentation. In *Proceedings of the 28th Annual Meeting of the Association for Computational Linguistics* (1990), pp. 70–76.

[77] Ward, N., Fuentes, O., and Vega, A. Dialog prediction for a general model of turn-taking. In *Eleventh Annual Conference of the International Speech Communication Association* (2010).

[78] Williams, J. An empirical evaluation of a statistical dialog system in public use. In *Proceedings of SIGdial 2011* (2011).

[79] Williams, J. An empirical evaluation of a statistical dialog system in public use. In *Proceedings of the SIGDIAL 2011 Conference* (2011), Association for Computational Linguistics, pp. 130–141.

[80] Williams, J., and Balakrishnan, S. Estimating probability of correctness for asr n-best lists. In *Proc SIGDIAL, London, United Kingdom* (2009).

[81] Williams, J., and Young, S. Characterizing task-oriented dialog using a simulated asr channel. In *Proceedings of the ICSLP* (2004), Citeseer.

[82] Williams, J., and Young, S. Partially observable markov decision processes for spoken dialog systems. *Computer Speech & Language 21*, 2 (2007), 393–422.

[83] Williams, J. D. A critical analysis of two statistical spoken dialog systems in public use. In *Spoken Language Technology Workshop (SLT), 2012 IEEE* (2012), IEEE, pp. 55–60.

[84] Williams, J. D., and Balakrishnan, S. Estimating probability of correctness for ASR N-Best lists. In *Proc. of SIGdial 2009* (2009), Association for Computational Linguistics, pp. 132–135.

[85] Yang, F., and Heeman, P. A. Initiative conflicts in task-oriented dialogue". *Computer Speech Language 24*, 2 (2010), 175 – 189.

Biographical Note

Ethan Selfridge was born in Eastford, Connecticut in 1984. He attended Rectory and Pomfret School before going to Reed College in 2002. In 2006 he graduated with a degree in Psychology. After working briefly at a start-up, he started at the Center for Spoken Language Understanding in the Fall of 2008 to pursue a graduate degree in Computer Science and Engineering. During his graduate training he worked primarily on increasing the turn-taking and initiative behavior of spoken dialogue systems using incremental speech processing and machine learning. His internship at ATT Labs – Research in the summer's of 2010 and 2011 produced 2 patent applications, and he continued to collaborate with the technical staff after the internships ended.

Publications Ethan O. Selfridge, Iker Arizmendi, Peter A. Heeman, and Jason D. Williams (2013): Continuously Predicting and Processing Barge-in During a Live Spoken Dialogue Task , Metz, France, August 2013 (Nominated for Best Paper)

Ethan O. Selfridge, Peter A. Heeman, Iker Arizmendi, and Jason D. Willams (2012): Demonstrating the Incremental Interaction Manager in an end-to-end "Lets Go!" dialogue system in Demonstration at IEEE Workshop on Spoken Language Technology (SLT), SLT 2012, Miami, FL, December 2012

Peter A. Heeman, Jordan Frye, Rebecca Lunsford, Andrew Rueckert, and Ethan O. Selfridge (2012): Using Reinforcement Learning for Dialogue Management Policies: Towards Understanding MDP Violations and Convergence , InterSpeech 20102, Portland, OR, USA

Ethan O. Selfridge, Iker Arizmendi, Peter A. Heeman, and Jason D. Williams (2012): Integrating Incremental Speech Recognition and POMDP-based Dialogue Systems , Seoul, South Korea, July 2012

Ethan O. Selfridge and Peter A. Heeman (2012): A Temporal Simulator for Developing Turn-Taking Methods for Spoken Dialogue Systems , South Korea, July 2012

Ethan O. Selfridge and Peter A. Heeman (2011): Learning Turn, Attention and Utterance Decisions in a Negotiative Slot-Filling Domain Technical Report CSLU-11-005, Center for Spoken Language Understanding, OHSU, October 2011

Ethan O. Selfridge, Iker Arizmendi, Peter A. Heeman, and Jason D. Williams (2011): Stability and Accuracy in Incremental Speech Recognition In Proceedings of the 12th Annual SIGdial Meetinig On Discourse and Dialogue, Portland, OR, June 2011. (Best Student Paper Award)

Peter A. Heeman, Rebecca Lunsford, Ethan O. Selfridge, Lois Black, and Jan van Santenn (2010): Autism and Interactional Aspects of Dialogue. SIGdial 2010, Tokyo, September 2010

Ethan O. Selfridge and Peter A. Heeman (2010): Importance-Driven Turn-Bidding for Spoken Dialogue Systems. In Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pages 177-185, Uppsala Sweden, July 2010.

Ethan O. Selfridge and Peter A. Heeman (2009): A bidding approach to turn-taking. Presented at the International Workshop for Spoken Dialogue Systems, July 2009