ABSTRACT

Specification and implementation of flexible human-computer dialog is challenging because of the complexity involved in rendering the dialog responsive to a vast number of varied paths through which users might desire to complete the dialog. To address this problem, we developed a toolkit for modeling and implementing task-based, mixed-initiative dialogs based on metaphors from lambda calculus. Our toolkit can automatically operationalize a dialog that involves multiple prompts and/or sub-dialogs, given a high-level dialog specification of it. The use of natural language with the resulting dialogs makes the flexibility in communicating user utterances commensurate with that in dialog completion paths—an aspect missing from commercial assistants like Siri. Our results demonstrate that the dialogs authored with our toolkit support the end user’s completion of a human-computer dialog in a manner that is most natural to them—in a mixed-initiative fashion—that resembles human-human interaction.

Keywords
Bag of words model; function currying; functional programming; human-computer dialogs; interactive voice response systems; k-nearest-neighbor classifier; lambda calculus; mixed-initiative dialogs; mixed-initiative interaction; natural language processing; partial evaluation.

1. PROBLEM AND MOTIVATION

Human-computer dialogs, which are used to improve information access from smart phone apps, ATMs, and airport kiosks to intelligent tutoring/training, are woven into the fabric of our daily interactions with computer systems. The problem addressed through our research is the automatic construction of mixed-initiative, human-computer dialog systems (see Fig. 1). Mixed-initiative interaction is a flexible interaction strategy whereby the user and the system engage as equal participants in an activity and take turns exchanging initiative as the user progresses toward the satisfaction of a particular goal facilitated by her interaction with the system [14]. Since “authoring a dialogue is like writing a movie script with many different endings” [17], “a central problem for mixed-initiative dialogue management is coping with utterances that fall outside of the expected sequence of the dialogue” [22] (e.g., see dialog in Fig. 2). Thus, “developing a mixed-initiative dialog system is a complex task” [15] and “[c]reating an actual dialog system involves a very intensive programming effort” [15].

This problem is important since dialog has been established as an effective mechanism through which to achieve a rich form of human-computer interaction (e.g., dialog-based systems are now used in areas as critical as health care [23]). Being able to automatically create a dialog system in a new domain is important. We feel that i) a mixed-initiative mode of interaction driven by user utterances and ii) communicated through the use of natural language (see lower right hand cell of Table 1) is the key to the effectiveness and widespread adoption of personal assistant technologies. This extended abstract discusses a research project that addresses (i) and (ii).

2. BACKGROUND AND RELATED WORK

Our research lies in the area of automatic mixed-initiative, dialog system construction, with a particular focus on the dialog management component (i.e., knowing what to prompt for and/or accept next based on what has already been communicated and the current utterance) of a dialog-based system [18]. Dialog-based systems can be classified based on the degree of flexibility and natural language supported (see Table 1). The increasing popularity of personal assistant technologies, such as Siri, Google Now, Cortana, and Alexa, is driving and expanding progress toward the long-standing, albeit challenging, goal of applying artificial intelligence to build human-computer dialog systems capable of understanding natural language [19]. There are multiple research projects which seek to automate the implementation of flexible, dialog-based systems [15, 17] What sets our research apart from these projects is our use of language-based concepts and operators, rather than task structures, to model dialog, which we discuss below.

Our work lies specifically in the dialog management area of dialog-based systems. The dialog management component plays a central role in the architecture of a traditional
Motivation: Enabling Complex Dialog

- The Subway dialog above illustrates a human-computer mixed-initiative interaction that, due to the complexity and variability of the dialog, is not possible to realize with other dialog modeling approaches. Trying to mix even 3 questions results in 8,191 possible unique dialog interactions.

  - Line (1) starts with a simple prompt for sub or salad and (2) shows the user responding directly to the prompt. This is the extent of flexibility (completely fixed) of most dialog systems today.
  - Line (3) shows the system soliciting for the next item in a script, sandwich size, but in (4), the user responds to a different but forthcoming solicitation for takeout. This out-of-turn interaction is a form of mixed-initiative interaction (MI) where the user and the system engage as equal participants in dialog.
  - In (5), the system again solicits for the unwrapped sandwich size and in (6), the user responds with a size and specifies the type of bread desired, completing another forthcoming solicitation for bread type. This illustrates a form of MI where the user provides information for more than one solicitation in a single utterance.
  - Line (7) shows the system accepting the user information from (6), and asking which specialty sandwich the user wants. In line (8), rather than providing information to the system, the user seeks information from the system, i.e., which specialty sandwiches have ham.
  - In (9), the system has successfully understood the user request for information and has provided the specialty sandwiches with ham. The user chooses one of the items with ham and also specifies the toppings peppers and olives.
  - Line (11) shows the system accepting the user information and asking if the user wants any additional toppings.
  - Our dialog toolkit supports all of these forms of mixed-initiative interaction.

Figure 2: Sample motivating, mixed-initiative dialog, built with our toolkit, running on a mobile phone.

dialog system, and is primarily concerned with controlling the flow of the dialog, while maintaining discourse history, sometimes referred to as system-action prediction, and coordinating with other (typically input/output) components of the system (e.g., automatic speech recognition, spoken language understanding, and presentation of results).

There are two main approaches to dialog management: task-based and data-driven. Our work combines the two: (i) our research targets task-based dialog systems whose goal is to support the user in satisfying clearly-defined goals by completing highly-structured tasks; and (ii) we use data-driven techniques (e.g., bag-of-words model and a k-nearest-neighbor classifier) to help support the users use of natural language to pursue these tasks (see Fig. 3—right). We focus on frameworks for (automatic) construction for their relatedness to our work [3, 4, 28].

The task-based approach involves modeling a collection of tasks to be supported by the system, using a modeling notation or language, and discerning how the user can be most effectively afforded the desired interaction flexibility in completing those tasks. Finite state automata (FSA), and other transition networks, context-free grammars (CFG), and events have been used as general task structures to model dialog [12].

Our approach factors the domain-dependent components of the system (e.g., the aspects of the dialog specific to the targeted domain) from the domain-independent components of the system (e.g., the dialog engine and management). Thus, our dialog engine acts as an interpreter, in the programming languages sense, for the given dialog specification. This approach provides a clean separation of the domain-dependent and independent aspects (e.g., control logic and dialog flow) [1]. This approach is used in the RavenClaw dialog management framework [3, 4, 31]. RavenClaw uses an agenda-based approach to task modeling [30, 31]. Our framework is an instantiation of this ‘separation of task model and dialog engine’ approach to dialog management (see Fig. 3—left).

3. APPROACH AND UNIQUENESS

Rather than agenda [31], rule-oriented [10], and the myriad of other task structures and task modeling approaches used for task-based dialog management, we use programming language theory. We designed a notation based on lambda calculus that serves as an authoring notation for specifying dialogs and also suggests implementation ideas. This distinguishes our model from other knowledge/task-based approaches which use hierarchical task/agency/role models. Using program transformations [24], including partial evaluation [16], and language concepts, to specify dialogs and to intentionally model multiple paths through a dialog without extensionally hardcoding each into the control flow of the implementation, is a fundamentally different approach to dialog modeling, management, and implementation.

Our approach is unique in that it involves thinking of dialog as a function and using concepts from programming language theory, including function currying and partial evaluation, to automatically modify that function to achieve a mixed-initiative mode of interaction. ‘As the user progresses through a dialog, we think of the steps that she takes as the evaluation of a function. Changing the evaluation method of the function (or transforming the function) then corresponds to different interaction policies [39] for the dialog (i.e., ways of mixing initiative). The overall idea is that different function evaluation strategies correspond to different interaction policies for the dialog (i.e., system initiated vs. mixed-initiative) or ways of mixing initiative’ [3]. The structure of an expression in our dialog-authoring notation, and the language concepts used therein, provides a pattern for implementing the dialog. Based on this foundation, we built a dialog modeling and implementation toolkit, which is capable of automatically realizing a variety of mixed-initiative dialogs given only a single, high-level specification of each.

While prior research projects have approached engineering interactive computing systems from the perspective of (functional) programming languages [11, 20, 28], only few have sought to marry human-computer dialogs with concepts from programming languages [4, 25]. Due to the conceptual analogs between natural languages and programming languages, viewing human-computer dialog modeling, management, and implementation from the perspective of programming language theory suggests a natural, yet under-
explored, approach to dialog representation and reasoning. The concepts from programming languages are not just helpful metaphors for dialog specification, but also lend insight into operationalizing dialogs.

Dialog-based systems such as Siri support utterances communicated through natural language, but are limited to action-requesting, information-seeking (e.g., ‘What is the weather forecast tomorrow?’), and information-providing utterances and, thus, only support a low degree of mixed-initiative interaction (see Fig. 3). We have enhanced our model for mixed-initiative dialog by using a bag-of-words model for a new dialog domain and a $k$-nearest-neighbor classifier to predict the context of a user utterance (i.e., map an unsolicited utterance to the dialog prompt to which it is a response) to improve the natural language and mixed-initiative capabilities of systems like Siri (see last row of Table 1). Fig. 4 (right) illustrates the design of the natural language processing unit in Fig. 3 (left). Fig. 5 demonstrates the natural language capabilities of our system. Our MPAA dialog toolkit is available at https://bitbucket.org/jwb_research/.

Figure 3: (left) Architectural design of our natural language, mixed-initiative dialog construction toolkit and engine. (right) Detailed data-driven architectural design of our natural language processing unit on left.

4. RESULTS AND CONTRIBUTIONS

4.1 Results

While ‘creating an actual dialog system involves a very intensive programming effort’ [13], our dialog authoring tool (see Fig. 3) is a contribution that simplifies that effort so that dialog designers can evaluate a variety of mixed-initiative, human-computer dialogs [5, 27]. Specifically, given $q$, the number of questions posed in a dialog, our system is capable of automatically implementing $2^q - 1$ dialog specifications (i.e., map an unsolicited utterance to the dialog prompt to which it is a response) to improve the natural language and mixed-initiative capabilities of systems like Siri. Fig. 4 (right) illustrates the design of the natural language processing unit in Fig. 3 (left). Fig. 5 demonstrates the natural language capabilities of our system. Our MPAA dialog toolkit is available at https://bitbucket.org/jwb_research/.

Figure 4: Illustration of the the low degree of natural language, mixed-initiative interaction (e.g., action-requesting, information-seeking, information-providing utterances) supported by systems such as Siri.

Figure 5: Demonstration of the enhancements over personal assistants like Siri that our approach fosters.
(i.e., combinations) of the prompts of the dialog. While the task structures for modeling dialog mentioned above, such as FSA s, CFG s, and events [12], are sound, and can be used to prove mathematical proprieties, tasks often need to be over-specified to model a rich and flexible form of human-computer interaction. Moreover, since dialogs can contain arbitrarily nested sub-dialogs, FSA s are less effective as general discourse structures [10]. Similarly, CFG s might be appropriate if the evolution of a dialog was something known a priori [10].

Evaluating models for mixed-initiative dialog is itself an unsolved problem for a variety of reasons including the extremely limited nature of existing data and the ambiguity of the very definition of initiative [13]. One way to capture the efficacy of a model is to evaluate how well the model fits data. In the context of our model, this means evaluating the frequency of dialogs that can be captured by our notation and how well it captures each. Given any value for $q$, the number of questions per episode, every dialog in the space $U_q$ can be specified using our dialog authoring notation. Since the specification expression of a dialog serves as a design pattern for implementing it, the number of sub-expressions in the specification is an evaluation metric for how well the notation captures the specification. A complete, mixed-initiative dialog can be captured by one expression: e.g., $\text{size blend cream}$. If we remove only one—(size blend cream)—of the thirteen episodes from this dialog, specifying it requires five sub-expressions: $\text{size blend cream} \cup \text{size blend cream} \cup \text{size blend cream} \cup \text{size blend cream} \cup \text{size blend cream}$. We specified each of the 8,191 dialogs in $U_3$, using our notation and computed the frequency that could be captured by 1, 2, ..., and 13 sub-expressions. Our results are shown in Fig. 6 (e.g., there are 46 dialogs that can be specified with one expression, and 2,977 that can be specified with four sub-expressions). For the details of our dialog-authoring notation (i.e., the fractional expressions above) based on functional programming language concepts, we refer the reader to [27].

Since there are no dialogs in $U_1$ that require greater than seven sub-expressions to model, and there are dialogs in the space with greater than seven episodes (e.g., the maximum number of episodes in any one dialog is thirteen for $q=3$), the use of our notation provides a compressed dialog specification. However, what is not illustrated in Fig. 6 is the number of episodes in each dialog that can be represented with a particular number of sub-expressions or, in other words, the magnitude of the results given in Fig. 6. For instance, if all 46 dialogs that can be represented with only one sub-expression only contain one episode, then there is no compression. To measure the efficacy of the compression, we computed the frequency of dialogs which can be specified at the observed compression percentages. For instance, 533 dialogs of the 8,191 could not be compressed at all (i.e., there is a one-to-one relation between the number of episodes and the number of sub-expressions). However, 1,197 dialogs can be compressed 33% (e.g., a dialog that involves nine episodes which can be specified with six sub-expressions), and 975 can be compressed 50%. Fig. 7 presents these compression results: over 20% of the dialogs (1,692/8,192) can be compressed 50% or more. While we cannot characterize the dialog specifications comprehensively beyond $q=3$ because it is not possible to enumerate and simulate all of them, we can say intuitively that the results for $q > 3$ are better than $q=3$ because the opportunities for compression increase as the number of questions posed in an episode increases. Therefore, both the number of sub-expressions required to specify a dialog as well as the percentage of dialogs being compressed to a high degree increase.

4.2 Contributions and Future Work

Dialog is essential in providing a rich form of human-computer interaction [8]. We summarize the contributions of our research as: we i) developed a language-based model for specifying and staging mixed-initiative, human-computer dialogs, ii) generalized and automated the activity of building a dialog system, and iii) evaluated its descriptive and staging capabilities by demonstrating that it can succinctly capture and stage a wide variety of dialogs, including those involving sub-dialogs. While “[c]reating an actual dialog system involves a very intensive programming effort” [13] and “complete automation in creating ... dialog applications remains an extremely difficult problem” [9], given a specification of a dialog in our dialog authoring notation, from among a variety of mixed-initiative dialogs, our system automates the implementation of the dialog. Designers of task-based dialog systems can use our dialog authoring notation and staging
engine as a dialog modeling and implementation toolkit to explore, prototype, and evaluate a variety of unsolicited reporting, mixed-initiative dialogs.

While the use of simulation for evaluation of dialog systems is common, the application of our results will benefit from a formal usability evaluation. We intend to conduct studies with users to evaluate the interface through which users experience the human-computer dialog (i.e., Figs. 2 and 5) as well as the interface for task modeling used by dialog designers to specify the dialog as part of future work. Evaluating the interface through which dialog participants experience the dialog will help us discern whether mixed-initiative dialogs resulting from our language-based model have desirable qualities (i.e., How effective and efficient are they? Does mixed-initiative dialog help the user in an information-seeking activity and how, e.g., time-to-task completion, satisfaction? For which types of dialogs or tasks is mixed-initiative interaction most effective?). We desire “computational agents carrying out our dialog theory to produce conversations with desirable qualities” [13]. To this end, we plan to conduct a study similar to [17] and, in a more broad context, using the results of [33].

For future work, we envisage the incorporation of mixed-initiative personal assistants designed/implemented with our toolkit into airport kiosks, ATMs, and interactive, voice-responsive systems, since the ubiquity of these platforms in a variety of service-oriented domains, such as education, healthcare, banking, and travel provide a fertile landscape for the use of our model for mixed-initiative interaction. We are also exploring the use of our model in a university course schedule application in an immersive, virtual environment. It will advance software development processes for virtual/cyberlearning environments, gaming, film, simulation, and telepresence, where mixed dialog flexibility is also critical.

5. REFERENCES


