Conversational Agents for Complex Collaborative Tasks

James Allen, Lucian Galescu, Choh Man Teng, Ian Perera

■ *Dialogue is a very active area of research currently, both in developing new computational techniques for robust dialogue systems and in the active fielding of commercial conversational assistants such as Apple's Siri and Amazon's Alexa. This article argues that, while current techniques can be used to design effective dialogue-based systems for very simple tasks, they are unlikely to generalize to conversational interfaces that enhance human ability to solve complex tasks by interacting with artificial intelligence reasoning and modeling systems. We explore some of the challenges of tackling such complex tasks and describe a dialogue model designed to meet these challenges. We illustrate our approach with examples of several implemented systems that use this framework.*

Even the most sophisticated current artificial intelligence (AI) systems cannot anticipate all the problem-solving scenarios to properly execute a solution without some human guidance. However, it becomes harder and harder for humans to provide this guidance as systems become more and more difficult to understand, control, and trust. For many black-box systems, the user has little control over the problem-solving strategies beyond parameter tuning, and the solutions are often given without grounds that are comprehensible to a human. In contrast, a system that could engage with humans in multimodal natural language dialogue, in which they jointly identify problems and refine, develop, and explore solutions, would constitute a major step forward toward expanding the repertoire of problems that can be (jointly) solved by leveraging both the human judgment and expertise and the system's computational and analytic capabilities, and gaining the user's trust that the solutions (jointly) arrived at are explainable and verifiable.

 ISSN 0738-4602

Most current dialogue systems only support interactions within very simple task models that can be represented as a short list of attributes. Once the values of these attributes are elicited, the task is essentially complete (see, for example, Williams et al. 2016 and the Dialogue State Tracking Challenge [DSTC][1]). Recent work has focused primarily on machine learning, motivated partly by the belief that machine learning leads to robustness and overcomes some of the brittleness found in earlier hand-engineered systems. However, the task models and dialogues supported remain very simple (for example, querying bus schedules, making restaurant reservations, booking plane tickets), and the task complexity has remained largely at the same level as that of systems developed decades ago, at least because since of the slot-filling dialogue managers described in Goddeau et al. (1996).[2]

We will demonstrate that it is possible to create robust conversational systems in much more complex domains. As opposed to the tasks of prior systems — which often can be solved by identifying speech acts and arguments for slot filling — tasks in complex domains may have no clearly defined solution or end state, and the reasoning required for both the user and the system cannot be articulated easily by a nonlinguistic user interface with predefined functions. In addition, these tasks cannot be solved by a standalone optimization process or a deep-learning algorithm, because in complex problem-solving the goal is often only vaguely outlined to begin with and becomes iteratively refined and modified as the problem-solving session progresses. Furthermore, even if such a black-box system could solve these tasks, the human user would have little trust in the results because the black box cannot provide explanations that the user can examine and understand. To tackle these tasks, we need systems that can provide humans with relevant information, understand their intentions within context, and integrate the human expertise and judgment to jointly explore potential solutions and identify desirable ones, providing transparent reasoning throughout the process. Such systems operate as active collaborators in the problem-solving process, interacting with the humans in the same ways that humans naturally use among themselves when they work together, both in terms of the mode of communication and the protocol implicit in negotiating a joint endeavor.

While the main ideas behind our collaborative problem-solving (CPS) approach are not new, it was only in the last few years that we integrated the natural language components and the CPS-based dialogue model into a fully domain-independent framework, thereby allowing third parties to bootstrap from these components to independently develop their own dialogue systems in a variety of domains. Some of the technical aspects of the CPS-based model are outlined in Galescu et al. (2018). In this article we consolidate our entire approach, using examples from multiple domains, to give the reader a better understanding of the capabilities and the broad applicability of our framework, in particular to tasks of complexity higher than can be handled by the predominant state of the art. Here we also extend the discussion of the key domain-independent capabilities and show in detail how the framework can be instantiated for specific applications. We also include an evaluation of our approach using several metrics.

Examples of Complex Collaborative Tasks

As motivation, we describe two example tasks that require a level of human-machine interaction and collaboration that is well beyond the capability of standard dialogue systems. For each domain we introduce our dialogue system implemented using the CPS framework. More example dialogues of complex tasks and CPS-based systems capable of supporting these tasks can be found on the Institute for Human and Machine Cognition (IHMC) website.[3]

Collaborative World Modeling

While researchers from a wide range of disciplines have developed complex simulation tools for exploring aspects of the world (for example, agriculture, economics, social stability, weather, hydrology), automated methods for combining such systems to answer larger questions about the world remain elusive. The state of the art in world modeling involves an extremely labor-intensive process, requiring person-years of effort by highly trained modelers to determine how a given scenario could correspond to a configuration of quantitative modeling engines, to identify or approximate the required data and parameter values needed, and then to actually run the resulting models over a set of scenario variations. For example, the Australian National Outlook (Hatfield-Dodds et al. 2015), which used mostly existing models for climate, land use, the economy, and other systems, took a large team of domain experts and scientists two years to produce. The Collaborative World Modeling System (CWMS) is an effort to build a collaborative conversational agent to assist humans in building such models. CWMS has to understand vague user goals, suggest and understand discussions about the problem-solving strategy, be able to plan and execute simulations, explore alternatives, and present and explain analyses. Figures 1 and 2 show a sample interaction with the current prototype system. More details can be found in Allen et al. (2018b) and Allen and Teng (2019).

To get a better feeling for how humans should be able to interact with AI systems, consider what happens in the dialogue in figure 1. It starts with the user suggesting a goal (1), which the system accepts and takes initiative to refine (2), and then suggests an initial course of action (4). To support this interaction, the system reasons that malnourishment is both an indicator of food insecurity and also that it is a value that can be computed by one of its reasoning engines. The system then constructs a plan to run a reasoning engine that can compute expected

1 U: *Can you analyze food insecurity in Sudan next year?*
2 S: *OK. Should we look at child malnourishment rates?*
3 U: *OK.*
4 S: *Should I compute a baseline estimate based on available data?*
5 U: *yes*
System identifies a reasoning engine that can estimate childhood malnourishment, and uses existing knowledge about Sudan (population, food availability, etc.) to estimate the number of malnourished children.
6 S: *OK. The percentage of malnourished children is about 39%.*
7 U: *What if 2019 is an El Nino year?*
System builds a plan to estimate the difference in crop yields in Sudan in El Nino years by interpolating simulation results from 30 years of prior weather data.
8 S: *I can estimate the effect with the following plan. Does that look good?*
9 U: *Yes*
10 S: *OK. The percentage of malnourished children increases by 5%.*
11 U: *What would happen if we change the planting date?*
12 S: *Shall I run simulations varying the planting date?*
13 U: *Yes*
System creates a plan to run a series of simulations that vary the planting date parameter, again estimating yields for an El Nino year by interpolating over results obtained from simulations of the past 30 years. The plan then assembles the results and presents to the user in a chart.
14 S: *The best scenario involves planting crops one week earlier than usual.*

Figure 1. A Dialogue between a User (U) and the CWMS System (S).

childhood malnutrition rates, based on baseline information on the expected availability of food and other variables. The answer computed is given in (6). The user then expresses a desire to elaborate the scenario based on predictions that there would be an El Niño event in the coming year. The system recognizes the user's intention is to run a new analysis with the new assumption. The change in crop yield is computed (figure 2) and fed through an economic model to compute the effect on food availability, and the change in food availability is then fed through a food security model to compute malnourishment rates.

The user follows up by asking about the effect of changing the planting date (11). The system knows a problem-solving strategy for exploring the effects of changing variables, and so asks whether it should construct a simulation experiment that estimates the crop yields for a range of different planting dates (12). Once the user concurs, the system builds and executes a plan for the experiment and shows a plot of the results to the user, identifying the best option. While this is as far as we have space to discuss it here, the user could easily continue on, for instance asking for elaboration (for example, *OK, and how would that affect the malnourishment rates?*) or exploring other options (for example, *What if we could increase the amount of fertilizer that is available?*) or pursuing some new strategy for dealing with the problem, such as shipping more food aid to the region.

Biocuration

In molecular biology, finding explanations for biologic observations and phenomena requires building and visualizing complex causal models with varying granularity and conceptual elements, and running simulations on these models to detect their dynamic properties. To manage this wide range of complex problem-solving behaviors, the Blabbing on Biocuration system (BoB) dialogue system integrates a variety of specialized agents with access to an extensive list of curated databases covering gene expression, protein activities, and molecular pathways, as well as knowledge extracted automatically from reading scientific publications in molecular biology (Allen et al. 2015; Gyori et al. 2017; Valenzuela-Escárcega et al. 2018).

Figure 3 shows an excerpt from an actual dialogue with BoB,[4] where the user is attempting to find support for the hypothesis that a drug has a particular effect on a gene that it does not target directly. BoB can

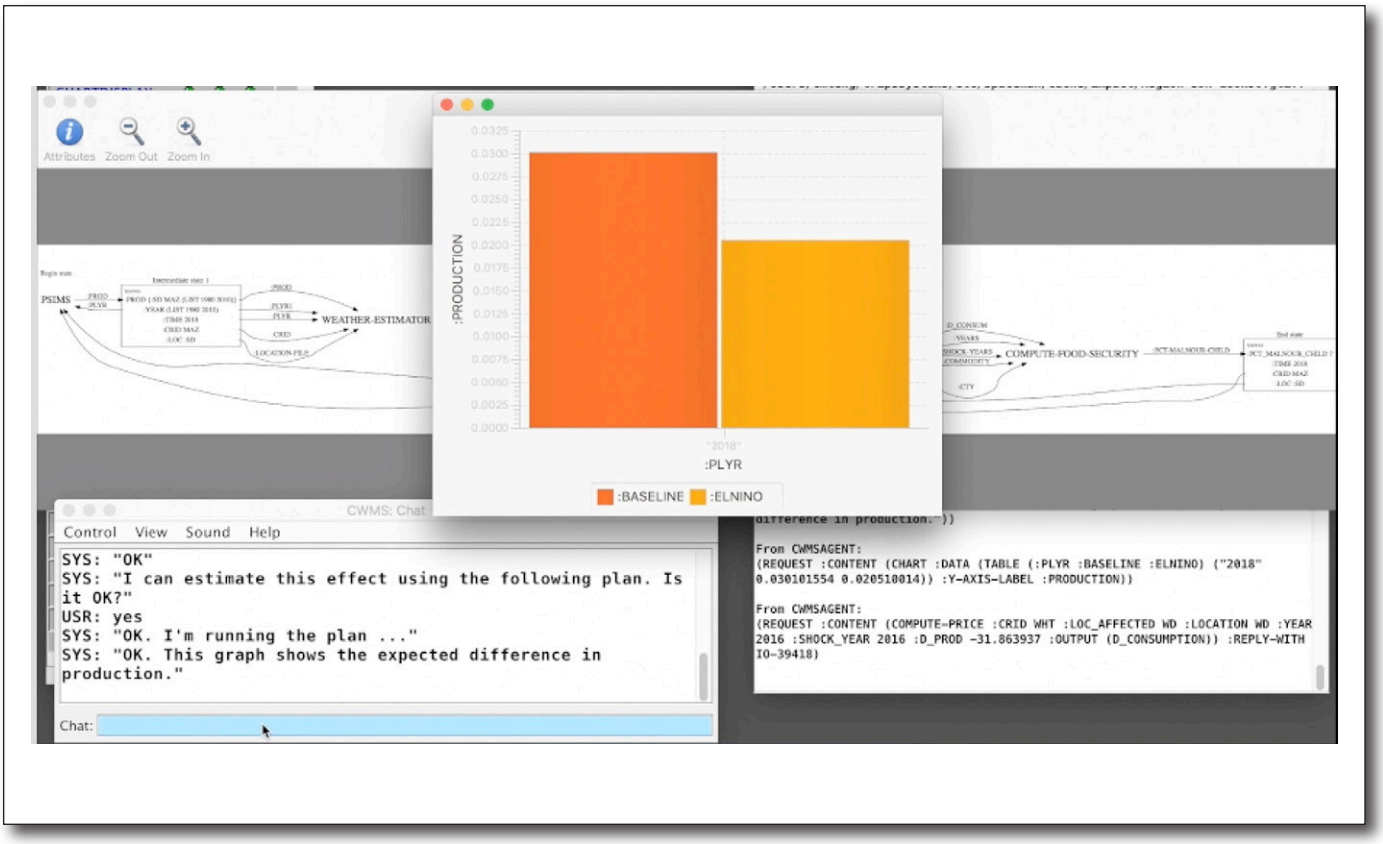


Figure 2. System Presents a Plan (in Background) Together with Results of Simulation of the Model.

The example shows the difference in expected crop production in an El Niño year.

help to identify possible pathways by which the drug can affect the desired gene. There are often multiple such pathways, so the mere fact that such a pathway exists is in itself useful, but typically this has insufficient explanatory power. BoB can provide additional help by assembling such pathways into molecular models, identifying missing pieces in the model, and making suggestions for revising the model, all in collaboration with the user. These models can be analyzed statically to find mechanistic support for the hypothesis. In addition, their behavior over time can be analyzed by running simulations. While the parameters for these simulations are generally set automatically, the user can explore what-if scenarios by asking the system to alter them and compare the outcomes of the simulations under different conditions. In addition to carrying out a dialogue with the user, the system also uses a graphical interface (figure 4) to display multiple views of the model being built as well as supporting evidence from databases and the literature, with hyperlinks that the user can follow to assess their reliability and usefulness in the biologic context of their problem.

Although CWMS and BoB involve substantially different types of AI reasoning systems and need to understand quite complex sentences expressing the user's intentions about different tasks, these two systems have been built on the same CPS framework, without having to develop separate language understanding and dialogue models for each domain. This shows that the CPS framework can serve as the basis for developing the next generation of conversational agents.

Background

In this section we review the current state of the art and show that significant extensions are needed to meet the challenges of complex domains such as world modeling and biocuration.

State-Based Dialogue Systems

Most current work in conversational agents is performed at essentially the same level of task complexity as systems dating back to at least the 1990s. To understand this notion of tasks, consider the representation used in the DSTC1, described in Williams et al. (2016). A dialogue system is formalized in terms of dialogue states, each consisting of a set of attributes (or slots) and their possible values organized as a frame. The focus of DSTC1 was providing information over the telephone about bus schedules.

1 U: *what proteins does SB52533 target?*
2 S: *The nominal target of SB525334 is TGFBR1.*
3 S: *Tell me what you want to do now.*
4 U: *Let's build a model*
5 S: *Okay.*
6 U: *SB525334 inhibits TGFBR1*
7 S: *Okay. I created a model where SB525334 inhibits TGFBR1.*
8 U: *what transcription factors regulate SMURF2?*
9 S: *I found three transcription factors that may regulate SMURF2: PBRM1, SMAD2, and TBL1XR1.*
10 U: *does TGFBR1 activate SMAD2?*
11 S: *Overall, I found that TGFBR1 can activate SMAD2.*
12 S: *I found 5 relations to support my conclusion. Look in the provenance tab to see a few examples of what I found.*
13 U: *TGFBR1 activates SMAD2*
14 S: *Okay.*
15 S: *It looks like a required activity is missing, consider revising to* Active TGFBR1 activates SMAD2.
16 S: *I added the fact that TGFBR1 activates SMAD2 to the model.*
17 U: *Active TGFBR1 activates SMAD2 and active SMAD2 increases the amount of SMURF2*
18 S: *Okay.* […]
19 U: *does SB525334 decrease SMURF2 in the model?*
20 S: *Yes, it is decreased.*

Figure 3. Excerpt from a Dialogue between a User (U) and the BoB System (S).

The user is working with BoB to build a model of how the drug SB52533 decreases the amount of SMURF2.

The slots that comprise a dialogue state are shown in figure 5, reproduced from Williams et al. (2016). The state of the conversation so far is captured by the values of the slots that have been instantiated. For instance, if the user said *I need to take a bus from City Hall tomorrow* then the dialogue state would have the TIME slot set to tomorrow's date (obviously relative to the time the conversation is occurring) and the ORIGIN POINT OF INTEREST would be set to City Hall. In the early conversational systems using this model (for example, Goddeau et al. 1996), the system would then examine the dialogue state to find an attribute that still needs to be filled. In this case, the system mainly needs to identify the destination to find appropriate bus routes. This would motivate a system response such as *Where do you want to go?* Once this question has been answered, the system would make an application-program-interface call to a bus-schedule database to identify the routes and present them to the user.

A significant advantage of such frame-based systems is that they enable robust semantic parsing (for example, using pattern matching or a trained neural network). The relevant frame significantly restricts the space of possible interpretations and parsing can be reduced to essentially a keyword/named-entity recognition task. For example, no matter the surrounding context, if a fragment that can denote a day (for example, *Saturday*, *tomorrow*, …) is found, then it is used to fill the DATE slot. Likewise, time expressions (for example, *3 PM*, *in the afternoon*, …) are used to fill the TIME slot. A simple grammar of addresses can recognize phrases to fill the DESTINATION_STREET slot. With a handful of special patterns, the frame can be instantiated even from ungrammatical or nonsensical sentences (whether initially misspoken or resulting from speech recognition errors).

Much of the research in dialogue systems in the past two decades has focused on creating better dialogue management strategies. A clear step forward was to track multiple possible dialogue states rather than a single one (Pulman 1997). As a simple example, consider an utterance that resulted in two hypotheses from the speech recognizer: the user said *City Hall Today* or *Seaton Mall Today*. Both locations were recognized as possible points of interest and could either fill the ORIGIN POINT OF INTEREST

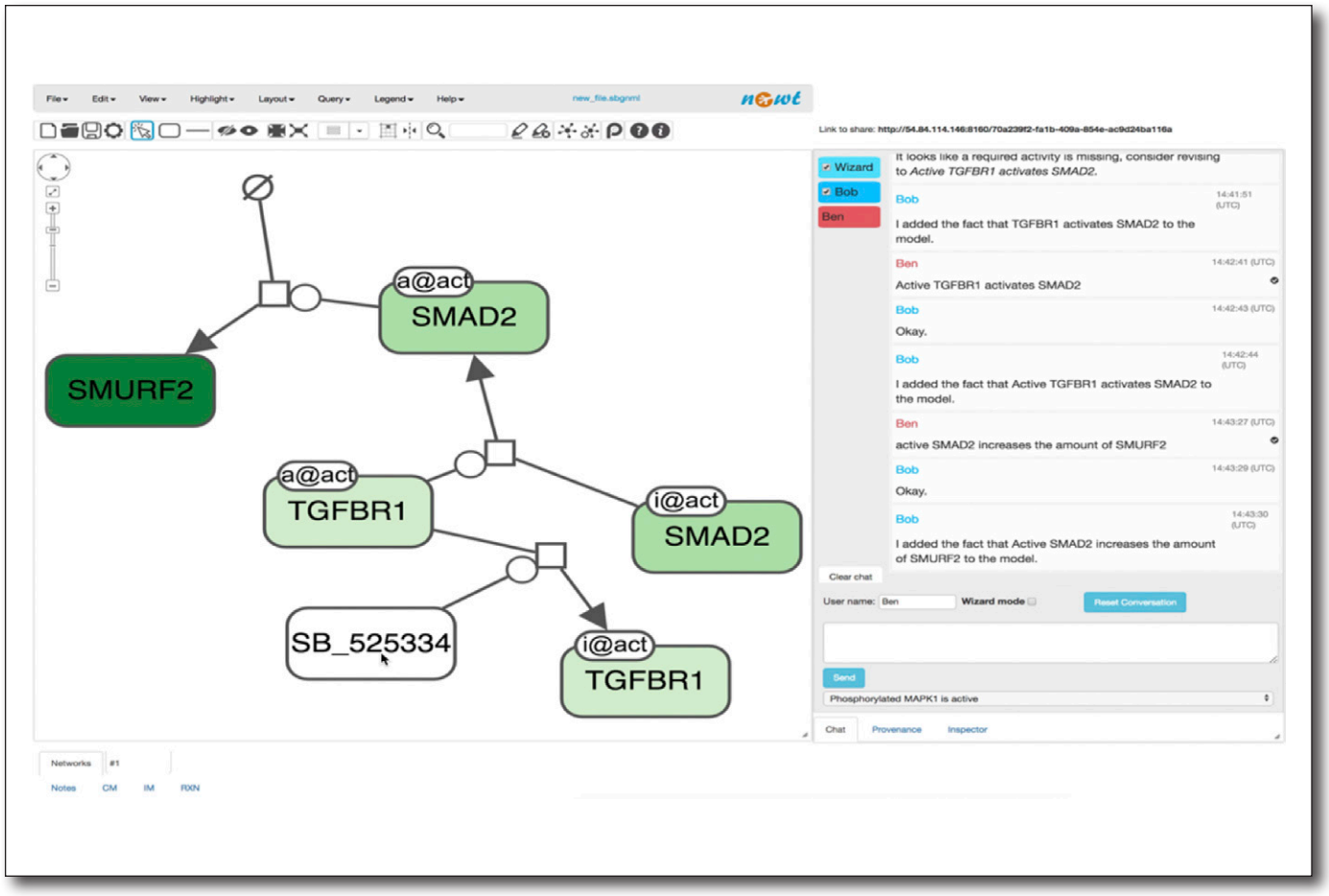


Figure 4. Screenshot of BoB's Interface.

The right panel is a chat-style language interface, and the left panel shows the current model. The user has access to other views with additional information.

or the DESTINATION POINT OF INTEREST slots in the frame. Assuming the speech recognition slightly preferred City Hall and destinations are more commonly stated before origins, the system can compute a distribution over the four possible dialogue states. See Kim et al. (2008) and Williams et al. (2016) for good examples of systems following this approach.

At this stage, the dialogue manager would need to decide between a number of plausible continuations: Ask user to confirm top speech recognition hypothesis (for example, *Did you say City Hall?*), ask user to confirm an entire interpretation (for example, *You want to go to City Hall?*), ask user to identify/confirm a slot (for example, *Where are you going to?*), or ask user to repeat the statement (for example, *I didn't understand. Would you repeat that?*). In any of these situations, when the user provides the next response, the possible interpretations of that utterance would be combined with the current context to provide updated probabilities over the possible states.

The earliest frame-based systems, to determine the next step, used hand-constructed rules (for example, Zue et al. 2000; Larsson and Traum 2000), and some of those dialogue managers have performed well in DSTC (for example, Wang and Lemon 2013). Much of the effort over the past decade, however, has focused on using machine learning to learn dialogue management strategies. There is a long history of using Partially Observable Markov Decision Process models (see the excellent review in Young et al. 2013) and more recently also neural net approaches (for example, Serban et al. 2016). These techniques generally require a large corpus of sample dialogues in the domain, ideally annotated with the correct current dialogue state (represented, for example, as a partially instantiated frame with the currently known values).

While we do not know the details of the implementations of commercial conversational assistants, such as Apple's Siri, Google's Google Assistant, and Amazon's Alexa, it is safe to say that they represent the conversational state using something equivalent in expressive power to the frame representations. Each task (for example, set an alarm, find a restaurant, navigation) has an associated set of information

Slot	Possible Values
BUS ROUTE	Route Numbers (e.g., 1 – 100)
DATE	Calendar date
TIME	Time of day
ORIGIN STREET	Street Address
ORIGIN NEIGHBORHOOD	Neighborhood name
ORIGIN POINT OF INTEREST	Name of distinguished location (such as museum, city hall, …)
DESTINATION STREET	Street Address
DESTINATION NEIGHBORHOOD	Neighborhood name
DESTINATION POINT OF INTEREST	Name of distinguished location

Figure 5. The Template for a Dialogue State in DSTC1.

that needs to be acquired, and once that information has been collected an application-program-interface call is made to the back-end application that supplies the functionality.

There are some generalizations of frame-based systems that extend the range and complexity of conversations they can support. For instance, the system may accommodate multiple tasks, each represented as a separate frame. To identify the intended task and frame from the first utterance (or when the user switches to a different task), the system associates with each frame keywords and phrases that are useful indicators for the frame, and checks which frames allow the extraction of the most information from the given sentence. For example, a sentence that mentions the word "bus" would be a strong indicator that the utterance should be interpreted with respect to the frame in figure 5, rather than, for example, another frame that involves making a restaurant reservation. Many neutral network-based approaches use joint models for simultaneously predicting the intent (task frame) and the slot fillers (for example, Liu and Lane 2016; Zhang and Wang 2016; Wang, Shen, and Jin 2018). Recently DSTC included a track on developing multitask dialogue systems (Li et al. 2020), although there have been earlier forays into this area (for example, Mrkšić et al. 2015; Wen et al. 2017; Nouri and Hosseini-Asl 2018). While tackling multitask dialogues does pose additional challenges compared with handling single-task dialogues, the increase in task complexity is fairly minimal.

Another generalization is to represent a task as a series or transition network of frames, where once all the information in one frame is acquired, the system can transition to a new state (with a new frame). This model is captured in the industry standard VoiceXML[5] and can be used to develop commercial systems such as the customer service systems one encounters over the telephone. In practice, these systems are typically driven by system prompts, that is, the system asks a question or provides options for the user to select, and then moves on to the next state based on the answer. Such systems are often referred to as *system-initiative*. In contrast, the frame-based systems described above are typically *user-initiative*, that is, the user initiates the conversation as opposed to answering system queries.

Currently, most work on modeling dialogue is based on neural net models for all or most of the system components (for example, Wen et al. 2017; Lin et al. 2019). These systems are trained on transcripts of dialogues, with or without annotation of the dialogue states (that is, frames). They then attempt to generate the next turn in a dialogue, given the complete dialogue history up to that turn. Most such end-to-end systems are evaluated on datasets of complete dialogues, either generated automatically or crowdsourced, but are never put to a real test with human users (some evaluation schemes use simulated users). DSTC8 (Li et al. 2020) introduced a form of human evaluation via Amazon's Mechanical Turk, where users converse with the system to achieve a given task. However, unlike traditional user experiments, this type of evaluation suffers from a significant weakness: The task is described in detail, step by step, in natural language, which users can follow almost exactly to conduct a reasonably successful conversation.

Of note, while there has been a large effort in developing more robust and effective state-based dialogue management techniques, there has not been much effort in developing systems for more complex tasks. As we will discuss below, as task models become more complex, there are significant

hurdles to overcome. First, relatively simple parsing techniques and methods of information extraction (for example, pattern recognition for slot filling and intent detection) start to lose their effectiveness as the domain of discourse becomes larger. Second, both probabilistic and neural network-based frameworks rely on having a fairly simple notion of state, and a relatively simple set of choices that can be made as the system's contribution. Both of these spaces increase substantially with the increase in task complexity. In addition, related to this second point, the more complex the dialogue states, the larger the corpora that need to be collected and, for some approaches, annotated to train the probabilistic models.

Task-Based Dialogue Systems

While state-based systems dominate the literature, there has been a steady development of conversational systems supporting more complex task models. These systems explicitly model the task being performed using hierarchical task representations, and engage the user in more complex, longer-term tasks such as tutoring, collaborative planning and control, and task learning. The notion that the structure of the dialogue reflects the structure of the underlying task was noted early on by Grosz (1974). As we will see later, this is not strictly a one-to-one correspondence, but the task structure does provide significant insight into the dialogue structure. The more complex relationship between the two was elaborated in Grosz and Sidner (1986).

A number of dialogue systems are driven directly from task models. For instance, STEVE teaches students about physical tasks using a virtual environment (Rickel and Johnson 1998). COLLAGEN (Rich and Sidner 1998) and, more recently, Disco (Rich and Sidner 2012), provide general frameworks for building systems that engage in collaborative conversation based on a library of explicit tasks (not tied to any specific domain). Another task-based model that supports multitasking in dialogue is described in Lemon et al. (2002).

Let us examine a task-based dialogue system in more detail. RavenClaw (Bohus and Rudnicky 2009) is driven by a hierarchical task model such as the one shown in figure 6 for booking rooms for meetings. The task (ROOMLINE) consists of four subtasks: logging in, in which the system welcomes the user and asks if the user is already registered and asks for their name; obtaining specifics of the reservation, including location, time and other information such as room size; querying to the back-end reservation system; and presenting and discussing the results. The dialogue engine is task-independent and includes a number of generic conversational skills, including language interpretation, response generation, clarification requests, and error detection and management. The system manages the dialogue by moving through the task tree from subtask to subtask as each is completed. Each task is executed based on its specification, which may include the following general behaviors:

- Giving the user some information (for example, the WELCOME task involves saying *Welcome*)
- Requesting information from the user (for example, ASKREGISTERED involves asking if the user is registered; DATETIME involves asking the user for the desired time of the meeting)
- Making a call to a back-end reasoner (for example, GETRESULTS involves querying the room reservation database)

As in frame-based systems, each task has a set of slots and semantic patterns for interpreting values for that slot. For example, the LOGIN task has a slot for the username, while the ASKREGISTERED task has a Boolean slot indicating whether the user is registered.

The dialogue manager operates by executing tasks in the task tree, maintaining a stack of active tasks, much like the attentional stack in Grosz and Sidner (1986). To execute the current task, the system either: performs one of the actions on the stack and, if appropriate, waits for a response; or, if there are no pending actions, pushes a subtask onto the stack and then executes that subtask. For instance, after the user logs in, the system asks when the user wants a room (subtask DATETIME). At that time, there are three tasks on the stack: The top is DATETIME, with a slot for reservation time. Below that is the task GETQUERY, with a time slot which it shares with DATETIME, plus the location slot and other slots from its other subtasks. At the bottom of the stack is the root goal, ROOMLINE. In interpreting the answer, all the slots on the stack are candidates to be filled. For instance, if the user said *Gavett Hall at 3PM Tuesday* in answer to the time question, the system would fill in not only the requested TIME slot, but also the LOCATION slot even though it has not asked about this (yet). This architecture thus supports the same robust parsing as phrase/keyword spotting in frame-based systems.

Task-based models can engage in dialogues over tasks more complex than can be easily modeled in state-based systems. In addition, because of the explicit task model the system has a representation of what its current goals and intentions are (so, in principle, it could answer questions about what it is doing). These models work well in instructional environments where the system designer essentially lays out a concrete lesson plan, or in applications where there are well-defined tasks to be accomplished, such as in a room reservation or a travel agent domain. However, task-based dialogue models are relatively underrepresented in current research because domains where they are most effective are typically too complex to construct a sizable corpus, and thus are not amenable to machine learning approaches.

Dialogue Systems Based on CPS

While task-based models support more complex dialogues than state-based models, they still fall short for

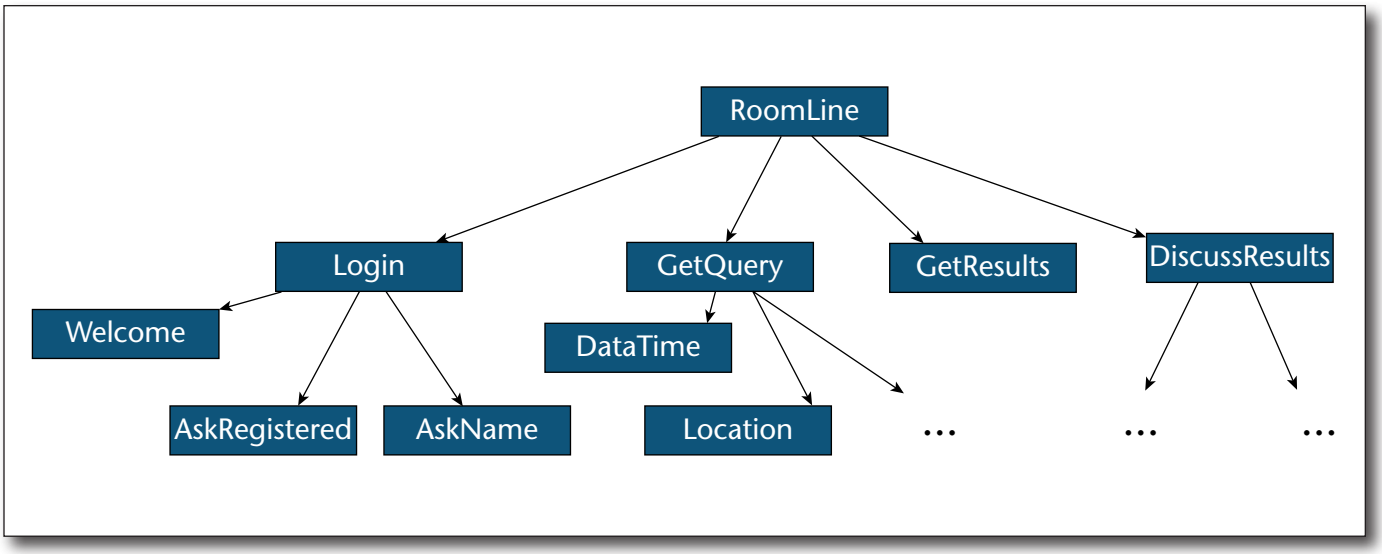


Figure 6. A Task Tree for a Room Reservation System.

Reprinted in part from Figure 3 in Bohus and Rudnicky (2009), with permission from Elsevier B.V.

a wide range of applications involving agent interactions. Consider an example of an actual human–human dialogue collected by one of the authors (figure 7). Participant A was trying to mail a letter to Mexico. Participant B was an administrator in the office. While it would be easy to build a dedicated state- or task-based system to handle the task of mailing a letter, this dialogue does not track any predefined task and none of the approaches discussed above could explicitly model the discussion and modification of the goal and subsequent novel solutions. (In fact, most of the conversation was to define exactly what the goal was, not developing or performing a task!)

A system participating in this conversation needs to be able to generate new tasks on demand, thus requiring reasoning capabilities similar to AI planners (for example, Ghallab, Nau and Traverso 2004), which can generate novel task models by combining operators from a plan library. In addition, building mutually agreeable plans requires intention recognition (for example, Allen and Perrault 1980; Kautz and Allen 1986) and mixed-initiative planning (Ferguson and Allen 2007).

Early work on SharedPlans (Grosz and Kraus 1996; Lochbaum et al. 1990) and plan-based dialogue systems (for example, Allen and Perrault 1980) laid good theoretical foundations for such systems. A core underlying principle was that communication acts can be formalized and reasoned about in terms of their effects on beliefs and goals of the dialogue participants. Perhaps the best developed formalism is described by Cohen and Levesque (1990). However, while there were some interesting demonstrations, these approaches have not been effective in building robust dialogue systems in practice. It is just too difficult to account for all the different actions that could occur in a dialogue from first principles alone.

When agents are engaged in solving problems together, they need to communicate to agree on what goals to pursue, what steps to take to achieve those goals, and to negotiate roles, resources, and more. In other words, the dialogue agents must take into account the nature of the joint activity itself. We call this *collaborative problem solving*, or *CPS*. Examples of early systems that took this approach include Chu-Carrol and Carberry (1998) and Litman and Allen (1987). In the early 2000s, Allen and colleagues described a preliminary plan-based CPS model of dialogue based on an analysis of an agent's collaborative behavior at various levels (Allen et al. 2002). A dialogue system based on this model is described in Blaylock and Allen (2005). Even earlier systems, such as the original The Rochester Interactive Planning System (Ferguson and Allen 1998; Allen et al. 2000), operated using similar intuitions but implemented the behaviors directly, without an explicit CPS model.

The CPS model consists of the following four conceptual levels: *An individual problem-solving level,* where each agent manages its own problem-solving state, and plans and executes individual actions. This level includes the AI systems that implement the functionality of the specific domain, as well as the overall management of these systems; *A CPS level,* which models and manages the joint or CPS state (shared goals, resources, situations) and is independent of any specific domain; *An interaction level,* where individual agents negotiate changes in the joint problem-solving state, independent of the particular domain; and *A communication level,* where language and/or other forms of communication realize the interaction level acts. While this may be domain-specific, in our systems we use generic semantic parsing and interpretation that applies

	Utterance	Collaborative Act	Problem-Solving Status
A0			Private Goal: mail letter to Mexico
A1	Do you know what first-class postage to Mexico is?	PROPOSE ADOPT GOAL [Know postage rate]	Goal 1 [Know postage rate] is proposed as Shared Goal
B2	No.	REJECT [B does not have the knowledge to accomplish goal]	Proposed goal rejected. But A still has private goal.
A3	How do I find out?	PROPOSE GOAL [build plan to know postage rate]	Goal 2[create plan …] is proposed as Shared Goal
B4	Do you really want to know, or do you just want to mail the letter?	REQUEST CLARIFY GOAL [Know postage or mail letter]	Goal 3 [identify goal …] is proposed by B
A5	Mail the letter.	IDENTIFY GOAL [Mail Letter]	Goal 3 satisfied. Goal 4 [mail letter] proposed
B6	No problem, we'll put this charge slip on it and the post office will figure it out	PROPOSE SOLUTION [have post office do it]	Goal 4 accepted as shared. Solution is proposed.
A7	Great, thanks. Here it is.	ACCEPT SOLUTION	Solution is Shared. Solution is executed.

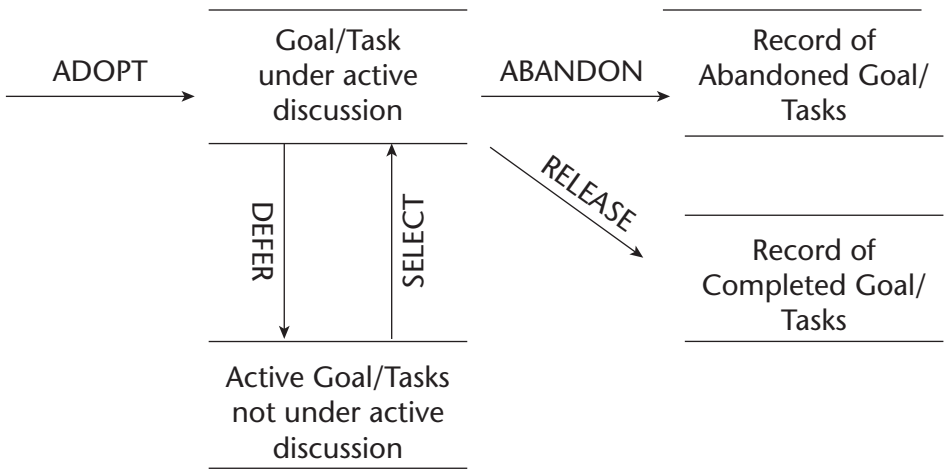
Figure 7. Collaborative Dialogue in Action.

to any domain. For natural language generation (NLG), communication acts are in terms of standard, domain-independent speech acts, whose content is expected to be domain-specific.

Consider first the general structure of a problem-solving state, either the collaborative problem-solving state established in the dialogue, or the individual problem-solving state of a single agent. Figure 8 shows the management of goals and ways to achieve goals (that is, tasks and solutions). We see there an encoding of the life cycle of goals and solutions: a new goal/solution may be adopted (ADOPT); an existing goal/solution may be focused on (SELECT) for pursuing in the subsequent dialogue; the current goal/solution may be deferred for now, possibly to be resumed later (DEFER); a goal/solution may be abandoned (ABANDON); or may be accomplished and dismissed (RELEASE). When problem-solving acts are used to interpret the behavior of a single agent's reasoning, it is a characterization of key parts of an intelligent agent's behavior. For instance, I might describe my behavior as follows: I decided to buy a rib-eye steak for dinner (ADOPT a goal), but after I found out how expensive it was I decided to buy a hamburger instead (ABANDON then ADOPT a new goal).

For shared goals in a collaborative situation, these problem-solving acts can only be accomplished via communication. In other words, the two agents have to agree before something becomes shared. Consider again the dialogue in figure 7. Before the conversation, agent A has the goal to mail the letter (A0). Utterance A1 attempts to introduce a shared goal to establish the price of the postage to Mexico, which agent A believes would allow successful completion of the goal (a proposal to an ADOPT GOAL act). Agent B does not accept the proposed goal because they do not have the information to accomplish this goal (B2). In response, agent A proposes a related goal, namely, to find a method to determine the postage (utterance A3). Agent B does not address the request directly but asks what the exact goal is (utterance B4). Thus, we describe B4 as a REQUEST CLARIFY GOAL. Agent A answers this question, which in this context is interpreted as identifying a new goal (IDENTIFY GOAL). With a joint goal finally established, agent B suggests the simple solution of placing a charge slip on the letter for the post office. Utterance B6 is interpreted as both an implicit acceptance and a proposal of a solution to the agreed-upon goal (PROPOSE SOLUTION). In response, agent A accepts the proposed solution (utterance A7). The dialogue ends with the accomplishment of the joint goal as agent A hands over the letter.

The way the individual problem-solving state is implemented and managed is idiosyncratic to each



Topic	ACT	Gloss	Example Sentence
Goals, Solutions	ADOPT	Introducing a goal	Let's plan food aid for Sudan.
	SELECT	Focusing on existing goal	Let's return to the food problem.
	DEFER	Temporarily putting a goal aside	Let's work on this later.
	ABANDON	Abandoning a goal	I don't care about this anymore.
	RELEASE	Completing a goal	We're done.

Figure 8. Life Cycle and Key Acts of a Problem-Solving Goal.

application domain and typically involves specialized reasoning engines to execute the actual tasks. In the CPS framework, the individual problem-solving state is managed by a component called the *behavioral agent* (BA). For example, the BA in CWMS encodes modeling tasks that create and execute simulations to support activities such as intervention planning and prediction. These tasks can then be executed by invoking back-end specialized reasoning engines, such as a crop modeling system for agricultural simulations. In the BoB system the BA coordinates the activities of many knowledge sources and reasoning engines, for building and reasoning with mechanistic molecular models, simulation and analysis of dynamic molecular models, pathway analysis, and specialized database lookups.

A key insight is that the collaborative problem-solving state can be task- or domain-independent and implemented in a general fashion, given a suitable interface to the BA, which will perform the domain-specific reasoning at the individual problem-solving level. The domain-independent dialogue manager coordinates the interpretation of the dialogue, interacting with the BA as needed. It bridges the divide between how humans interact when problem solving, and how the back-end systems perform the problem-solving processes for the domain (see figure 9).

Challenges for Complex Dialogue Systems

To build dialogue systems for complex tasks, we face a number of challenges: the language is relatively unconstrained; the exact nature of the tasks cannot be anticipated and coded in advance; and the system behavior cannot be characterized by static policies. We will discuss how these considerations impact slot-filling and state/task-based systems and the CPS systems.

Language Understanding

The slot-filling approach to language understanding allows robust interpretation of sentences even in the presence of speech recognition errors and ungrammatical utterances. It is highly limited, however, as it is based on predefined (by explicit definition or by learning) extraction patterns that are associated with each slot. This works fine for slots such as TIME

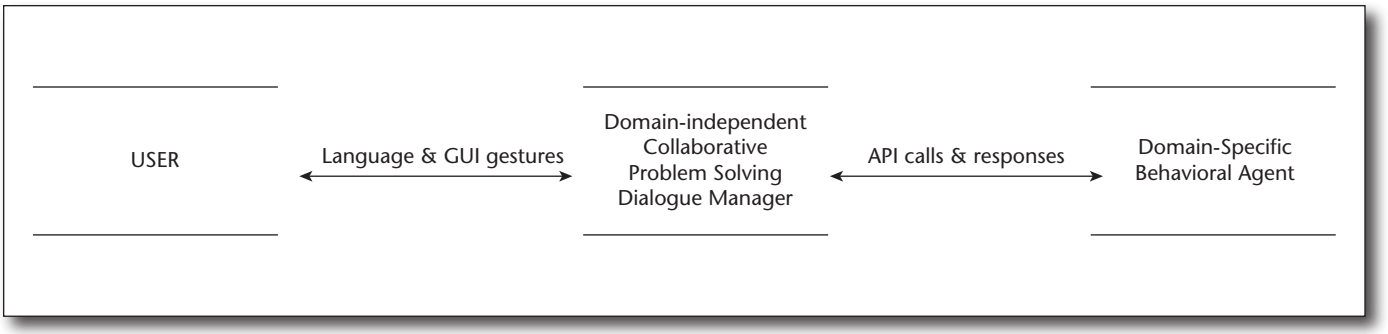


Figure 9. The CPS Model Bridges the Divide between Intuitive Human Behavior and Specific AI Reasoners.

OF DAY, RESTAURANT TYPE (for example, Italian, Lebanese), or MODE OF TRANSPORT (for example, bus, car, train) that are encoded as part of a state/task model. But it is unclear how such techniques could be used to handle most of the utterances in the CPS dialogues we presented. For instance, what slots in a task could match utterances such as *What if 2019 is an El Niño year?* Even if we had a task model with YEAR and WEATHER slots, filling the year slot with 2019 and the weather slot with El Niño would not capture the intent, which is to compare the results of the current analysis to one where a variable (the weather condition) has been changed (constrained to the presence of El Niño events). Or, *Consider what transcription factors are phosphorylated by MAP3K7 and regulate CXCL8?*, which involves a conjunction of complex event descriptions. Cohen (2019) presents many examples of utterances in other domains whose content cannot be easily captured as a collection of slot values, such as *What is the closest parking lot to the Japanese restaurant nearest the Space Needle?* We clearly need compositional semantic parsing that captures complex relations between objects, as well as complex objects such as events and nominals with relative clauses.

Specifying Task Models

Tasks are often not well defined at the start and have to be constructed incrementally during the dialogue itself. Consider a relatively basic transportation planning domain, namely the original TRIPS system (Ferguson and Allen 1998), in which a human and system collaboratively build a schedule of transport actions to accomplish some goal (for example, *Use truck 3 to get the people in the city Abyss, then go to Bath and get the people there. Meanwhile, use the helicopter to get the people in Calypso*). Here the task is constructed in the dialogue and then executed and monitored and may be revised in subsequent dialogue if the situation changes. The actual task is arbitrarily complex, and it is not feasible to enumerate all possibilities in advance. Rather, the dialogue interactions can be characterized by metalevel actions, for example, add this step to the plan, change a parameter value (for example, the vehicle) in this planned action, replace this action in the plan with a different action. Furthermore, each turn in the dialogue has to be interpreted in the context of the transportation plan built so far. An utterance such as *Let's send the helicopter to Bath instead* could refer to many different possible modifications to the plan (for example, *Instead of sending it to Calypso? Instead of sending a truck to Bath? Instead of taking the people back from Calypso directly?*).

Determining System Behavior

In the state/task-based dialogue models, the range of system actions is quite limited, typically consisting of a few actions: asking the user for a slot value, performing a clarification or confirmation, or performing a back-end application-program-interface call. In the CPS domains, the possible system actions are the result of a complex problem-solving process, where the system needs to recognize the user's intention, system planning may occur on the basis of this intention, and problems may arise in planning that need to be resolved. While the range of possible actions can be enumerated at the meta-level, as we discussed, the actual actions are essentially unlimited given they could be based on any possible aspect of any possible plan that can be constructed.

A Framework for Collaborative Problem-Solving Systems

One of the main goals of our recent work has been to create tools for generic linguistic interpretation and intention recognition, and to provide a dialogue shell independent of domain-specific problem solving. The domain-specific BA then instantiates the higher-level intentions into concrete problem-solving actions and verifies that such actions make sense in the domain context. As a consequence, in the CPS model the back-end problem solvers are relatively insulated from the need to worry about linguistic issues of sentence understanding as well as discourse and dialogue management.

We will describe the CPS framework in more detail in this section and return to the problems of natural language understanding in complex tasks in the next section.

Operations on the Problem-Solving State

The interaction level consists of an interaction speech act where the content of the act is an operation on the problem-solving state. As the simplest example, a shared goal can be established between agents A and B if A proposes a goal and B accepts it.

(1) PROPOSE_A ADOPT G1 :as (GOAL)
Let's analyze the food security situation in Sudan next year

(2) ACCEPT_B ADOPT G1 :as (GOAL)
OK

There are two key parts: the communicative act (for example, PROPOSE, ACCEPT), and the interaction act (for example, ADOPT), which identifies what action the communicative act is attempting to perform on the collaborative state. The main communication acts are shown in table 1. These acts are augmented by the specific operation on the CPS state that is being proposed or accepted, for example, a new top-level GOAL in the above. One might also suggest refining a current goal. For example, the second part of utterance 2 in figure 1 proposes adopting G2 (looking at malnourishment rates) as a subgoal to goal G1 (analyzing food security), formalized as follows:

(3a) PROPOSE_B ADOPT G2 :as (SUBGOAL :of G1)
Shall we look at child malnourishment rates?

Another key relation involves refining or changing an existing goal. This commonly occurs during clarifications. For instance, the dialogue might have continued as follows:

(3b) PROPOSE_A ADOPT M1 :as (MODIFICATION : OF G1)
Focus on the eastern part of the country

In this case, agent A refined the goal to a more specific region to be analyzed. Once agent B accepts this, the shared goal will be updated. Table 2 shows the different relations between the new act and the existing CPS state.

Managing Domain-Specific Intentions: The EVALUATE-COMMIT Cycle

The CPS manager interprets and drives the interactions that embody the collaborative problem-solving negotiation between the user and the system (that is, the interaction level in the above discussion). This cannot be done accurately without an ability to reason about the domain-specific intentions as well. For instance, the sentence *Can you analyze food insecurity in Sudan next year* in figure 1, after appropriate semantic parsing, could be identified as likely to be a PROPOSE of a new top-level goal. This hypothesis can be derived solely based on the current problem-solving context (no goal has been agreed to yet, this being the first utterance) and the form of the speech act (a REQUEST), but it cannot be confirmed without checking that analyzing food insecurity is a reasonable collaborative goal in the current context, using domain-specific knowledge and reasoning.

To make the system as domain-independent as possible, the CPS manager generates a ranked list of candidate CPS acts based on general knowledge, and requests the BA to evaluate the likelihood of each in turn given its domain-specific knowledge about the current problem-solving state. If the BA deems a hypothesis acceptable, the CPS manager commits to the act and changes the CPS model, thereby identifying what the system believes was the intended interpretation. This interchange can be formalized as follows, where G1 is the hypothesized goal derived from a user utterance:

CPS Manager → BA :
$$\left(\text{EVALUATE} : \text{content} \left(\text{ADOPT G1} : \text{as} \left(\text{GOAL}\right)\right)\right)$$

BA → CPS Manager :
$$\left(\text{ACCEPTABLE} : \text{what} \left(\text{ADOPT G1} : \text{as} \left(\text{GOAL}\right)\right)\right)$$

CPS Manager → BA :
$$\left(\text{COMMIT} : \text{content} \left(\text{ADOPT G1} : \text{as} \left(\text{GOAL}\right)\right)\right)$$

The CPS manager requests that the BA evaluate a hypothesis about the CPS act of adopting G1 as a goal. On receiving from the BA that it deems the hypothesis ACCEPTABLE, the CPS manager commits to the interpretation (and issues a confirmation to the user). Once the acceptance is generated, G1 becomes a shared joint goal for both the system and the user.

On the other hand, the BA might not find the hypothesis acceptable. If the BA cannot infer an appropriate intention underlying an utterance, it would respond with FAILURE. If the BA can identify the intention but refuses (or is unable) to agree to it, it would respond with UNACCEPTABLE, with an optional reason: In systems we have implemented so far the most common reason is that the agent cannot perform the requested task or action because there are not sufficient resources.

With a FAILURE, the CPS manager can suggest an alternative from its list of candidates and the BA will evaluate its appropriateness. This EVALUATE-COMMIT cycle (figure 10) is critical for enabling intention recognition that exploits both the linguistic context (that is, the exact phrasing of utterances and the discourse context) and the domain-specific problem-solving context.

In certain cases, the structure of the utterance and the problem-solving context might not be sufficient for the CPS manager to identify the problem-solving intention. The CPS manager can then send an underspecified intention to the BA and have it identify the intention. For instance, consider a dialogue in a

Act	Typical Use	Example (Blocks World)	Example (BoB)
PROPOSE	Suggest a modification to the CPS	*Let's build a tall tower.*	*I want to find out what activates Ras.*
ACCEPT	Accept a modification to the CPS	*OK*	*Sure.*
REJECT	Reject a modification to the CPS	*No*	*I don't want to do that.*
ASSERTION	Make a claim about the state of world	*We do not have any blue blocks.*	*Ras activates Raf.*
ASK-IF	Ask whether something is true about the world or the CPS state	*Are there any red blocks left?*	*Does SB525334 decrease SMURF2 in the model?*
ASK-WHAT-IS	Ask for some object/value to be identified	*How tall a tower?*	*What proteins does SB52533 target?*
ANSWER	An assertion issued in response to a question (including yes/no)	*Five blocks* (answer to above question)	*The nominal target of SB525334 is TGFBR1.* (answer to above question)

Table 1. The Main Communicative Acts in CPS.

Communicative Act	Relation Name	Relation to Current CPS State
PROPOSE	GOAL	Proposed as a new top-level goal (that is, pursued for its own sake)
	SUBGOAL :of G	Proposed as a subgoal of an existing goal G
	MODIFICATION :of G	Proposed as a modification of an existing goal G
ASSERTION	CONTRIBUTES-TO :goal G	Asserted in relation to the specified goal G
ASK-IF ASK-WHAT-IS	QUERY-IN-CONTEXT :goal G	Query is in the context of goal G

Table 2. Different Relations of a New Act to the Current CPS State.

collaborative blocks-world task where either the user or the system can manipulate the blocks:

User: *Let's build a tower.*
System: *OK*
User: *I will move the blocks.*

Without specific knowledge of the domain, the CPS manager might not be able to determine the intention behind the assertion *I will move the blocks*. But it does hypothesize that the assertion is relevant to the current established goal of building a tower (call this G3) in some way. The message exchange is as follows:

```
CPS Manager → BA :
(EVALUATE :content (ASSERTION U1 :as
(CONTRIBUTES – TO :goal G3)))

BA → CPS Manager :
(ACCEPTABLE :what (ASSERTION U1 :as
(CONTRIBUTES – TO :goal G3))) :effect
(ADOPT U2 :as (MODIFICATION :of G3))

CPS Manager → BA :
(COMMIT :content (ADOPT U2 :as
(MODIFICATION :of G3)))
```

The BA proposes that the assertion modifies the shared goal G3 (filling in a constraint about who will move the blocks to build the tower). If the CPS manager accepts it then it will commit to the creation of the modified shared goal. Not in this case, but another possible reply is that the BA determines that the assertion should make a subgoal of G3.

Managing the BA's Contribution to the Dialogue: WHAT-NEXT

So far we have discussed how user utterances in a dialogue are interpreted to update the collaborative state. Here we will discuss how to manage the BA's utterances. Unlike a majority of other dialogue systems, the CPS framework does not enforce strict turn taking. Both the user and the system may produce multiple utterances in a row (see for example the dialogue in figure 3). For the CPS manager to control and coordinate system behavior, the BA has to wait until asked before it can contribute to the joint CPS state and dialogue. We call this the WHAT-NEXT message. Every time the interpretation of a user utterance is completed, the CPS manager evaluates the current state and sends a WHAT-NEXT message if the system is allowed to take the turn. For instance, if

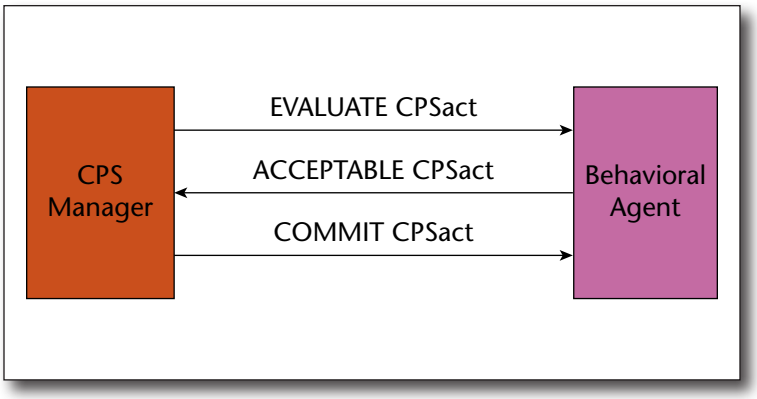


Figure 10. The EVALUATE-COMMIT Cycle for Intention Recognition.

there is pending user input, the user utterance takes priority and is processed first. This ensures that the system takes into account all the information from the user that could supplement or even modify the current state, to avoid situations where the system plans a response to the user's first utterance before considering the second one. For example:

User: *Let's build a tower.*
User: *It should be 5 blocks tall.*
System: *How tall should it be?*
< responding before considering the second user utterance >

The turn management allows both the user and the system to plan and execute multiple utterances within their turn.[6]

In response to the WHAT-NEXT request, the BA has a number of options. For example, it might invoke its planners and reasoners, or it might take an inventory of available resources. Eventually, however, it should respond to the CPS manager, even if just to say it is waiting for a task to finish. The CPS manager can then proceed to coordinate the next step in the collaborative problem-solving process. Table 3 summarizes a range of responses that might be returned.

A Generic Shell for CPS

The framework described in the last section defines the interface between the language interpretation/dialogue management components and the AI reasoning systems but makes no commitment to how these components are implemented. Given the diversity of applications where CPS could be used, BAs may differ significantly in their structure, just as the back-end reasoning engines they use may differ dramatically. As long as they support this interface, they can be integrated into the CPS framework. The same is true of the language and discourse processing components. However, a key strength of our approach is that much of the language and discourse processing can be implemented in a domain-independent fashion and be used in multiple collaborative systems in different domains and with radically different AI reasoning systems.

We have created a generic dialogue shell for systems that support collaborative problem-solving dialogues. This is described in detail in Galescu et al. (2018) and the code for the system (called *Cogent*) is publicly available on GitHub (the link is provided in the sidebar). This dialogue shell has been used to build systems in a range of domains, including a mixed-initiative system for planning and execution in blocks world (Perera et al. 2017); learning about structures in blocks world (Perera et al. 2018); an assistant to a biologist for building, visualizing, running, and modifying complex biologic causal models (Gyori et al. 2017; Burstein et al. 2020); helping a human composer create and edit music scores (Quick and Morrison 2017); playing cooperative games (Kim et al. 2018); and World Modeling (Allen and Teng 2019). Each one of these systems uses very different forms of domain-specific reasoning, but all use the same CPS framework and interface to the generic dialogue shell.

Generic Language Understanding for CPS

One of the unique strengths of Cogent, as exemplified in its instantiations in vastly different domains, is that the language and discourse processing can be constructed in a domain-independent manner. In contrast, virtually all current dialogue systems use domain-specific slot-filling parsers (whether hand-built or learned), and a new parsing system needs to be custom-constructed for each new domain, often starting by collecting (and annotating) a large corpus. As mentioned earlier, the slot-filling approach is not viable for such tasks as the complexity and variety of possible utterances require a compositional analysis of meaning. Progress will be greatly hampered unless we can build such a parsing system once and reuse it in new domains. In this section we briefly describe such a system. More details can be found in Allen and Teng (2017) and Allen et al. (2018a).

The core engine for processing language is the TRIPS parser. The name reflects its roots in the original TRIPS system (Ferguson and Allen 1998), which focused on a transportation domain. In the twenty-plus years since, TRIPS has been developed into a domain-general, broad coverage, deep semantic parser for both dialogue and open text such as web pages and scientific articles. By *broad coverage*, we mean that the lexicon substantially covers typical English usage (on the scale of WordNet; Fellbaum 1998). By *deep*, we mean that all words are assigned senses that are organized into an ontology, and that each sense has associated semantic roles, semantic preferences, syntactic linking templates, and axiomatization. By *ontology*, we mean not only a hierarchy

Response	What It Means	Example Utterances to Report Status to User
ANSWER	BA provides an answer to a user question	*We have three blocks on the table* (in answer to *How many blocks are available?*)
PROPOSE	BA proposes a problem-solving operation to the user	*ADOPT SUBGOAL: Let's consider rainfall first.*
		ABANDON GOAL: Shall we give up?
	BA asks the user a question	*ASK-WH: How tall should the tower be?*
		ASK-IF: Should I run the simulation?
FAILURE	BA reports a failure to perform some action	*There are no more red blocks.*
		We can't run the simulation as we have no rainfall data.
EXECUTION-STATUS	BA reports a goal/task is (successfully) completed	*I've built the tower.*
		The simulation is complete.
	BA reports that a goal/task is in progress	*I'm still working on it.*
		I'm running the simulation now.
	BA reports that it is waiting for the user	*I'm waiting for your decision.*

Table 3. Possible BA Responses to WHAT-NEXT

of concepts with inheritance of properties, but also axioms that capture the relationships between concepts, especially temporal and causal relationships. The TRIPS parser produces a rich representation of the sentence's meaning assigning word senses from its ontology to most words and linking them with well-founded semantic roles. Figure 11 shows a sample parse in graphical form. At a superficial level, the TRIPS representation looks similar to the abstract-meaning representation (Banarescu et al. 2013), but there are fundamental differences. Most importantly, all words in the TRIPS representation have senses in the TRIPS ontology, whereas, in abstract-meaning representation, for the most part only verbal forms and their derivational forms have sense tags. In addition there is no distinction in abstract meaning representation between a statement that a particular peach is juicy (for example, *The peach is juicy*) and a statement that all peaches are juicy (for example, *Peaches are juicy*). Such distinctions about quantifiers and others are critical for effective intention recognition. The TRIPS representation has a formal semantics that generalizes other well-known formalisms such as minimal recursion semantics (Copestake et al. 2005), hole semantics (Bos 2002), and dominance constraints (Koller et al. 2003). Thus, while the basic logical form does not scope quantifiers, operators, and adverbials, it can encode scoping constraints and supports tractable algorithms for scope disambiguation (Manshadi et al 2018).

The output from the parser is passed through a series of graph-based transformations, which rewrite and simplify the graphs into deeper representations. All of them use a subsystem that matches and rewrites graphs using rules defined in terms of the ontology. We say a pattern graph P matches a target graph T if there is a one-to-one mapping of the nodes and arcs in P to a subset of T such that the ontology type of each node in P is equal to or a supertype of the ontology type of the corresponding node in T. For example, a rule that identifies a likely intended speech act using conventional linguistic signals can be summarized as follows and shown graphically in figure 12 (terms with the prefix ONT:: are types in the TRIPS ontology):

> *If node A has a :content link to node B and B has a :modality link to node C, and (1) node A is ONT::SA_YN-QUESTION, (2) node B is a subclass of ONT::EVENT-OF-CHANGE*, and (3) node C is a subclass of *ONT::ABILITY, then transform the graph into an ONT::PROPOSE act with a :what link to node B.*

Matching the parse for *Can you analyze food insecurity in Sudan next year* (figure 11) against the above rule, we can derive that the sentence should be interpreted as an interaction act PROPOSE. This transformation mechanism is used in successive phases of interpretation described below.

Conventional Speech Act Interpretation

The first stage involves mapping the parsing output to a ranked set of possible intended speech act interpretations based on its lexical/syntactic/semantic structure, building from work originally by Hinkelman and Allen (1989). The example discussed above and shown in figure 12 depicts a simple rule that maps sentences of the form *Can you do X* (for example, the sentence in figure 11) to a proposal to adopt X as a shared goal. There are approximately 100 handbuilt rules that identify common phrasings with likely intentions in English. For instance, there are multiple ways in which goals might be proposed, including:

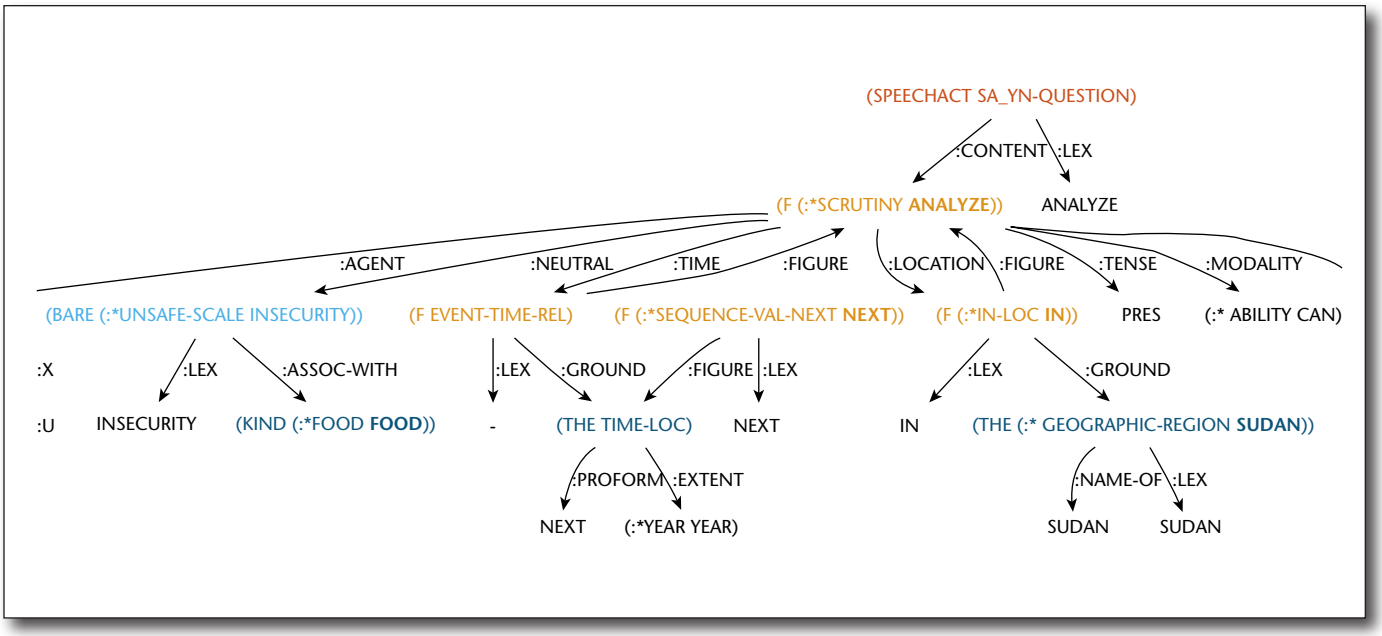


Figure 11. A Fragment of the Parse for "Can you analyze food insecurity in Sudan next year?"

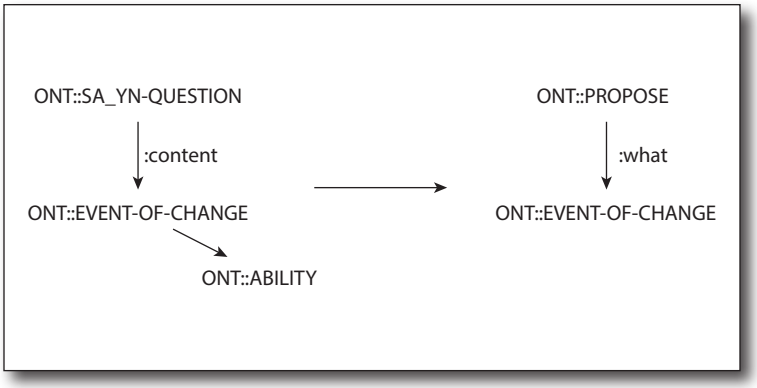


Figure 12. A Simple Transformation Rule for a Conventional Proposal.

I want you to do X. Why don't we do X?
Should/Can we do X? Let's do X.

Other rules relate to common forms of acceptance, agreement, and rejection, as well as sentence fragments as answers (for example, *two green blocks*). Some of the more complex rules are patterns that match conditional statements and map to speech acts such as conditional ASK-IF (*Is X if Y?*) and ASK-WHAT-IS (*What is X if Y?*) acts. These patterns encode conventional language use for English and are independent of any domain, but in conjunction with domain-specific named entity recognition they can be deployed for utterances with specialized vocabularies (for example, *Is Erk inactivated if I add Selumetinib?*).

It is also important to note that multiple patterns might match an input. For instance, *Do you know how to open the door?* might be simply a yes-no question about the hearer's abilities to open the door or, more likely, a proposal that the hearer actually open the door. The output of this first phase is a ranked list of possible conventional interpretations.

Reference Resolution

Another phase of processing rewrites subgraphs that capture definite descriptions and other referring expressions to terms that they refer to. A typical case involves pronominal reference to objects in the discourse, for example,

U: *Take a block out of the box.*
U: *Then put another block in it.*

Other cases refer to described events and activities, for example,

S: *Should I compute a baseline estimate?*
U: *How long will it take?*

TRIPS provides a rudimentary reference resolution capability that identifies likely antecedents of referring expressions by considering semantic compatibility and salience heuristics based on recency and grammatical role. Often the semantic constraints are derived from knowledge of the types of arguments the relations can take. For instance, in the first example, the word *in* is disambiguated to a relation ONT::IN-LOC in the TRIPS ontology, which is defined as a relation between two physical objects, the second of which is a container of some sort. To interpret the *it* reference, the system looks for the most recent mention that could be a container, in this case the box. In the second example, *it* is the subject of take, which is disambiguated to ONT::TAKE-TIME. The semantic

constraints on this ontology type indicate that its subject should refer to an event or plan. Thus, *it* is resolved to the most salient event in the discourse, that is, the proposed action of computing a baseline. Although the reference resolution mechanism does not operate using rewriting rules, it does rewrite the relevant terms by adding appropriate referential chains from referring expressions to their antecedents.

Domain-Specific Ontology Simplification

This optional stage is domain-dependent, and allows the semantic representation obtained from the above graph rewriting to be further transformed into different structures, even in terms of a different (domain-specific) ontology. This allows the representation to be simplified and customized to facilitate reasoning in the BA. For instance, in biocuration, a wide range of verb senses can be used to indicate the causal relation of regulation, which in the TRIPS ontology can be expressed as instances of ONT::CONTROL-MANAGE (for example, *controls*), ONT::OBJECTIVE-INFLUENCE (for example, *affects, impacts*) and others. Rather than having the BA deal with all these variations, we can define a single transformation rule that maps any node labeled with one of these types to a new node with a domain-specific relation named, for example, BOB::REGULATE. Furthermore, ONT::CONTROL-MANAGE has dozens of subtypes that are senses of additional relevant verbs (for example, *govern,* which belongs to ONT:: GOVERNING, a grandchild of ONT::CONTROL-MANAGE). These descendent types are automatically also included in the transformation. Such canonicalization and transformation can greatly simplify the reasoning the BA has to perform to interpret the user's utterances. A detailed description of the ontology mapping mechanism as applied to event extraction in the biology domain can be found in Allen et al. (2015).

Managing the Collaboration

Starting from a semantic parse of the user input, an utterance is successively transformed into deeper, and if desired more domain-specific, representations using several levels of graph-based rewriting rules. The resulting output is then passed to the CPS manager, whose job is to manage the shared problem-solving state by coordinating the interactions between the human and the BA. The CPS manager maintains the state regarding the negotiation of goals. Each state has a set of graph patterns that determine the appropriate action at the CPS level as well as a transition to the next state. The graph-matching mechanism described above is used to determine the active transitions. On entering a new state, the system performs the actions associated with the state. For example, it may send a message to the BA, and its transitions would interpret the response from the BA. As an illustration, consider the user utterance *Let's build a tower* and a fragment of the state and transition specification that deals with a simple propose-and-accept interaction to establish a new goal (figure 13). The CPS manager starts in the state labeled START. One of the transitions from START specifies a pattern that matches if the user proposes an event of type ONT::EVENT-OF-CHANGE. This matches the building event obtained from the utterance, and the system moves to state S1 and issues a call to the BA to evaluate the hypothesis that the user is proposing to ADOPT a new goal. Two transitions leave S1, matching the ACCEPTABLE and UNACCEPTABLE responses from the BA, respectively. Following the ACCEPTABLE transition to S2, the system sends a COMMIT act to the BA and generates an ACCEPT act to inform the user, thus establishing the shared goal of building a tower. From S2, one of the transitions (among others not shown) can be followed if there are no pending speech acts (that is, the user has not said anything else since we started this processing), in which case the system moves to S4 where it issues a WHAT-NEXT request to the BA to allow the system to take initiative for a response. The complete transition network to manage the CPS interactions consists of about thirty states and 120 transitions. As with language interpretation, the collaboration management transition network is fully domain-independent and used in all applications.

The actual system is more complex as the CPS manager also needs to consider the current shared state to make decisions, such as whether a proposal should be considered a new top-level goal or a subgoal to an existing goal. The CPS manager can rank the hypotheses based on the current state as well as linguistic and other domain-independent constraints, but ultimately, the decision of which hypothesis to accept can only be made after consulting the BA with its domain-specific reasoning.

Instantiating a Cogent-Based System

Because Cogent provides generic natural language understanding and (CPS-based) dialogue management, to create a new dialogue system the main effort is on the development of a BA that coordinates the domain-specific back-end AI systems and an NLG component. There are no requirements for how these two components should be implemented, except that the BA must implement the protocol described in the "Framework for Collaborative Problem-Solving Systems" section for managing the CPS state (the EVALUATE-COMMIT cycle for goals, handling WHAT-NEXT for taking initiative), and be able to map from CPS acts (ADOPT, ABANDON, RELEASE) to individual problem-solving acts it can execute. The BA also needs to be able to map the semantic interpretations produced by Cogent into possibly idiosyncratic representations used by the back-end AI systems. This process can be simplified by using the ontology mapping mechanism described in the "Generic Language Understanding for CPS" subsection.

The NLG component is highly domain-specific. Other than some domain-general speech acts (for

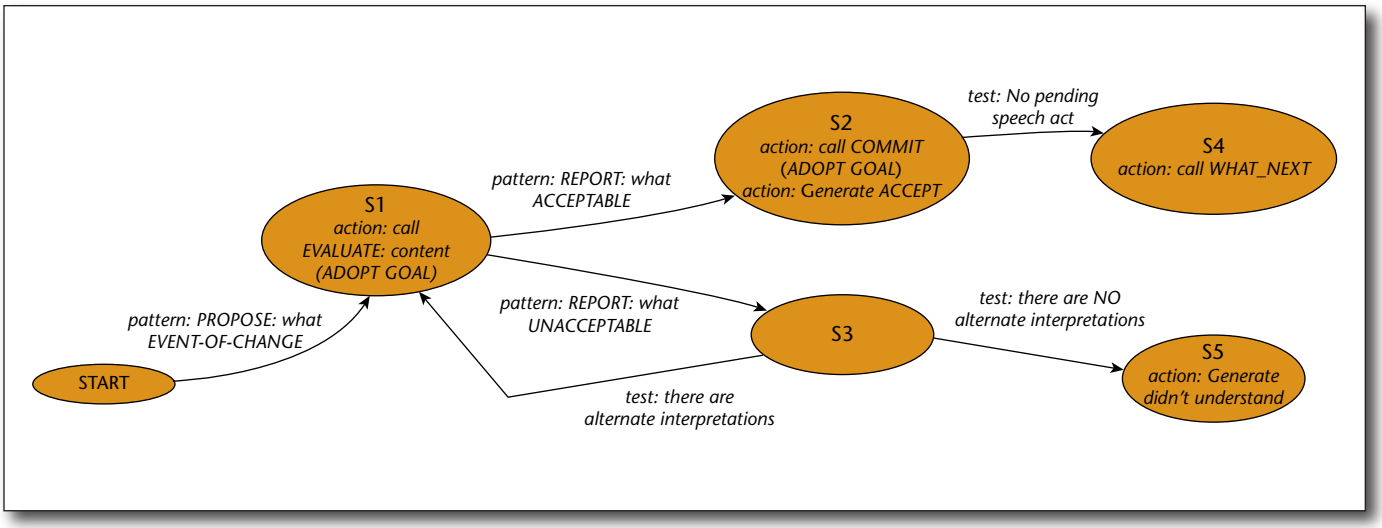


Figure 13. A Fragment of the CPS Manager Model (Simplified).

example, greetings, acknowledgments, reports of failure during natural language understanding) from the CPS manager, the content to be generated mainly comes from the BA itself, therefore it makes sense that both the BA and the NLG component be developed jointly. All Cogent-based systems currently implemented use some type of template-based NLG.

Regarding language understanding, the generic TRIPS parser and ontology provide reasonable coverage of utterances encountered in any domain, except for specialized lexical items such as organization acronyms and protein names. Thus, for most domains it is essential to implement a named entity recognizer. In Cogent, a preprocessing component called *TextTagger* takes databases of domain-specific names and augments the parser input with appropriate ontology types together with standardized identification information relevant to the domain (for example, International Organization for Standardization codes). In some domains, such as the blocks world, this could be as simple as a list of names of blocks and other objects in the domain. In others, such as the biocuration domain, TextTagger reads and processes millions of terms from biologic terminology and ontology resources, including names of proteins, chemicals, and other relevant objects. In the World Modelers domain, TextTagger reads large resources providing information about geographical regions (for example, local districts, towns, states, regions, and countries). An application program interface for augmenting parser input is provided such that developers can build their own named entity recognizers into the language-processing pipeline.

While our motivation for this framework is to facilitate the development of dialogue systems for domains of high complexity, we note that a BA at the level of complexity used in current dialogue-state tracking systems can easily work within Cogent. It would simply never use the PROPOSE act itself, because these systems do not ever have any initiative. The user may PROPOSE goals (*I am looking for a cheap restaurant in Cambridge*), ask questions (via ASK-IF/ASK-WHAT-IS) or make ASSERTIONs to specify values for various parameters (slots). The BA may use ASK-IF/ASK-WHAT-IS acts for getting values for its required slots, ANSWER for relaying answers to questions, and EXECUTION-STATUS to update the user when an action (for example, a booking) is done.

Evaluation

We described in this article a framework for developing systems that support dialogue-based interaction between humans and complex intelligent systems. It is hard to imagine how a conceptual framework could be evaluated directly. Rather, the worth of a framework is revealed by the breadth and depth of the systems that can be implemented in it, and to some extent the ease with which such systems can be developed and used. To draw an analogy with the state-based framework, researchers do not evaluate the state-based model of dialogue on its own, but rather they evaluate functional systems that are implemented using said state-based model. Similarly, here we describe a few studies of the BoB system, which is the most extensively studied and used system based on the CPS framework to date.

The BoB system described in the "Examples of Complex Collaborative Tasks" section was built on Cogent and involved a number of research teams in AI and biology. The BA, conforming to the specification described in the "Framework for Collaboration Problem-Solving Systems" section, was developed by SIFT (Smart Information Flow Technologies). The BA manages several domain-specific agents developed by experts in molecular biology at Harvard Medical School, Tufts University, and Oregon Health & Science University. We extended Cogent with a number of

specialized named entity taggers for biology (including genes, proteins, drugs, drug–protein and protein–protein interactions, cellular processes, and diseases), and with some domain-specific ontology mappings as described in the "Generic Language Understanding for CPS" section. Note that, however, parsing and interpretation are not specialized for the domain. The same domain general parsing and CPS manager are used across all Cogent-based systems, including BoB.[7]

BoB is under active development and has been regularly undergoing different types of evaluations, including some user studies. In one such study conducted using an early version of the system, eight biologists (most of whom were well versed in molecular biology but not necessarily experts in biologic modeling) were recruited by Harvard Medical School, Tufts University, and Oregon Health and Science University to test the system on three types of problems. For two of the problems the goal was to formulate and test a hypothesis that explained an observation about the effect of a drug on one or more molecular targets. The user was to first construct a plausible biologic mechanism, and then build a model for this hypothesized mechanism and check its validity by running simulations with this model (figure 3 provides an example of a dialogue while solving this kind of problem). The third problem was more open-ended; it was to look for a drug candidate that had a desirable outcome on a set of genes involved in a particular type of cancer. The users were given a short video introduction to BoB and a list of typical questions and statements that BoB could understand, although they were encouraged to express themselves in any way they found natural.

The subjects worked under three scenarios: using BoB plus the occasional assistance of a BoB expert to help with the interaction (but not with solving the problem itself); using BoB alone; and using information sources available online (a large list of resources was provided and the subjects were free to use any additional resources they were familiar with). In all scenarios, subjects were limited to thirty minutes per problem. Due to this time constraint and the complexity of the problems, full task completion was not expected (only one of the users, who was an expert modeler, completed his first two problems in full). Thus, standard evaluation metrics such as task-completion time, could not be used. Instead, a third-party evaluator (MITRE Corporation) devised a set of metrics for success in four subtasks (each with several milestones, which we will not go into here): discovering relevant information; finding complete molecular paths between the drug and the measured protein(s); building a biologically plausible model that addressed the experimental result; and successfully simulating the experimental result. The user performance for each of the four subtasks was assigned a score of 1 for completion, 0.5 when some but not all of the subtasks' milestones were achieved, and 0 for no milestone achieved. These scores were summed up to compute the final task completion score, which ranged from 0 to 4. The results for all three evaluation scenarios are summarized in table 4. An analysis of the transcripts found that, for a total of 491 user inputs, the system responded appropriately to seventy-two percent of them. In addition, the system was robust enough that users could continue (for example, by reformulating a request) even when the system did not understand or did not have the necessary problem-solving capability to respond appropriately initially.

The magnitude of the scores reflected the difficulty of the problems the users had to tackle, particularly for users who, while knowledgeable of molecular biology, were not modeling experts. However, all users were able to accomplish at least some of the subtasks. Based on the scores of the three test scenarios and responses from user surveys after the tasks, it was clear that the users found using BoB was far more efficient than using internet resources alone. They were able to progress much faster and further along toward solving the problems. They also found BoB fairly easy to use. It was reported that users had little need for assistance, and where assistance was provided (in the first test scenario), the users found it helpful and straightforward, which suggested that users inexperienced with BoB would need relatively little training with the system to be able to use it productively. Many of the users judged that BoB was a very helpful tool that they would like to use. Indeed, some of the biologists at Harvard Medical School and Tufts have integrated BoB into their regular suite of tools used during their research.

MITRE is also carrying out periodical stress tests on BoB, using the following series of hallmarks for guidance. Robustness: The system's language understanding and conversational capabilities are able to handle variations in how users might express themselves (lexical and syntactic variation, spelling errors) and conversational breakdowns (for example, the user not answering or providing incorrect answers to a question from the system, or switching topics). Explainability: The system can provide reasons for its behavior and explain its failures. Context awareness: The system uses dialogue context to improve understanding. Habitability: The system guides and enables users to use language naturally within the constraints of the system. Bidirectionality: The system actively and meaningfully contributes to the problem solving (and the conversation), rather than simply responding to questions and directives.

While explainability and habitability are, by and large, functions of the BA and the system's graphical user interface, Cogent's natural language understanding capabilities play a large role in the system's robustness. They also play a role in linguistic context awareness, although the testing was focused more on the task context (again, a function of the BA). The CPS model is crucial in enabling bidirectionality, although the content of the system's contributions is based on its domain-specific problem-solving

Task	BoB + assist	BoB	Web Tools
Discovered supported information (0-1)	0.94	0.81	0.69
Completed paths (0-1)	0.69	0.50	0.44
Built a model (0-1)	0.63	0.50	0.38
Ran simulation(s) (0-1)	0.25	0.25	0.00
Total score (0-4)	2.50	2.06	1.50

Table 4. Average Scores for User Evaluations.

capabilities. Figure 14 shows an example of the system taking initiative to make suggestions on how to improve a mechanistic model. The system also kept in sight the overall goal (finding how ERBB3 activates JUN) over the course of the dialogue. When the model under construction became capable of explaining the goal, BoB actively detected this change and informed the user of it, summarizing the causal explanation derived from the constructed model.

We will focus on robustness and bidirectionality in our discussion, as these metrics were most relevant to the domain-independent dialogue and CPS model. MITRE designed a set of 124 test inputs (in the context of a dialogue). Overall, the system scored eighty-eight percent on handling the robustness tests (including indirect ways of asking questions, typos, and other spelling mistakes in every-day as well as biology-specific words, such as synonyms for biologic entities). On bidirectionality, the system achieved a score of seventy-five percent. From MITRE's experience in evaluating such systems, it was judged that scores above sixty percent were expected to lead to a good user experience with the system.

These evaluations indicated that third parties were able to integrate sophisticated domain-specific AI systems within the Cogent shell, and build an efficient and effective dialogue system capable of helping users solve complex problems. The underlying CPS model and the system's domain-independent language understanding and dialogue management capabilities were a viable approach to solving complex tasks in collaboration with the human user.

It should be noted that, because these were stress tests, with deliberately ill-formed phrasings and spelling mistakes designed to resemble specific alternatives, they reflected an expectation of system performance under a worst-case scenario. Nonetheless, although the results from both these tests and the user studies mentioned above were encouraging, further and more extensive evaluation would be needed to better understand the behavior and performance of the model.[8]

Concluding Remarks

We have discussed a framework for dialogue systems that can partake in dialogues for tasks significantly more complex than possible with current state-based and machine learning-based approaches. This model supports dialogue systems in which humans can collaborate with AI reasoning systems to jointly tackle complex problem-solving tasks. By developing a system that exploits an abstraction of the CPS process that is portable across domains, we provide a rich environment for building a wide range of new applications without the need to develop each system from scratch. Significantly, this model can be used in any domain that can be cast as CPS, including applications in which the task models cannot be defined in advance, which broadens the repertoire and complexity of tasks that can be addressed by conversational agents.

Our solution provides a well-defined interface between the generic dialogue system and the domain-specific AI reasoning systems that vary from domain to domain. What is required to implement a new system is the development of a domain-specific BA that interacts with the generic CPS model and coordinates the back-end reasoning systems. More details on the BA for CWMS and how it drives various agricultural and economic reasoning engines can be found in Allen and Teng (2019). More details on the BA in the BoB system, and how it drives multiple biologic simulators and reasoning agents, can be found in Burstein et al. (to appear). The code for our generic dialogue components, including the parser, is available on GitHub and is described in Galescu et al. (2018).

While the framework has proven to be effective for building dialogue systems across a variety of complex domains, there is still much room for improvement. For language interpretation, the parser currently exploits only very simple features of the discourse context. Possible interpretations are ranked mostly based on static semantic preferences for arguments for each predicate, as well as domain-specific preferences for senses. Taking better account of the nuances of the dynamically evolving context would result in more accurate parsing. In addition, more complex discourse interactions, such as answering multiple choice clarification questions, currently are handled fairly formulaically, putting the burden of fine disambiguation on the BA.

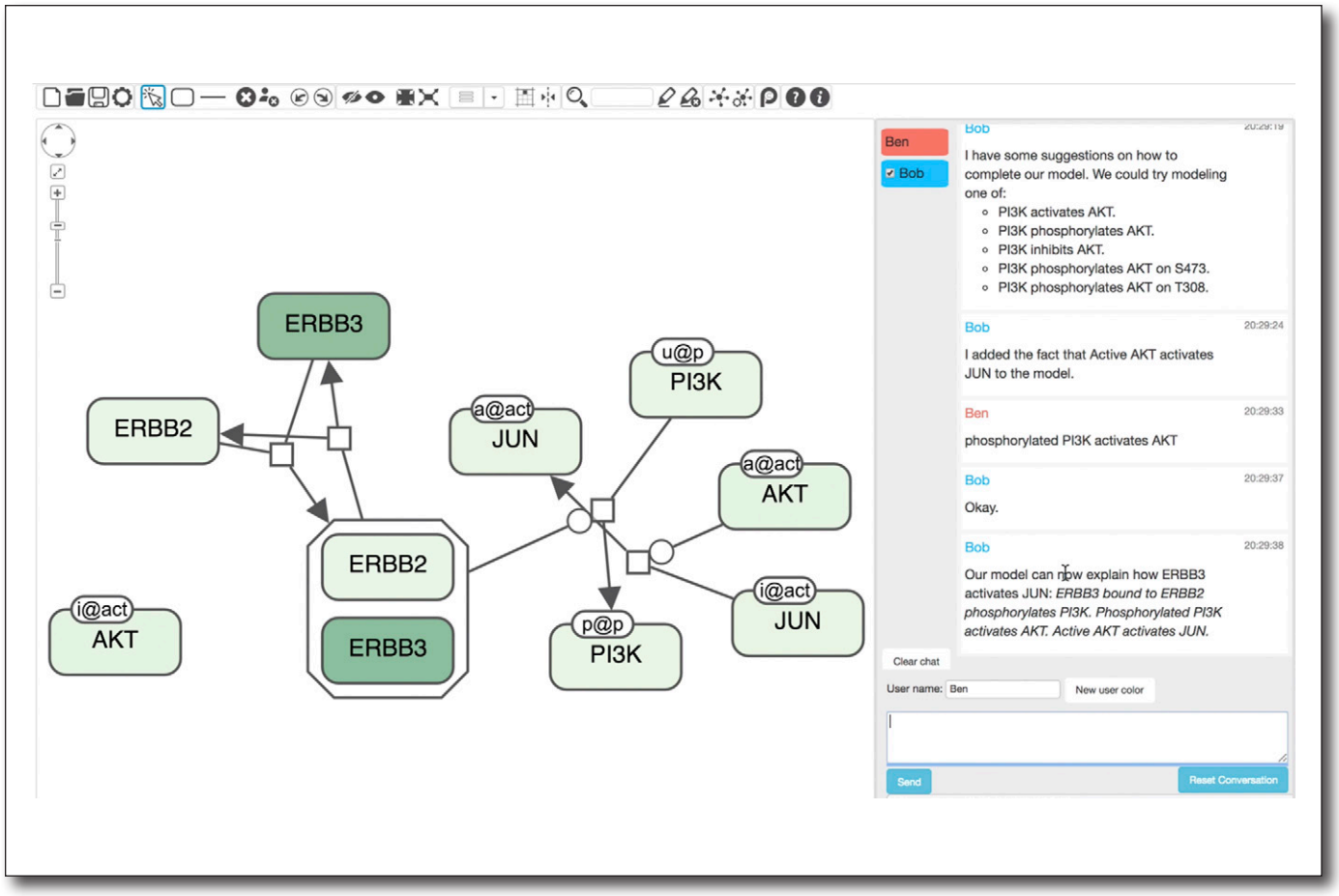


Figure 14. Excerpt from a Dialogue with BoB Showing System Initiative.

Ben is the user; Bob is the system.

Links to TRIPS-Based Systems

For further details on TRIPS-based systems, the reader is referred to the following links:
trips.ihmc.us/parser/cgi/lex-ont, for browsing the TRIPS lexicon and ontology
trips.ihmc.us/parser, for on-line interfaces to the TRIPS parser customized for different domains, including CWMS, BoB, and Cabot (blocks world)
trips.ihmc.us/cogent/video, for examples of demos and dialogues carried out with several Cogent-based dialogue systems
github.com/wdebeaum/cogent, for source code for the generic dialogue shell based on CPS (Cogent), which includes the parser, dialogue management, and the CPS manager, but no BA

For CPS state management, the system relies on a state transition network to determine the permissible changes and the actions to be performed. The CPS manager can handle many common interaction scenarios, including clarification dialogues and redirection and modification of the problem-solving subtasks. However, unexpected responses and interruptions sometimes could derail the problem-solving process. In most cases the system can recover and continue, but with some loss of context and state information. Refining and expanding the transition network to better manage the problem-solving states is one of our highest priorities.

Although our framework and system significantly reduce the efforts required to build AI systems that can collaborate with humans, this is not to say that

building such systems is now easy. The development of the BA remains a complex task, as most of the common-sense inference needed to understand the user intent and plan responses must be encoded there. Building a BA is, strictly speaking, outside the scope of Cogent. However, we will use our experience in interfacing with a variety of BAs (including some we built ourselves) to improve and support their integration and development.

While many challenges remain for building truly robust collaborative systems, we believe that the partition of responsibilities we have outlined in this article, with the dialogue being modeled by our domain-general model of CPS, will provide a framework for building truly useful systems in the future — systems that are capable of meaningful collaboration with humans to tackle tasks of a complexity found in real-life problems.

Acknowledgments

The systems and evaluations mentioned in this article would not have been possible without the invaluable contributions of our many collaborators at Smart Information Flow Technologies, Harvard Medical School, Tufts University, Oregon Health & Science University, MITRE Corporation, SRI International, and the University of Florida. We also thank the reviewers for this special issue for their comments that greatly improved the presentation.

This research was supported by the Defense Advanced Research Projects Agency (Army Research Office contracts W911NF-14-1-0391, W911NF-15-1-0542, W911NF-17-1-0047, and W911NF-18-1-0464).

Notes

1. www.microsoft.com/en-us/research/event/dialog-state-tracking-challenge
2. Semantic Scholar shows 174 publications for the query (ATIS "intent detection" "slot filling" "neural network") since 2003, over half of which have been published in the last 3 years. The ATIS task domain is over 30 years old.
3. See trips.ihmc.us/cogent/video
4. A publicly available web-based interface to the BoB system is available at www.dialogue.bio.
5. www.w3.org/TR/voicexml21
6. The CPS "manager" also manages real-time aspects of the dialogue. If the user does not respond to a question or request within a certain amount of time, it reminds the user that it is waiting for a response, and ultimately sends a WHAT-NEXT request to the BA to allow it to decide what to do next.
7. BoB itself has been used, without change, as the dialogue component for a multimodal speech-based system for visualization and exploration of bioinformatic data (Hutchens et al. 2020).
8. As of this writing, another user study is being planned for a new version of BoB, with improved natural language understanding and problem-solving capabilities. In this study users will be biology experts who will try to use BoB to explore problems in their own areas of research.

References

Allen, J.; Bahkshandeh, O.; de Beaumont, W.; Galescu, L.; and Teng, C. M. 2018a. Effective Broad-Coverage Deep Parsing. In *Proceedings of the Thirty-Second Association for the Advancement of Artificial Intelligence (AAAI) Conference on Artificial Intelligence, AAAI 2018*, 4776–83. Palo Alto, CA: AAAI Press.

Allen, J.; Blaylock, N.; and Ferguson, G. 2002. A Problem Solving Model for Collaborative Agents. In *Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems: Part 2*, 774–81. New York: Association for Computing Machinery.

Allen, J.; Byron, D.; Dzikovska, M.; Ferguson, G.; Galescu, L.; and Stent, A. 2000. An Architecture for a Generic Dialogue Shell. *Natural Language Engineering* 6(3–4): 213–28.

Allen, J.; de Beaumont, W.; Galescu, L.; and Teng, C. M. 2015. Complex Event Extraction Using DRUM. In *Proceedings of Biomedical Natural Language Processing*. 1–11. Stroudsburg, PA: Association for Computational Linguistics.

Allen, J., and Teng, C. M. 2019. Human-Machine Collaboration for World Modeling. *In Modeling the World's Systems*. Pittsburgh, PA: University of Pittsburgh.

Allen, J.; Teng, C. M.; and Galescu, L. 2018b. Dialogue as Collaborative Problem Solving: A Case Study. *Advances in Cognitive Systems* 7(2019): 195–214.

Allen, J. F., and Perrault, C. R. 1980. Analyzing Intention in Utterances. *Artificial Intelligence* 15(3): 143–78.

Allen, J. F., and Teng, C. M. 2017. Broad Coverage, Domain-Generic Deep Semantic Parsing. In *Computational Construction Grammar and Natural Language Understanding: Papers from the 2017 Association for the Advancement of Artificial Intelligence (AAAI) Spring Symposium Series*. Technical Report SS-17-02. Palo Alto, CA: AAAI Press.

Banarescu, L.; Bonial, C.; Cai, S.; Georgescu, M.; Griffitt, K.; Hermjakob, U.; Knight, K.; Koehn, P.; Palmer, M.; and Schneider, N. 2013. Abstract Meaning Representation for Sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, 178–86. Stroudsburg, PA: Association for Computational Linguistics.

Blaylock, N., and Allen, J. 2005. A Collaborative Problem-Solving Model of Dialogue. In *Proceedings of the Sixth Special Interest Group on Discourse and Dialogue (SIGdial) Workshop on Discourse and Dialogue, SIGdial 2005,* 200–11. Stroudsburg, PA: Association for Computational Linguistics.

Bohus, D., and Rudnicky, A. I. 2009. The RavenClaw Dialog Management Framework: Architecture and Systems. *Computer Speech & Language* 23(3): 332–61.

Bos, J. 2002. Underspecification and Resolution in Discourse Semantics. PhD thesis. Saarland University, Saarbrucken, Germany.

Burstein, M., Friedman, S., McDonald, D., Rye, J., Plotnick, A., Bobrow, L., and Bobrow, R. 2020. Using Multiple Contexts to Interpret Collaborative Task Dialogs. To appear in *Advances in Cognitive Systems*, 9.

Chu-Carrol, J., and Carberry, S. 1998. Collaborative Response Generation in Planning Dialogues. *Computational Linguistics* 24: 355–400.

Cohen, P. R. 2019 Foundations of Collaborative Task-Oriented Dialogue: What's in a Slot? In *Proceedings of the Special Interest Group on Discourse and Dialogue (SIGdial) Workshop on Discourse and Dialogue, SIGdial 2019*. Stroudsburg, PA: Association for Computational Linguistics.

Cohen, P. R., and Levesque, H. 1990. Rational Interaction as a Basis for Communication. In *Intentions in Communication*. P. R. Cohen, J. Morgan, and M. E. Pollack, editors. The MIT Press. Cambridge, MA.

Copestake, A.; Flickinger, D.; Pollard, C.; and Sag, I. A. 2005. Minimal Recursion Semantics—An Introduction. *Research on Language and Computation* 3(2–3): 281–332.

Fellbaum, C., 1998. *WordNet: An Electronic Lexical Database*. Cambridge, MA: MIT Press.

Ferguson, G., and Allen, J. 1998. TRIPS: An Integrated Intelligent Problem-Solving Assistant. In *Proceedings of the National Conference on Artificial Intelligence, Association for the Advancement of Artificial Intelligence, AAAI 1998*. Palo Alto, CA: AAAI Press.

Ferguson, G., and Allen, J. 2007. Mixed-Initiative Systems for Collaborative Problem Solving. *AI Magazine* 28: 23–32.

Galescu, L.; Teng, C. M.; Allen, J.; and Perera, I. 2018. COGENT: A Generic Dialogue System Shell Based on a Collaborative Problem Solving Model. In *Proceedings of the Nineteenth Annual Meeting of the Special Interest Group on Discourse and Dialogue, SIGdial 2018,* 400–49. Stroudsburg, PA: Association for Computational Linguistics.

Ghallab, M.; Nau, D.; and Traverso, P. 2004. *Automated Planning: Theory and Practice*. San Francisco, CA: Morgan Kaufmann

Goddeau, D.; Meng, H.; Polifroni, J.; Seneff, S.; and Busayapongchai, S. 1996. A Form-Based Dialogue Manager for Spoken Language Applications. In *Proceeding of Fourth International Conference on Spoken Language Processing, ICSLP 1996*. Piscataway, NJ: Institute for Electrical and Electronics Engineers (IEEE).

Grosz, B. 1974) The Structure of Task Oriented Dialogs. In *Proceedings of the Institute for Electrical and Electronics Engineers (IEEE) Symposium on Speech Recognition*. Piscataway, NJ: IEEE.

Grosz, B. J., and Kraus, S. 1996. Collaborative Plans for Complex Group Action. *Artificial Intelligence* 86(2): 269–357.

Grosz, B. J., and Sidner, C. 1986. Attention, Intention, and the Structure of Discourse. *Computational Linguistics* 12(3): 175–204.

Gyori, B. M.; Bachman, J. A.; Subramanian, K.; Muhlich, J. L.; Galescu, L.; and Sorger, P. K. 2017. From Word Models to Executable Models of Signaling Networks Using Automated Assembly. *Molecular Systems Biology* 13(11): 954.

Hatfield-Dodds, S.; Adams, P. D.; Brinsmead, T. S.; Bryan, B. A.; Chiew, F. H. S.; Finnigan, J. J.; Graham, P. W.; Grundy, M. J.; Harwood, T.; McCallum, R., et al. 2015. *Australian National Outlook 2015: Economic Activity, Resource Use, Environmental Performance and Living Standards 1970–2050*. Canberra, Australia: Commonwealth Scientific and Industrial Research Organisation (CSIRO).

Hinkelman, E., and Allen, J. 1989. Two Constraints on Speech Act Ambiguity. In *Annual Meeting of the Association for Computational Linguistics, ACL 1989*. Stroudsburg, PA: Association for Computational Linguistics.

Hutchens, M.; Krishnaswamy, N.; Cochran, B.; and Pustejovsky, J. 2020. Jarvis: A Multimodal Visualization Tool for Bioinformatic Data. In *International Conference on Human-Computer Interaction (HCII): Late-Breaking Articles*. Berlin, Germany: Springer.

Kautz, H. A., and Allen, J. F. 1986. Generalized Plan Recognition. In *Fifth National Conference on Artificial Intelligence, Association for the Advancement of Artificial Intelligence, AAAI 1986*. Los Altos, CA: William Kaufmann.

Kim, K.; Lee, C.; Jung, S.; and Lee, G. 2008. A Frame-Based Probabilistic Framework for Spoken Dialog Management Using Dialog Examples. In *Proceedings of the Special Interest Group on Discourse and Dialogue, SIGdial 2018,* 120–7. Stroudsburg, PA: Association for Computational Linguistics.

Kim, S.; Salter, D.; DeLuccia, L.; Son, K.; Amer, M. R.; and Tamrakar, A. 2018. SMILEE: Symmetric Multi-Modal Interactions with Language-Gesture Enabled (AI) Embodiment. In *Proceedings of the Sixteenth Annual Conference of the North American Chapter of the Association for Computational Linguistics (ACL): Human Language Technologies,* 86–90. Stroudsburg, PA: Association for Computational Linguistics.

Koller, A.; Niehren, J.; and Thater, S. 2003. Bridging the Gap Between Underspecification Formalisms: Hole Semantics as Dominance Constraints. In *Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics, EACL-03*, 195–202. Stroudsburg, PA: Association for Computational Linguistics.

Larsson, S., and Traum, D. R. 2000. Information State and Dialogue Management in the TRINDI Dialogue Move Engine Toolkit. *Natural Language Engineering* 6(3–4): 323–40.

Lemon, O.; Gruenstein, A.; Cavedon, L.; and Peters, S. 2002) Multi-Tasking and Collaborative Activities in Dialogue Systems. In *Proceedings of the Special Interest Group on Discourse and Dialogue, SIGdial 2002*. Stroudsburg, PA: Association for Computational Linguistics.

Li, J.; Peng, B.; Lee, S.; Gao, J.; Takanobu, R.; Zhu, Q.; Huang, M.; Schulz, H.; Atkinson, A.; and Adada, M. 2020. Results of the Multidomain Task-Completion Dialog Challenge. Paper presented at the Eighth Dialog System Technology Challenge (DSTC8) Workshop, February 8, 2020, New York, NY.

Lin, Z.; Huang, X.; Ji, F.; Chen, H.; and Zhang, Y. 2019) Task-Oriented Conversation Generation Using Heterogeneous Memory Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019*. Stroudsburg, PA: Association for Computational Linguistics.

Litman, D. J., and Allen, J. F. 1987. A Plan Recognition Model for Subdialogues in Conversations. *Cognitive Science* 11(2): 163–200.

Liu, B., and Lane, I. 2016) Attention-Based Recurrent Neural Network Models for Joint Intent Detection and Slot Filling. In *Interspeech 2016, 17th Annual Conference of the International Speech Communication Association,* 685–9. Stroudsburg, PA: Association for Computational Linguistics.

Lochbaum, K. E.; Grosz, B. J.; and Sidner, C. L. 1990. Models of Plans to Support Communication: An Initial Report. In *Proceedings of the Eighth National Conference on Artificial Intelligence, Association for the Advancement of Artificial Intelligence (AAAI),* 485–90. Cambridge, MA: AAAI Press/The MIT Press.

Manshadi, M.; Gildea, D.; and Allen, J. 2018. A Notion of Semantic Coherence for Underspecified Semantic Representation. *Computational Linguistics* 44(1): 39–83.

Mrkšić, N.; Séaghdha, D. Ó.; Thomson, B.; Gašić, M.; Su, P.; Vandyke, D.; Wen, T.; and Young, S. 2015. Multi-Domain Dialog State Tracking Using Recurrent Neural Networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL) and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015*, 794–799. Stroudsburg, PA: Association for Computational Linguistics.

This *AI Magazine* special topic article on Conversational AI (along with articles on the same topic in the fall issue) have been selected and edited by guest editors Alborz Geramifard (Facebook AI), Dilek Hakkani-Tur (Amazon Alexa AI), Peter Henderson (Stanford University), Alex Rudnicky (Carnegie Mellon University), and Asli Celikyilmaz (Microsoft Research)

Nouri, E., and Hosseini-Asl, E. 2018. Toward Scalable Neural Dialogue State Tracking. Paper presented at the NeurIPS (NIPS) Second NIPS Workshop on Conversational Artificial Intelligence (CAI), Montreal, Quebec, Canada, December 7, 2018. arxiv.org/pdf/1812.00899.pdf.

Perera, I.; Allen, J.; Galescu, L.; Teng, C. M.; Burstein, M.; McDonald, D.; Rye, J.; and Friedman, S. 2017. Natural Language Dialogue for Building and Learning Models and Structures. In *Proceedings of the Thirty-First Association for the Advancement of Artificial Intelligence (AAAI) Conference on Artificial Intelligence,* 5103–4. Palo Alto, CA: AAAI Press.

Perera, I.; Allen, J. F.; Galescu, L.; and Teng, C. M. 2018. A Multimodal Dialogue System for Learning Structural Concepts in Blocks World. In *Proceedings of the Nineteenth Annual Meeting of the Special Interest Group on Discourse and Dialogue, SIGdial 2018,* 89–98. Stroudsburg, PA: Association for Computational Linguistics.

Pulman, S. 1997. Conversational Games, Belief Revision and Bayesian Networks. Paper presented at the Seventh Computational Linguistics in the Netherlands (CLIN VII) meeting. citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.47.6998&rep=rep1&type=pdf.

Quick, D., and Morrison, C. T. 2017. Composition by Conversation. In *Proceedings of the Forty-Third International Computer Music Conference,* 52–7. Vienna, Austria: International Society for Contemporary Music.

Rich, C., and Sidner, C. 1998. COLLAGEN: A Collaboration Manager for Software Interface Agents. *User Modeling and User-Adapted Interaction* 8(3/4): 315–50.

Rich, C., and Sidner, C. L. 2012. Using Collaborative Discourse Theory to Partially Automate Dialogue Tree Authoring. In *Intelligent Virtual Agents*. Vol. 7502. Y. Nakano, M. Neff, A. Paiva, and M. Walker, editors. Springer. Berlin, Heidelberg.

Rickel, J., and Johnson, W. L. 1998. STEVE: A Pedagogical Agent for Virtual Reality. *Agents* 98: 332–3.

Serban, I. V.; Sordoni, A.; Bengio, Y.; Courville, A.; and Pineau, J. 2016. Building End-to-End Dialogue Systems Using Generative Hierarchical Neural Network Models. In *Proceedings of the Thirtieth Association for the Advancement of Artificial Intelligence (AAAI) Conference on Artificial Intelligence*. Palo Alto, CA: AAAI Press.

Valenzuela-Escárcega, M. A.; Babur, Ö.; Hahn-Powell, G.; Bell, D.; Hicks, T.; Noriega-Atala, E.; Wang, X.; Surdeanu, M.; Demir, E.; and Morrison, C. T. 2018. *Large-Scale Automated Machine Reading Discovers New Cancer-Driving Mechanisms Database*. Washington, DC: US National Library of Medicine.

Wang, Y.; Shen, Y.; and Jin, H. 2018. A Bi-Model Based RNN Semantic Frame Parsing Model for Intent Detection and Slot Filling. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018*. Stroudsburg, PA: Association for Computational Linguistics.

Wang, Z., and Lemon, O. 2013. A Simple and Generic Belief Tracking Mechanism for the Dialog State Tracking Challenge: On the Believability of Observed Information. In *Proceedings of the 14th Annual Meeting of the Special Interest Group on Discourse and Dialogue, SIGdial 2013*. Stroudsburg, PA: Association for Computational Linguistics.

Wen, T.; Vandyke, D.; Mrkšić, N.; Gaćić, M.; Rojas-Barahona, L. M.; Su, P.; Ultes, S.; and Young, S. 2017. A Network-Based End-to-End Trainable Task-Oriented Dialogue System. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017*, 438–49. Stroudsburg, PA: Association for Computational Linguistics.

Williams, J.; Raux, A.; and Henderson, M. 2016. The Dialog State Tracking Challenge Series: A Review. *Dialogue and Discourse* 7: 4–33.

Young, S.; Gašić, M.; Thomson, B.; and Williams, J. D. 2013. POMDP-Based Statistical Spoken Dialog Systems: A Review. *Proceedings of the IEEE* 101(5): 1160–79.

Zhang, X., and Wang, H. 2016. A Joint Model of Intent Determination and Slot Filling for Spoken Language Understanding. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016*. Palo Alto, CA: Association for the Advancement of Artificial Intelligence (IAAA) Press.

Zue, V.; Seneff, S.; Glass, J.; Polifroni, J.; Pao, C.; Hazen, T.; and Hetherington, L. 2000. Jupiter: A Telephone-Based Conversational Interface for Weather Information. *IEEE Transactions on Speech and Audio Processing* 8(1): 85–96.

James Allen (jallen@ihmc.us) is the John H. Dessauer Professor of Computer Science at the University of Rochester and associate director of IHMC. He is a fellow of the Association for the Advancement of Artificial Intelligence and the Cognitive Science Society, and the 2020 winner of the Simon Prize for Advances in Cognitive Systems.

Lucian Galescu (lgalescu@ihmc.us) is a research scientist at IHMC. He received a PhD in computer science from the University of Rochester in 2003, and his research has primarily been focused on dialogue systems and statistical natural language processing.

Choh Man Teng (cmteng@ihmc.us) is a research scientist at IHMC. Her research interests include language understanding, uncertain reasoning, and knowledge representation.

Ian Perera (iperera@ihmc.us) is a research scientist at IHMC. His research interests include situated dialogue understanding, automatic evaluation of teamwork dialogue, and event extraction from text.