Advances in Cognitive Systems 9 (2020) 71–90 Submitted 5/2019; published 12/2020

Using Multiple Contexts to Interpret Collaborative Task Dialogs

Mark Burstein, Scott Friedman, David McDonald, Jeff Rye, Alex Plotnick, Laurel Bobrow
{BURSTEIN, SFRIEDMAN, DMCDONALD, RYE, APLOTNICK, LBOBROW} @ SIFT.NET
SIFT, LLC, Lexington, MA 02421 USA

Robert Bobrow RJBOBROW @ GMAIL.COM
Bobrow Computational Intelligence, LLC, Lexington, MA 02421 USA

Abstract

CLIC is our architecture as a behavioral reasoning agent for agent-based, multi-domain, collaborative dialog. CLIC performs the domain-specific reasoning enabling the system to ask and answer questions, discuss goal refinements, and take turns acting on them. The agent is responsible for interpreting utterances as shared goals, be they questions or assertions, and then determining how to further advance those goals. It decides how to act or suggest user actions, and it can report problems or ask the user questions. We have demonstrated the system in several domains. In this paper we discuss the architecture and associated forms of reasoning performed by CLIC as it interprets semantic representations of user utterances to derive goals and actionable intentions. Critical to the interpretation process is the use of multiple sources of context. These contexts include semantic items previously mentioned or referenced in either participants' contributions to the dialog, the shared and changing visual environment, and items implicit in shared goals, such as the envisioned results and methods to achieve them. We also report how the system resolves references using those contextual sources when determining how to act.

1. Introduction

This paper describes the behavior of the CLIC system (Communication with Language-Integrated Contexts), that we developed as part of an architecture for dialog systems that would function effectively in multiple domains. The overall architecture is designed to engage in two-way discussions about shared goals and tasks. CLIC is the behavioral reasoning component, also called the Behavioral Agent (BA), and its role is to interpret semantic representations of user inputs from parsed text as actionable goals or assertions to be interpreted in their context. The BA must be sensitive to multiple kinds of contextual background during reference resolution. These sources arise from the reasoning involved in planning, acting, question answering, and interpreting answers to clarification questions the system poses, where it must correctly use a variety of domain resources. Based on this architecture and examples of its operation, we make several theoretical claims about the process of goal-oriented dialog.

- Systems for this task must support references to multiple forms of context beyond prior dialog utterances, including references to the shared environment, unmentioned but inferred goals or outcomes, and partial plans discussed to achieve them.
- Resolving references found in user utterances and questions is a co-compositional process, where the possible interpretations of the utterances as actionable goals and the available referent types are mutually constraining.
- When a semantic parser's interpretation retains aspects of surface syntactic or clause structure in its output representation, different ways of phrasing goals or asking questions can produce different semantic representations that are requests to achieve the same goal.

We begin by situating our work in the background of related work that led to it. We then lay out the architecture of the overall dialog system and the place of CLIC within it. We then show how the architecture uses multiple contexts and explain its mechanisms for goal reasoning. We present examples related to our claims and empirical results using context to encode and revise goals. We close with a brief discussion of our experiments and our future plans.

2. Background

We start by briefly describing the earlier research that led to the conception of dialog that frames our work. We then expand on the larger project of which the CLIC system is a part, describe components developed by our collaborators, and explain briefly how the elements of the system are integrated and coordinated.

2.1 Related Work

Given our perspective — that dialog should be task-oriented, collaborative, and precise — many of the presently popular approaches to conversation and chat are not relevant. Undirected conversational systems, such as the ones developed to compete for the Alexa Prize (Khatri et al., 2018), where the goal is simply to engage people in the dialog for as long as possible, are not comparable in that, without a purpose, there is no need to establish a shared goal context and interpret utterances to further those goals.

Even task-oriented dialog systems like those being developed for the Dialog State Technology Challenge (Williams et al., 2016) are not quite on point. These hold speech-driven conversations about tasks such as ordering pizza, choosing a restaurant, or booking a hotel, and they are usually trained end-to-end from speech in to speech out using some variation on deep neural networks, e.g., Madotto et al. (2018). However, their dialog-management components use simple, direct APIs to answer user questions. There is little or no context developed between questions, and their conception of language understanding is simply filling slots in task-specific attribute-value frames, which was the state of the art 30 years ago in air traffic control dialogs (Young, 1991).

To find dialog systems that can be usefully compared to ours we have to go back to the classical literature, such as Winograd's SHRDLU (1973). Along with the other major systems of that era,[1] SHRDLU pioneered capabilities now considered the baseline for knowledge-based dialog systems:

1. These include Lunar (Woods et al., 1972), Margie (Schank et al., 1973), and GUS (Bobrow et al., 1977).

- A linguistically principled grammar with extensive coverage of English constructions;
- An explicit semantic representation of the things it knows about, which it uses to interpret statements and referring expressions;
- Asking clarification questions when the user has not provided enough information;
- Maintaining a dynamic model of its current situation: a semantic record of what has been said and the plan being followed, that lets it use and understand pronouns, as well as providing an introspective basis for describing what it has done and why.

Later large-scale efforts to develop speech-based dialog systems (e.g., Wahlster, 2000) followed the same general architecture earlier ones. The initial work in building large systems exposed many limitations and unhandled language phenomena that led to foundational work on the nature of man-machine dialog. Our approach to dialog emerged from a particular thread through this work, including the central role of plans and how the flow of topics is systematically related to the task structure, along with the nature of intentions and their relationship to speech acts, the state of a person's beliefs, and the structure of dialog as a collaborative process.[2] Another thread focused on 'dialog moves' and their planning, as well as conversation among embodied agents in games with a purpose.[3] Our work following the first thread builds on the work over several decades by James Allen and his colleagues on coordinating multi-component systems within a coherent theory of collaborative task-oriented conversation.

2.2 System Overview and Architecture

As part of DARPA's Communicating with Computers program (Cohen, 2017), SIFT teamed with James Allen's group at IHMC and the University of Rochester to develop a common architecture for multi-modal human-computer interaction. The system engages in two-way dialogs that both develop and pursue shared goals. Both the human and automated participants in the conversation ask and answer questions, taking turns in shared task environments. We developed the system using the TRIPS agent-based architecture and its framework for dialog systems (Blaylock & Allen, 2005).

Figure 1 shows the system's structure and the components or agents each research team developed. TRIPS distributed architecture handles communications between modules called agents (Allen et al., 2001). The TRIPS natural language PARSER and Interpretation Manager (IM) agents create a logical semantic form for each utterance, which is fed to the Collaborative Problem Solving Agent (CPSA) to maintain a domain-independent dialog state, including interpreting user utterances as assertions, goal specifications, or goal modifications, passing these to our Behavioral Agent (BA), and coordinating responses to the user. Interactions between the CPSA and BA use a goal-oriented coordination protocol (**evaluate**, **adopt**|**reject**, **commit**, **what-next**, **succeed**|**fail**) as described by Blaylock and Allen (2005). SIFT also developed an agent for generating responses in natural language. although we do not discuss it in this paper.

CLIC acts as the domain reasoning agent (BA) in both of our dialog system domains. In the first domain, Blocks World, the user and system collaborate to assemble structures with a set of blocks on a table: one version used real blocks and the other was a virtual environment, as shown in Figure 2.

2. **Plans**: Grosz (1980); Grosz and Kraus (1996); Hobbs and Evans (1980); **Intentions**: Cohen and Perrault (1979); Cohen and Levesque (1990); **Collaboration**: Rich and Sidner (1998); Lochbaum (1998)
3. Cassell et al. (1999); Traum (2000); Steedman and Petrick (2007)

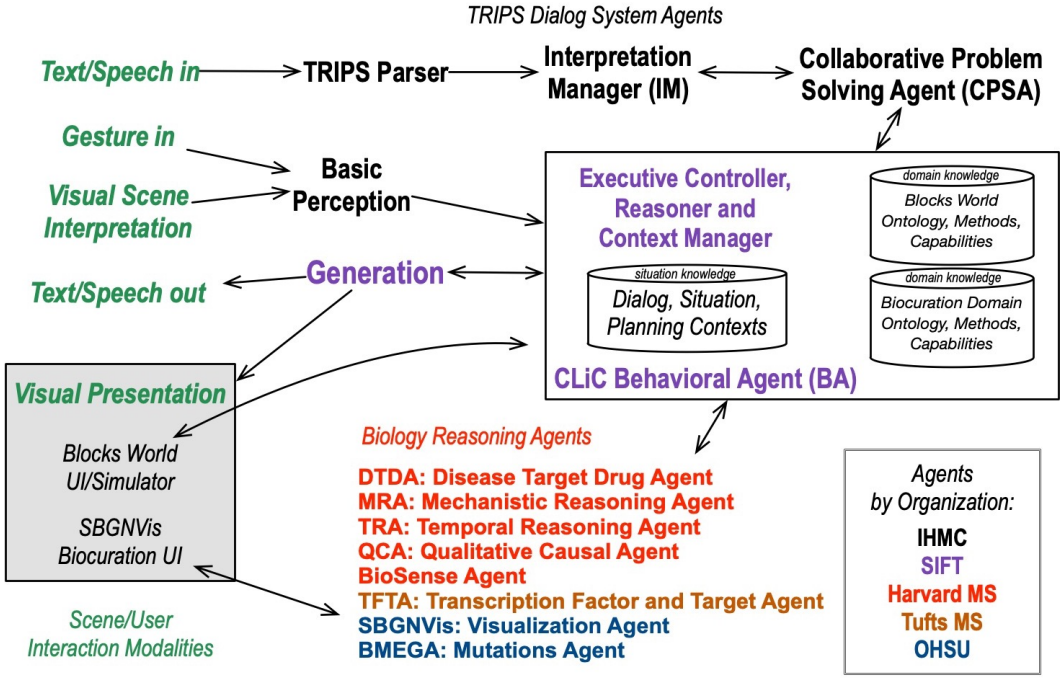


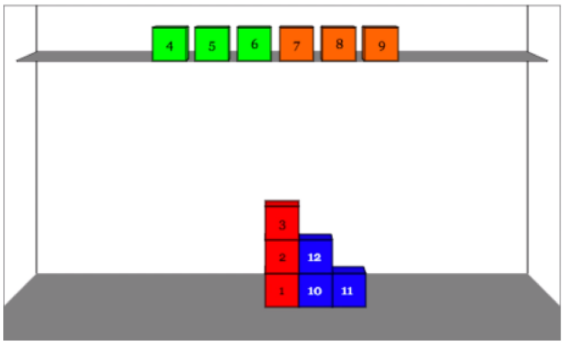
Figure 1. Overview of the common architecture for context-driven dialog.

When configured and used for the Blocks World, the overall system is called CABOT. In the second domain, Biocuration, the BOB system assists research biologists with the discovery of information about signaling pathways, as well as simulation and analysis of reactions in those pathways. BOB is a much more complex system, with some 60 capabilities, developed in collaboration with systems biologists in Peter Sorger's lab at Harvard Medical School, Brent Cochran at Tufts University School of Medicine, and a group led by Emek Demir at Oregon Health and Sciences University.

In the Blocks World, the BA is responsible for goal interpretation and action, maintaining a model of the shared visual scene, constructing plans to achieve goals, checking the viability of those plans, and selecting actions to further those goals. For example, In the dialog on the right side of Figure 2, shown as performed in our browser-based interactive simulator, there are 12 blocks in total that start on the shelf shown at the top of the screen. When CABOT checks the viability of the user's stated goal, it reports a resource shortage ("*We don't have enough blocks.*"). Since the user's revised goal is phrased as a directive, the system assumes that it should move the blocks to build the figure. In the dialog on the left, when the user makes a mistake in placing the second block (marked with $*$), CABOT responds by suggesting an action that would get the plan back on track.[4] Depending on the situation and phrasing of the goal statement, either party can move the blocks. Context-dependent phrases like "*Keep going*" and "*Add another*" are interpreted as references to perform another plan step while "*all red blocks*" is interpreted as referring to appropriate scene elements excluding those blocks that are already placed and part of the goal configuration.

4. Friedman et al. (2017) have discussed this Blocks World example at greater length.

User: Let's build a staircase.
CABOT: Ok
CABOT: Put a block on the table.
User places Coke block.
CABOT: Put a block on the Coke block.
User puts Mcdonald's block on table. *
CABOT: Push the Mcdonald's block and the Coke block together.
User: ok.
CABOT: Put a block on the Coke block.
User: ok.
CABOT: You built it!



User: Let's build a 5 step staircase.
CABOT: We don't have enough blocks.
CABOT: What should we build?
User: Build a row.
CABOT puts B10 on table
User: Keep going.
CABOT puts B1 on table
User: Add another.
CABOT adds B11
User: Make a staircase.
CABOT puts B12 on B10
User: Put all red blocks on B1.
CABOT: We built it!

Figure 2. Two blocks world dialogs, one where a person manipulates real blocks and the other using our Web-based interface. In the first one, the person moves the blocks. In the other, the system does these actions.

Tables 2 and 3 show examples of Biocuration dialogs. In the this domain, the BA does less planning and, rather than invoking agents to perform actions on the world, it identifies when user questions or objectives can be achieved by tasking one or several domain specialist agents developed by the team biologists. In Biocuration, the representation of the visual scene is limited to shared graphical objects, such as those used in signaling network diagrams.

3. Approach

In this section, we provide a more detailed description of the architecture of CLIC's Behavioral Agent (BA), Figure 3 shows the module's internal structure. The BA is responsible for:

- Interpreting the parser's logical forms as grounded and operationalizable descriptions of goals, including directives, questions, and assertions;
- Evaluating user-provided goals and accepting them as shared objectives;
- Planning how to achieve goals and evaluating their feasibility;
- Executing or suggesting next steps when a response to the user is required;
- Formulating responses to the user when actions or plans succeed or fail.

We give examples to illustrate aspects of the CLIC architecture that let it ground references to entities and draw inferences from contextual knowledge during communication and collaboration.

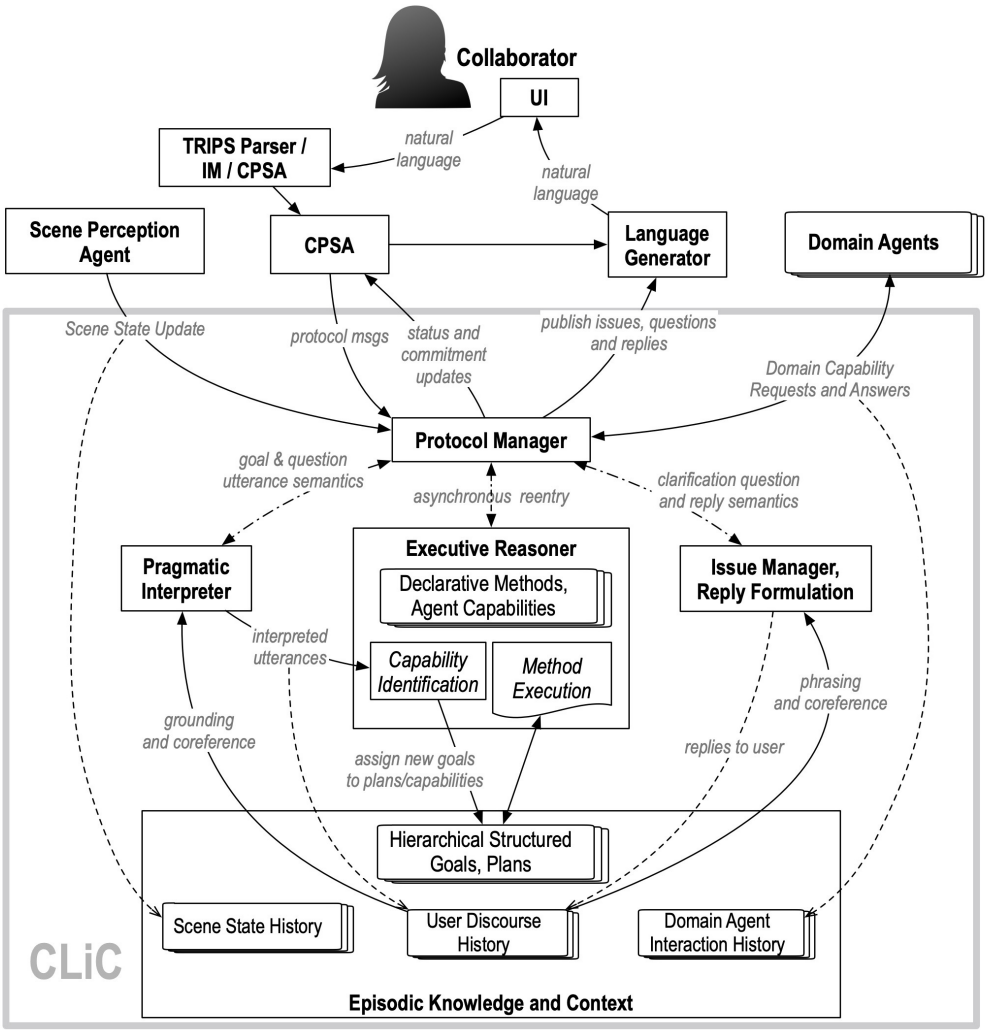


Figure 3. Architecture of the Behavioral Agent for coordinating dialog and action.

3.1 Functions of the Behavioral Agent

The BA interprets the representation of the user's statement or question through an inferential rewrite process that produces a form for comparison to internal descriptions of the current world state, the envisioned goal, and preexisting plans or methods. It creates goals for each new user input and selects methods (capabilities) to achieve them. In Blocks World, it engages the executive to determine what next actions might achieve those goals. The system then either proposes an action for the user to take or executes the action itself when it has the initiative or been told to do so. In addition to acting on shared goals, CLIC creates an additional, internal goal for each user input to ensure it responds to the user, either reporting a result or problem, asking a clarification question,

answering a question or proposing/performing an action. The response goal for user questions in Biocuration ensures it produces some type of reply characterizing its success or failure. If the given goal or question was unachievable or unclear, the response is a failure report, possibly accompanied by a clarification question. A reply may answer a question (when information is found, a success), point out an issue, suggest a user action, or ask for a clarification, among other things. We describe clarifications briefly in Section 6.

In the Blocks World, the BA is also does scene interpretation. Block locations and colors are provided by the Basic Perception agent when observing real blocks or by the simulation running in a web UI. The BA uses this information to identify composite object structures made of blocks (rows, columns, etc.) in goals and intermediate plan states. It matches perceived relationships among blocks against descriptions of the expected relationships in those structures to determine achievement status. The system also resolves references from referring phrases to identified grounded objects (e.g., "*the top red block*").

The BA'S Protocol Manager handles all asynchronous messaging interactions with other agents. This coordinates the activities of the Pragmatic Interpreter, Executive Reasoner and Issues/Reply Manager. At the BA'S core is the Executive Reasoner, which was developed around a reactive goal execution model that took many of its operating principles from ICARUS (Choi & Langley, 2018), but is built on top of a reasoning engine called SPIRE, developed by Scott Friedman, that reimplements and extends many basic features of FIRE and Companions (Forbus et al., 2009). The Executive is run every time a change occurs. For example a new message from the CPSA can be a new shared goal to consider (**evaluate**) or a request for a status update (**what-next**). BA responses to the former indicate the acceptability of the proposed goal, while responses to the later include both a goal status and the semantic content of a reply to be generated as text. Other agents interacting with the BA send changes to the state of the world or answers to requests or queries from the BA itself, responses to which indicate the successes or failures of those subgoals. The Executive incorporates any new information produced by each of these agent messages and then considers whether it can advance any of its goals further as a result.

Of particular note for the work presented here is how CLIC maintains different contexts, which it uses to record multiple episodic timelines: its interpretations of the changing scene, the language discourse history, and the current goal and envisioned plan. Other key elements include:

- **Pragmatic Interpretation** of user statements, questions, and proposed goals (as parsed by TRIPS), which produces a canonical form with references grounded in the different contexts;
- **Capability Identification**, which compares interpretations to the available system capabilities using partial matching (in order to give feedback on near misses);
- **Goal Formation**, which creates operational goals based on the interpretation of the user utterance and the set of fully or partially matching capabilities and previous goals;
- **Issue Management**, which handles the representation of failures in responses and the spawning of subgoals for processing interruptions to handle such things as seeking clarification on goal specifics before continuing.

Next we describe how these four elements interact to support the cognitive process of task-oriented collaborative dialog.

3.2 Basic Process Flow

This paper focuses on how goals are formulated or revised given an utterance with respect to the current state of the world, the current plans, and the available capabilities. As described above, the BA determines a representation of composite structures in scenes, and their relationship to intended, possibly shared, goals and steps to achieve them. It develops plans for how to construct composite structures that have been declared as goals (e.g., a staircase), and identifies plausible next steps to suggest to the human collaborator, including corrective actions, when the user has the role of moving the blocks. Once a structure has been identified as a goal, possibly after refinements have been discussed, the BA builds a plan to construct it and may execute actions in the world (if it has the initiative) by invoking actuation agents to move the blocks.

In a previous paper (Friedman et al., 2017), we described how the BA does action selection by using analogical structure matching to localize the current scene state with respect to the envisioned results of individual plan steps. When it finds a matching step, it chooses a next action from the plan, which either the user should perform or it should execute itself. When the result of the user's action does not match the expected next state, but a plan state can be found that is a close partial match that could be reached by a repair action, it suggests the repair (e.g., "*Push them together*"). Finally, there is the Issue/Reply Manager, which determines how the BA formulates replies to the user when goals succeed or fail, as we describe in Section 6.

Figure 4 gives an overview of the BA's processing for the Blocks World, as illustrated in the dialogs of Figure 2. Users can either move blocks or type a sentence. Moves are received by the BA as a state update and create a new state context. Sentences are parsed by TRIPS and interpreted as goals or assertions by the CPSA before transmission to the BA. The Pragmatic Interpretation process includes ontology translation and various kinds of simplification to form actionable goals or goal modifications, and it invokes methods to update its goals and projected state changes. In Blocks World, shared goals are to build structures of blocks, which requires some planning, and changes to the observed world state can cause it to suggest repair actions. On the other hand, Biocuration goals are most frequently created by user questions when seeking information.

When asked by the CPSA to **evaluate** a goal proposed by the user, the BA decides whether to accept that goal by first envisioning the goal state and then looking for a way to achieve it. If there is an issue (e.g., insufficent resources), it rejects the goal and describes the issue. If the BA **accepts** the goal, the CPSA asks **what-next** and the BA can either perform an action to further that goal or suggest one for the user to perform. After attempting to execute its selected actions and either succeeding or failing, the Issue/Reply Manager translates success (including found answers to questions) or failure (with reason) into replies to the user.

Mechanically, when a user-specified goal is accepted by the BA, it is established as a shared goal, and activated so that it is be considered by the BA Executive. To determine acceptability, CLIC first invokes its Pragmatic Interpreter, which does ontology translation and canonicalization of the semantic logical form of the goal as provided by the TRIPS parser. We describe in Section 4 how this process produces a simplified, canonical goal statement that eliminates most of the variation in semantic forms due to linguistic differences in phrasing and produces an actionable description that can be related to system capabilities in the current domain and context.

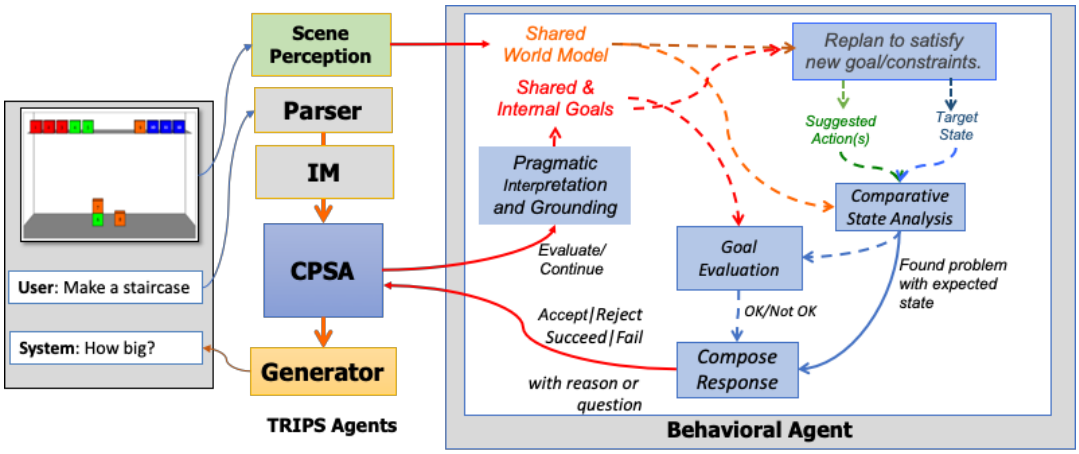


Figure 4. Blocks World reasoning within the Behavioral Agent.

During the interpretation and evaluation processes, contextual reasoning and reference resolution procedures draw on temporally interleaved contextual streams: (a) the objects and structures in the visual scene; (b) the objects in the most recent actions; (c) the entities in semantic descriptions of participants' utterances, including items inferred to be in those descriptions, and (d) recent objectives and planned steps, including the current goal envisionment. Different contexts are queried to find implied references in those contexts that are consistent with the referring phrases. Which contexts are queried depends on the type of the utterance (current-state questions, goals or goal modifications, rejections of the most recent action). We discuss this in the next section.

4. Contextual, Pragmatic Simplification of Parser-Generated Semantics

User's questions and requests can be phrased in a wide variety of ways. Even semantic parser output must be further interpreted to determine how to act. CLIC translates the parser's representation to a canonical form that is consistent with its representation of the operations it can perform. This representation is based on a semantic vocabulary of Elementary Composable Ideas (ECIs) that includes concepts for common actions/events, relations and entity types that appear across many domains (McDonald et al., 2018). The development of a library of ECIs is an important goal of the CWC program (Cohen, 2017).

ECIs have internal structure that describe how their roles are related and, for events, the processes that characterize how relations between roles change. For events, these include **move**, **transfer**, **create**, **build**, **damage**, **destroy**, **perceive**, **emit**, **communicate**, **find**, and **confirm** or **reject**. ECIs for entities describe the habitats in which they can exist and the affordances they provide in those habitats (McDonald & Pustejovsky, 2014). Physical containers, for example, afford a containment relation ('in') when they are closed or open and upright. Conceptual containers are entities with boundaries and an interior (*e.g.*cars, buildings, cities).

We use event ECIs to represent actions to achieve shared goals. CLIC's Pragmatic interpreter uses ontology mapping (from TRIPS logical forms in its native ontology, to one based on ECIs) along with rewrite rules that perform inferences to simplify the variety of surface' sentential se-

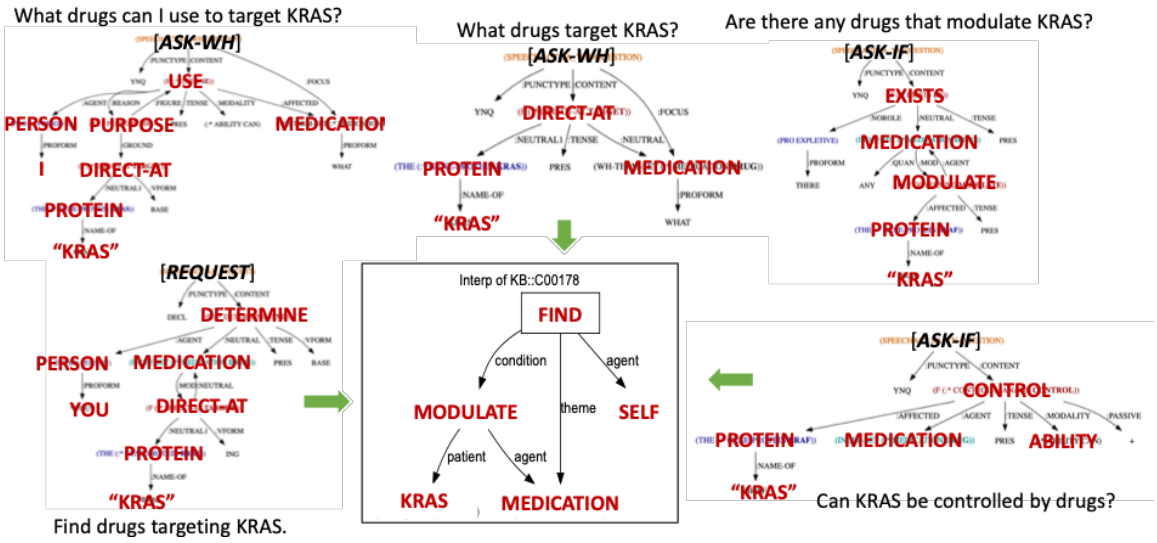


Figure 5. Pragmatic Interpretation reduces surface semantic variation to a description of an achievable goal. Ontological classes are shown in red for emphasis.

mantic forms to ECI representations of possible system goals. Although the parser canonicalizes standard syntactic variations like active vs. passive voice, different polite ways of asking and different uses of prepositions and existentials produce multiple semantic forms with similar meanings.

Figure 5 illustrates why the pragmatic simplification process is necessary, showing some ways one might ask for drugs that control the activity of the protein KRAS, which plays a role in many cancers. In the figure, the different ways of asking "*What drugs target KRAS?*" are accompanied by graphical depictions of the TRIPS semantic descriptions produced by its parser, with the ontological categories highlighted in red, along with the simpler uniform representation CLIC produces. Using a set of inferential rewrite rules in the BA'S pragmatic interpreter, these forms are reduced to one basic goal, in this case finding all known agents of the specified type that act on the given class of object. The rules address situations like:

- **What** *agents* **target** *patient***?** – FIND information about *agents* whose purpose or affordance is to act on certain classes of *patient*.
- **What** *agents* **can I use for** *purpose or entity***?** – FINDS *agents* with specified affordance or one that acts on *entity type*.
- **Find, list, show me, can you tell me?** – COMMUNICATE the result from a FIND information.
- **Can** *agent of type* **do** *function* **to** *patient***?** – A CONFIRM question, but one that can often usefully be answered by supplying the retrieved exemplars when there is more than one.

The standardized representations produced by this process are then grounded by replacing referring expressions with their most likely referents. These are found by searching the system's background knowledge for known terms, as well as the discourse and interaction contexts as described below. Once references are grounded, the full description is matched against available capabilities (actions available to known agents through messaging) to select an action to take or plan method to expand.

Table 1. The capability for finding a drug.

```
(define-capability FIND-TARGET-DRUG ?g :agent DTDA
  :form ((a ?g :isa kb::FIND :agent ont::sys :theme ?drug :condition ?modulate)
         (a ?modulate :isa kb::MODULATE :agent ?drug :patient ?prot)
         (a ?prot :isa ont::gene-protein)
         (wh-term-isa ?drug ont::MEDICATION))
  ...)
```

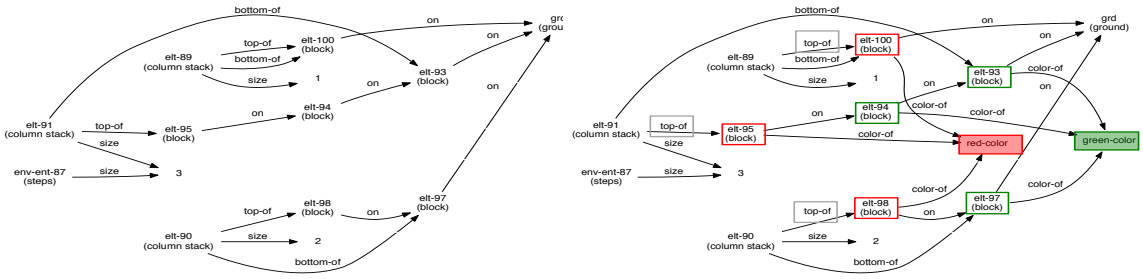
Table 1 shows a simplified pattern (**:form**) for the **find-target-drug** capability. If matched, it results in a query to the DTDA Bio agent that will answer the question about drugs for KRAS. Sometimes a capability will have several such patterns. Most capabilities for Biocuration require the BA to send messages to other agents that return answers or fail with specified issues. These are internalized in the BA'S context history before a reply is formulated for the user. The remainder of the capability definition deals with marshaling the message from the information in the question, handling the answer, and preparing for a response to the user.

At present, there are over 100 capabilities in the Biocuration domain supplied by ten agents and the BA itself. These cover a suite of information gathering, modeling, and analysis functions. Most have optional arguments that establish contextual variations and an ability to handle sets of entities produced by prior answers when those entities are referenced in a new question. We are working on an approach that will let the BA compose and use the results of multiple capabilities to formulate answers to user queries that would otherwise need to be broken into several questions. A simple example is asking *"What kinases are regulators of SMURF2?"* rather than having to ask *"What are the regulators of SMURF2?"* followed by *"Which of those are kinases?"*.

5. Reference Resolution In Different Contexts

Human collaborators often discuss goals at the onset of problem solving, sometimes revising collaborative goals nonmonotonically. This requires a mutual understanding of the goals, enabling collaborators to propose modifying these shared goals, along the way mentioning solution parameters, or solution constituents that may have only been implied but not referred to overtly earlier. We will illustrate how CLIC does this by stepping through the sequence of events and knowledge states in the goal revision dialog illustrated in Figure 6.

The human collaborator first specifies a goal to CLIC, such as *"let's build a 3-step staircase"* or *"build a 5-block stack."* As shown in Figure 4, the system takes a semantic goal description provided by the TRIPS CPSA and interprets it as an ECI expression, in this case to **create** a composite structure. From the phrase *"staircase"* the interpreter identifies the ECI object class **steps**. Given a **size** for that structure, specified or inferred, CLIC can envision a composite relational structure that it can plan to build. This envisionment is a qualitative spatial description of the desired object. Using the constitutive specification in the ECI, the system knows to plan to build a sequence of descending-height **stack** instances, where each of those is comprised of a number of **block**s relationally **on** one another. Figure 6 (a) shows the envisionment for the staircase, with spatial relations omitted.



(a) *Let's build a 3-step staircase.*

(b) *Make the blocks green.*
Make the top blocks red.

Figure 6. CLIC (a) envisions the structure implied by a goal and (b) revises the goal by modifying properties (colors) of referents of *"the blocks"* and then *"the top blocks"*.

If the person then specifies the goal further by saying *"Make the blocks green"*, CLIC interprets this "make" as a newly proposed **change** goal, rather than a **create** because it has interpreted the sentence as *cause the blocks to have the color green*. In this case, the thing to be changed is the goal specification, not the state of the world. CLIC reaches this interpretation by first trying to ground *"the blocks"* in its representation of the visual scene and failing because there are no blocks on the table yet. It then searches its goal envisionment, shown in Figure 6 (a), finds three blocks, and develops a goal with those blocks as the referent. The action thus formed and executed internally adds a color relation for each one, producing a revised envisionment for the shared top-level goal with all green blocks. Subsequent revisions may be nonmonotonic. For example, if the person then says *"make the tops red."*, CLIC's interpreter grounds its interpretation of the phrase *"the tops"* to objects (blocks) satisfying the **top-of** predicate, satisfied by blocks with nothing above them, and asserts the specified color for those objects. Figure 6 (b) depicts the result.

Note that the **block** instances in CLIC's envisionments do not correspond to any specific blocks in the scene and the envisionment is not constrained by the resources available, although it does support domain-level action once mappings to scene objects are made. When told to start building, it uses this envisionment to (1) check for available resources (and articulate any resource problems it encounters) and (2) create a plan to build the structure. Planning and plan execution in this Blocks World is comparatively straightforward, so we will not describe those processes here. This example illustrates how a machine can share a hypothetical, language-grounded goal with a person. The person can specify, refer to, and propose revisions to parts of the goal and have the system make the appropriate changes. This is qualitatively different from grounding language in a shared visual setting, but CLIC uses the same knowledge representations for internal goals and external situations, so it can handle both contexts using the same reasoning processes.

In biology, there are many kinds of regulation. Transcription factors play a role in gene transcription, while kinases are important in phosphorylation. Separate data resources are available to the TFTA agent that detail what regulates what in each case, and it provides domain-specific capabilities to answer questions about those topics. Other agents similarly provide capabilities to answer queries or perform other actions, like building and evaluating models, from which the BA must

Table 2. References (underlined) to entities in prior question (2) or answers (3 and 4).

User: What is SMURF2?	(1)
BoB: SMURF2 is an E3 ubiquitin-protein ligase which accepts ubiquitin from ...	
User: What regulates **it**?	(2)
BoB: By 'it' I assume you mean SMURF-2.	
Transcription factors that bind the SMURF-2 gene: PBRM1, SMAD2, and TBL1XR1	
Transcription factors that regulate SMURF-2 from the literature: TSC22D3	
Kinases that regulate SMURF-2 from RNAi knockdown in GEO: AKT1, CDK19, CDK8, CDK9, ...	
Genes that regulate SMURF-2 from the literature: BMP7, CFH, COL2A1, CTNNB1, IKBKB, ...	
User: **Which of those transcription factors** also regulate STAT3?	(3)
BoB: From among PBRM1, SMAD2, TBL1XR1, and TSC22D3, I found that SMAD2 regulates STAT-3.	
User: **Which of the kinases** are in the MAPK signaling pathway?	(4)
BoB: I found those 12 kinases in 2 matching pathways: ...	

choose. The BA selects capabilities to use by matching its canonicalized representation of each user question or statement against the patterns representing the available capabilities. The representation of an interpreted question typically specifies the type of entities sought and the role they play in some condition relating them to other specific named entities. The action to be taken is selected based on a best match between that representation and a capability pattern. For questions like *"What transcription factors regulate STAT3?"* or equivalently *"What transcribes STAT3?"*, the result is a request for the FIND-TF capability, which returns the information needed to formulate a natural language answer to the question. If the question does not directly specify a desired result type (e.g., *"Which of those regulate STAT3?"* does not say that *"those kinases"*), then CLIC must determine the types of possible referents to find a suitable capability. The referents found for *"Which of those"*, together with the capabilities that match the question, constrain the action to pursue.

Consider the dialog with BOB shown in Table 2, which includes three references to previous questions or answers. In the second question (2), *"it"* is recognized as referring to the focus of the previous question. In the next question (3), *"those transcription factors"* refers to the entities in the immediately preceding answer, but only those of the specified type, which appear on two different lines of the answer. In the last question (4), *"the kinases"* refers to another set of the proteins mentioned in the answer to (2), rather than the proteins in the answer to question (3), since only (2) mentions entities of the correct type (kinase). Candidate referents are found by a reverse chronological search through the semantic representations of questions and their answers. Each such representation is stored in a context data structure, CLIC's record of the discourse between the two parties to the conversation. The reference identification mechanism that searches these contexts uses recency as a preference, and the semantic type and number information provided in the referring expression as search constraints. When necessary it uses the semantic role (e.g., agent vs. patient) played by the referring expression as a constraint on the search .

Referring expressions only sometimes provide the types of the entities intended to be referenced. When a question requests additional information about entities in a previous answer, but does not directly name the type of those entities (*"Which of those are ..."*, *"Do any of them ..."*, etc.), the

Table 3. Referents (underlined) influencing capability selection in lines (6) and (8).

User:	What miRNAs regulate STAT5A?	(5)
BoB:	I found 34 miRNAs targeting STAT5A: MIR-1180-5P, MIR-1197, MIR-124-3P, ...	
User:	**Which of those** target CDH1?	(6)
BoB:	Of those 34 miRNAs, I don't know of any miRNAs targeting CHD1.	
User:	What transcription factors are regulated by STAT5A?	(7)
BoB:	I found 131 transcription factors regulated by STAT5A: ARID5B, ... and ZNF776.	
User:	**Which of those** are regulated by STAT3?	(8)
BoB:	Of those 131 transcription factors, I found 53 transcription factors regulated by STAT3: BCL6, CEBPA, CTCF, ... and ZNF395.	

BA considers the types of all the entities in the previous answer as possibly satisfying the role played by the pronoun (*"those"*). Capabilities that match the interpreted question with one of those types substituted for the pronoun each offer a way to satisfy the goal — the types of the referents that are found impact the BA'S selection of an action. Table 3 illustrates this idea. In (6), *"those"* refers to the microRNAs in the answer to (5), so the system uses the FIND-MIRNA capability to answer. In (8), *"those"* refers to the transcription factors returned as the answer to (7), so CLIC invokes the FIND-TF capability to answer the question. This demonstrates how reference resolution can result in the use of different capabilities (**FIND-MIRNA** vs. **FIND-TF**) despite the fact that the questions do not explicitly mention the type of entity desired and are otherwise semantically very similar.

6. Issue Management and Clarification Dialogs

Many things can go wrong during the CLIC's processing of a goal, and the system must report failure with some explanation or ask a clarifying question. Failures can result from interpretation problems or bioagent issues. Ambiguities can arise during interpretation or action selection, as words can be misspelled and require confirmation when corrected, terms may be used imprecisely and require disambiguation, and so forth. Sometimes context supply the missing information, but other times the system must ask for additional details in order to form a coherent goal. These issues can arise from different parts of the BA'S processing or during another agent's handling of its requests. All interrupt the normal process of formulating a successful response and may lead to a subdialog.

To prepare these responses, we developed the **Issue Manager**, which is triggered when problems arise in other parts of the system. While not a full meta-cognitive layer (Cox et al., 2016), this module is responsible for handling failure or problem conditions when raised elsewhere in the architecture, producing responses to the user that may lead to subdialogs. It is responsible for establishing the context in which answers to system-generated clarification questions are interpreted and for using those answers to resume the task that was interrupted.

Figure 7 illustrates the overall process of handling a clarification dialog, including the messages passed to other parts of the TRIPS architecture. In this case, the user has asked a question about a class of microRNAs but has not specified the full name of a particular microRNA, only the prefix that indicates a family of such RNAs. The BA initially has interpreted and forwarded the question to the TFTA agent, but it replied with a failure and an accompanying request to resolve the ambiguity

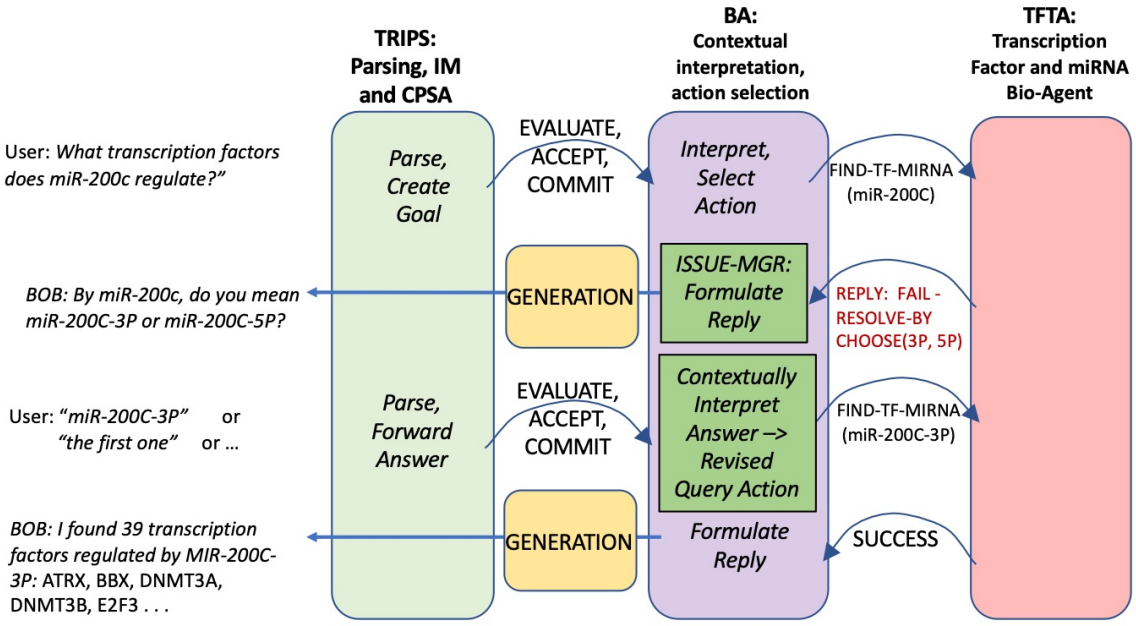


Figure 7. Information flow in BOB while processing a clarification question. The BA handles the generation and processing of a question to the user and then refines its query to TFTA.

between two possible referents. Failures of this kind cause the Issue Manager to spawn two internal goals: one to formulate and ask the user a question and another to handle the user's response. In the figure, the system asks for clarification about which specific member of the microRNA family was meant by the term *"miR-200c"*. The user's reply is interpreted in the context of that question and spawns a revised suboal to resubmit the user's question to TFTA. To this end, the BA modifies the user's original question with the result of the clarification dialog and executes a revised request to TFTA, with the selected entity replacing the representation of the family name. The system then reports the answer from that query.

Asking the clarification question is straightforward, as CLIC just needs a choice to be made among possible referents. On the other hand, interpreting the answer in the context of that clarification question is more complicated. First, even when making a simple choice between two entity names, as with *"Did you mean miR-200C-3P or miR-200C-5P?"* the user might name one of them or might say something like *"I meant the latter"* or *"The second one"*, in which case the system uses the context of the question to interpret what this reference to a place in a sequence.

7. Evaluation: Testing Performance with Representative Users

We have demonstrated and evaluated the CLIC agent in both Blocks World and Biocuration at different stages of its development. During the second year of the project, we tested four naive users solving problems with the Blocks World system, CABOT. They saw a three-minute video example of its use and were then asked to build sample figures from pictures they were given. Approximately

60% of the utterances were interpreted and responded to properly at that time. This study motivated an architectural redesign that led to the version of CLIC described here, using distinct interpretation phases for ontology translation and pragmatic interpretation/operationalization. This let the system handle a much wider variety of phrasings.

More comprehensive tests were performed on BOB by our partners at Harvard Medical School (Ben Gyori), the Tufts School of Medicine (Brent Cochran), and OHSU (Emek Demir). For this study, eight biologists who had not seen the system were asked to solve problems. We gave each participant 30 minutes to solve each of two problems in which they were asked to research and model short but unfamiliar signaling reaction sequences. As a baseline for comparison, some participants performed one of their problems using regular Web resources. The results were quite encouraging. Summing over all problems attempted using the system, 491 questions or statements were posed to the system. The system replied appropriately to 72% of those queries and directives. Several researchers were able to solve problems using the system that they would not have been able to handle in the time allotted using Web resources alone.

Several things were clear from these preliminary studies. First, accurate interpretation is critical for the system to produce meaningful responses to scientific questions. Second, users ask questions in different ways, so reasoning is needed to determine the goals that underly each question. Subjects were not told that the system could handle references to prior questions and answers, so they avoided doing it entirely.

A second study by evaluators from MITRE Corporation with a biology background tested the system to stress its capability to respond given alternate phrasings, handle questions requiring reference resolution, and produce informative responses to misspellings, incorrect words, and missing system capabilities. These evaluators are periodically reviewed the system's conversational capabilities to determine its level of robustness and user support, as well as its ability to give good feedback when it cannot fully answer. They developed five hallmarks for desirable system behavior:

- **Robustness:** Interacts with users to repair conversational errors and breakdowns;
- **Provides Rationale:** Provides reasons for its behavior and explain its failures;
- **Context Awareness:** Uses context to improve understanding and make communication more efficient;
- **Habitability:** Enables people to use language naturally to express themselves within the constraints imposed by the system;
- **Bi-directionality:** Contributes to the conversation, rather than simply responding to directives and questions.

They chose 124 questions to probe the system's coverage of these issues, some of which had not been fully addressed in our development at the time. They divided questions into groups that exercise each of the hallmarks. Areas where BOB did not perform well had not been a focus of our efforts to that point. In particular, at the time of this test we had not addressed how BOB would use multiple capabilities to answer questions or recognize and report on the many of the sources and types of failures that it might have encountered. BOB also did not give useful feedback when asked questions that did not match any of existing capabilities, instead simply saying "*I don't know how to do that.*"

Table 4. Summary of MITRE's usability testing of the dialog system.

Category	Capability being tested	Total	Passed	%
Robustness	Representative questions	22	22	100%
	Alternative ways of asking a question	8	5	63%
	Imperatives	4	4	100%
	Minor word changes	10	9	90%
	Typos (not in entity names)	6	5	83%
	Alternative biology terminology	7	5	71%
	Entity synonym/family/complex	4	4	100%
	Misspelled entities	8	7	88%
Robustness Total		**69**	**61**	**88%**
Context Awareness	Use references to prior answers	6	1	17%
	Use references to prior questions	4	4	100%
Context Awareness Total		**10**	**5**	**50%**
Provides Rationale for Errors and Missing Capabilities	Requests for related capabilities	5	1	20%
	Queries requiring multiple capabilities	11	4	36%
	Queries with unrecognized entities	5	1	20%
	Questions about system capabilities	5	1	20%
Rationale Total		**26**	**7**	**27%**
Habitability	Language consistency	12	9	75%
	Follow-up Questions	3	1	33%
Habitability Total		**15**	**10**	**67%**
Bi-directionality	Makes relevant, unsolicited suggestions	**4**	**3**	**75%**
	Totals	**124**	**86**	**69%**
	Legend:	<25%	25-60%	>60%

Table 4 presents a more complete table of results from the spring 2019 tests. Overall, 88% of questions asking for basic capabilities were handled correctly and 100% of questions that referenced entities from a prior question were answered appropriately. Also, at the time of testing, only certain bioagents were capable of using information from prior answers, so only one of six questions tested use of prior results was successful. Rectifying that issue required the introduction of a uniform shared representation for biological entities that all agents now use. We also added new capabilities to address questions about the system's own knowledge, such as what data sources were used, and what tissues or diseases could be asked about. The system can now provide rationale for failures and identify general question types that make it possible to suggest rephrasings.

In the spring of 2020 we carried out our most ambitious test. We recruited a small group of biologists to use BOB to work on their own research problems. They worked remotely, using a Dockerized version of the system, and were able to upload their own data files describing gene expression values in different cell line, sometimes thousands of genes and hundreds of cell lines. In a future paper we will describe how this data is presented to the user, who can cluster it along different dimensions and select from it to carry out standard BOB questions. The results are still being analyzed, but anecdotally we know that some biologists were so successful during the testing period that they have asked to keep using the system going forward.

8. Discussion and Conclusions

In this paper, we discussed the architecture of the dialog reasoning agent CLIC as used in a multi-domain, task-oriented dialog system. The architecture draws on principles found in other reactive goal achievement systems, but also includes components that support collaborative dialog. We emphasized several novel characteristics of the architecture that are motivated by important features of such systems:

Many kinds context can be referenced during goal-oriented discussions, including the *discourse history*, a *shared visual scene*, and *elements of goals, envisioned results and potential plans*. We presented examples where CLIC resolved references to elements of shared scenes and envisioned goals, even when those elements had not been explicitly mentioned in the discourse.

Pragmatic interpretation and capability identification are key parts of deciding how to respond to a user's stated questions and goals. Including these steps let us separate the translation of the many ways user requests are phrased from the operationalization of those requests, producing more uniform goal statements to use in deciding how to act. We described a number of language patterns that reduce raw parser semantic output to a much smaller set of goal types.

Resolving pronominal references is a co-agreement process among the current set of contexts, restrictions specified in user requests, and the set of actions or capabilities the system could use to respond. We illustrated this using examples of different contextual influences on the reference resolution process. In one case, semantically typed references to elements of prior answers (e.g., *"the kinases" vs. "the transcription factors"*) resulted in different capabilities being selected to respond to otherwise similar questions. In another case, an untyped reference, *"those"*, once grounded in the discourse context, determined the capability invoked.

Problems can arise during the agent's internal reasoning processes that CLIC must report to the user. Some of these can initiate subdialogs to interactively refine goals or methods. Problems include interpreting user utterances with spelling errors, misused words, or imprecise references, operationalizing goals under consideration, and problems during planning and execution. A meta-reasoning system should handle interactions with users when these problems occur, regardless of where they occur internally. CLIC's Issues Manager supports some interactions of this type, including asking clarification questions and refining goal/plan dialogs, but further work is needed.

CLIC is still under active development, both from a domain capability and reasoning perspective. Our evaluations suggest that the system can be improved by better explanations of why it cannot answer a question and it could be more helpful by suggesting rephrasings of user questions. We plan to improve the system's representation of its own internal processes so that it can describe its own failures better.

We are also exploring ways to understand what users intended when their questions could not be mapped to capabilities, in order to better express what was wrong and what they might do instead. In our final test, practicing biologists brought their own data to a Dockerized version of BOB and used it to advance their own research. This suggests that with further hardening of the system, as well as additional agents and capabilities that real users want, we should be able to to make a version of our biology system, BOB, available for biology researchers to use within the year.

Acknowledgements

We wish to thank our many collaborators at IHMC, Harvard, Brandeis, Tufts, and OHSU for their participation in this work. This work was supported by Contract W911NF-15-C-0238 with the US Defense Advanced Research Projects Agency and the Army Research Office. Approved for Public Release, Distribution Unlimited. The views expressed are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

References

Allen, J., Ferguson, G., & Stent, A. (2001). An architecture for more realistic conversational systems. *Procedings of the Sixth International Conference on Intelligent User Interfaces* (pp. 1–8). Santa Fe, NM: ACM Press.

Blaylock, N., & Allen, J. (2005). A collaborative problem-solving model of dialogue. *Proceedings of the Sixth SIGdial Workshop on Discourse and Dialog* (pp. 200–211). Lisbon, Portugal.

Bobrow, D., Kaplan, R. M., Kay, M., Norman, D. A., Thompson, H., & Winograd, T. (1977). A frame driven dialog system. *Artificial Intelligence*, *8*, 155–173.

Cassell, J., Bickmore, T., Campbell, L., Chang, K., Vilhjálmsson, H., & Yan, H. (1999). Requirements for an architecture for embodied conversational characters. In N. Magnet-Thalmann & D. Thalmann (Eds.), *Computer animation and simulation'99*, 109–120. Milano, Itally: Springer.

Choi, D., & Langley, P. (2018). Evolution of the icarus cognitive architecture. *Cognitive Systems Research*, *48*, 25–38.

Cohen, P. (2017). Context in communication. *Proceedings of the AAAI Spring Symposium on AI for the Social Good* (pp. 303–306). Stanford, CA: AAAI Press.

Cohen, P. R., & Levesque, H. J. (1990). Intention is choice with commitment. *Artificial Intelligence*, *42*, 213–261.

Cohen, P. R., & Perrault, C. R. (1979). Elements of a plan-based theory of speech acts. *Cognitive Science*, *3*, 177–212.

Cox, M. T., Alavi, Z., Dannenhauer, D., Eyorokon, V., Munoz-Avila, H., & Perlis, D. (2016). Midca: A metacognitive, integrated dual-cycle architecture for self-regulated autonomy. *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence* (pp. 3712–3718). Palo Alto, CA.

Forbus, K. D., Klenk, M., & Hinrichs, T. (2009). Companion cognitive systems: Design goals and lessons learned so far. *IEEE Intelligent Systems*, *24*, 36–46.

Friedman, S., Burstein, M., Rye, M. J., & Kuter, U. (2017). Analogical localization: Flexible plan execution in open worlds. *Proceedings of the 2017 ICCBR Computational Analogy Workshop* (pp. 33–42). Trondheim, Norway.

Grosz, B. J. (1980). Utterance and objective: Issues in natural languge communication. *AI Magazine*, *1*, 11–20.

Grosz, B. J., & Kraus, S. (1996). Collaborative plans for complex group action. *Artificial Intelligence*, *86*, 269–357.

Hobbs, J. R., & Evans, D. A. (1980). Conversation as planned behavior. *Cognitive Science*, *4*, 349–377.

Khatri, C., Venkatesh, A., Hedayatnia, B., Ram, A., Gabriel, R., & Prasad, R. (2018). Alexa Prize – State of the art in conversational AI. *AI Magazine*, *39*, 40–55.

Lochbaum, K. E. (1998). A collaborative planning model of intentional structure. *Computational Linguistics*, *24*, 525–572.

Madotto, A., Wu, C.-S., & Fung, P. (2018). Mem2seq: Effectively incorporating knowledge bases into end-to-end task-oriented dialog systems. ArXiv:1804.08217.

McDonald, D., Burstein, M., & Pustejovsky, J. (2018). Assembling the ECIpedia: Refining concepts in context. *Poster collection: Sixth Annual Conference on Advances in Cognitive Systems*. Stanford, CA.

McDonald, D., & Pustejovsky, J. (2014). On the representation of inferences and their lexicalization. *Advances in Cognitive Systems*, *3*, 143–162.

Rich, C., & Sidner, C. L. (1998). Collagen: A collaboration manager for software interface agents. In S. Haller, S. McRoy, & A. Kobsa (Eds.), *Computational models of mixed-initiative interaction*, 149–184. Dordrecht: Springer Science & Business Media.

Schank, R., Goldman, N., Rieger, C. J., & Riesbeck, C. K. (1973). MARGIE: Memory, analysis, response generation and inference in English. *Proceedings of the Third International Joint Conference on Artificial Intelligence* (pp. 255–261). Stanford, CA.

Steedman, M., & Petrick, R. P. (2007). Planning dialog actions. *Proceedings of the Eigth SIGDIAL Workshop on Discourse and Dialogue* (pp. 265–272). Antwerp, Belgium.

Traum, D. R. (2000). 20 questions on dialogue act taxonomies. *Journal of Semantics*, *17*, 7–30.

Wahlster, W. (Ed.). (2000). *Verbmobil: Foundations of speech-to-speech translation*. New York: Springer-Verlag.

Williams, J., Raux, A., & Henderson, M. (2016). The dialog state tracking challenge series: A review. *Dialogue & Discourse*, *7*, 4–33.

Winograd, T. (1973). A procedural model of language understanding. In R. Schank & K. Colby (Eds.), *Computer models of thought and language*, 152–186. San Francisco: W. H. Freeman.

Woods, W. A., Kaplan, R. M., & Nash-Webber, B. (1972). *The Lunar Sciences Natural Language Information System: Final Report*. Report 2378, Bolt Beranek and Newman, Cambridge, MA.

Young, S. (1991). Using semantics to correct parser output for ATIS utterances. *HLT '91: Proceedings of the Workshop on Speech and Natural Language* (pp. 106–111). Pacific Grove, CA.