Dialog Management for Data Communication

Eric Saund
CHAT Area

v1 2016/07/08

1 Abstract

This document discusses dialog management for communicating data, such as telephone numbers and addresses. The project has scientific and technical goals. The scientific goal is to build a computational model that accounts for observed data in human-to-human conversations, and that motivates a technical implementation. The science should respect and draw from Cognitive Science and Sociology, especially Conversation Analysis. The technical goal is to build a naturalistic a conversational agent that is easy and effective for people to interact with to conduct useful information exchanges with computers and robots. This document is being developed in conjunction with the TPN (Tell Phone Number) application within the CHAT (Computer Human Agent Technology) group.

2 Topic Data Model

Dialog is carried out by human or machine actors, or *agents*. Agents are concerned with information in domain of interest, which we call *Topic Data*. Topic Data is distinguished from other content of a conversation that are concerned with managing the conversation and the roles and relationships of the participants.

Each agent brings a computational or mental model of the domain, which we call a *topic data model*. In the case of telephone numbers, the topic data model is quite simple in structure, consisting of a sequence of digits, some or all of which are grouped and labeled as distinct segments or fields. We use the term *data index* to refer to the slots, or locations in the data structure where data values, i.e. the telephone number digits themselves, reside. [1]

U.S. Telephone Number Format

Segment Name:	Area Code	Exchange	Line Number	Extension
Digit Indices:	1-3	4-6	7-10	11-{12, 13, 14, 15}

For the purposes of this study, we confine the Topic Data to a simple structured format of this type. If the dialog management issues can be worked out, then we may extend to less regimentally structured topic data such as org charts, family structures, product data, commercial services, and whatnot.

Each agent is proposed to maintain information about their own state of belief about the topic data. In the case of a well-structured topic data domain like telephone numbers, we can assume that all conversants share the same model for the topic data's structure (although some participants may lack names for some of the segments). Therefore, the communication task is constrained to communicating data values registered with their appropriate indices.

[1] "Area Code" is a well known label. "Exchange" and "Line Number" are known by name to some but probably not most human conversants. Some people refer to the Exchange as the "Prefix". Many people refer to the Line Number as the "last four digits" or "last four numbers". Extension is optional and can contain 2-5 digits. When the extension is written out, it usually is indicated by an "x". In conversation, the "x" is normally indicated by the word, "extension". The format for addresses is slightly more complex, but contains standardized fields.

3 Agent Belief Model

A key feature of our approach is one we call a Cognitive Belief Model framework. Under this framework, agents each maintain a model not only of their own belief about the data and other aspects of the conversation, but also what is believed by their conversational partner.

The Cognitive Belief Model stands apart from the main popular prior approaches to dialog management. Many published dialog management systems are designed for a computer to receive single- or multi-utterance queries from a human user, then generate responses based on accumulated desiderata and constraints, but without ever modeling the user's knowledge state or goals [ ]. Other approaches propose that computer agent and human user share goals and perform a joint search through goal space, with each party given the ability to execute moves based on its own knowledge []. Some existing approaches might maintain information about the some aspects of the conversation partner's beliefs implicitly, for example within hidden states in a probabilistic model. We maintain partner beliefs as explicit elements of the model, which supports reasoning, inference, and generalization. [how to prove significance of this, against the claim, "a NN can be trained to do anything"]

The Cognitive Belief Model enables a dialog management system to reason about what information has not only been conveyed, but also how it is understood by the other party. Moreover, it enables and supports protocols, strategies, and devices for correction and repair of misalignments at the topic data level, as well as other conversational levels. As an initial example, consider the following dialog excerpt:

(1) A: `the area code is six five zero`
(2) B: `six four zero`
(3) A: `no that's six five zero`
(4) B: `okay`

In (2), B has echoed the first and third digits of the area code correctly, but the second digit incorrectly. This enables A to detect that B believes incorrect information. In (3) A issues a correction. In (4), B then echoes a confirmation that they received A's correction utterance. Should A trust that B has it right now? The correctness of B's belief about the area code cannot be established with the information given. But equipped with a model of B's previous stated belief and how it might or might not have been corrected upon hearing (3), A is in a position to reason about the possibility that B still has the area code wrong, and perhaps to pursue the matter further.

In our current implementation, we represent belief models in terms of probability distributions over possible values. For example, a Digit model consists of the following data structure containing three elements, where each element is a value-probability pair.

Digit: ((possible-value-1 = v_1, p_1),
(possible-value-2 = v_2, p_2),
(value-unknown = v_?, p_u))

v1 and v2 are logical counterparts to digits in the set {'zero', 'one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight', 'nine'}. The value v_? indicates the assertion that the value of the Digit is known to be unknown. p_1, p_2, and p_u are probabilities that collectively sum to 1. Thus a Digit can represent for example, the belief that the agent knows that they do not know the value of the digit (v_1 = 'seven', p_1 = 0, v_2 = 'three', p_2 = 0, p_u = 1.0); or, that they believe the digit is a four but maybe it could be a five (v_1 = 'four', p_1 = .8, v_2 = 'five', p_2 = .2, p_u = 0). (In the first case, the values assigned to v_1 and v_2 can be

arbitrary because their respective probabilities are 0.)

4 Dialog Levels

From literature and observations, we adopt the view that conversational exchange can be categorized into three Dialog Levels:

- The *Topic Data* level pertains to the information ostensibly intended to be communicated. In the case of telephone numbers, the topic data is simply the digits of a telephone number.
- The *Dialog Management* level is concerned with the back-and-forth mechanics of ensuring that the conversation flows smoothly, for example that speakers take turns, that information is presented in manageable chunks, that information is being heard and understood, and that informational elements drawn from the senders' topic data model are placed correctly in the corresponding slots of the receivers' topic data model.
- The *Roles & Interpersonal* level is concerned with the role of each party with respect to the conversational purpose, and with the relationship between the parties. Aspect of Role include epistemic status (how much knowledge about topic data the participant possesses), obligation to provide or receive information, and constraints the participant brings such as being in a hurry, and obligation to make sure the information is accurate. Aspects of Interpersonal relation include relative epistemic status, deference, politeness, cooperativeness vs. antagonism, and trust.

By its lexical and prosodic content, each utterance in a conversation may incorporate input from each of these dialog levels, and similarly impact the receiver on each of the levels.

4.1 Dialog Management Level

The Dialog Management Dialog Level is concerned with operating and adjusting conversation protocols. Protocols are implicit rules or guidelines for interpreting and producing utterances in context. Dialog Management protocols allow for dynamic adaptation of various parameters that govern efficiency and accuracy of information transfer. And they provide libraries of cues to support social functions for the Roles and Interpersonal Dialog Level. In our computational model, we focus on [N] main Dialog Management parameters.

- **Communication Channel** is concerned with the signal and linguistic qualities of the acoustic (or text) medium. Are spoken words heard clearly and understood by the listener? Communication channel incorporates acoustic signal quality, speaker/hearer language and accent compatibility, and signal delay.
- **Readiness** Agents can signal that they are prepared to engage in data communication, or else are busy and the partner should wait.
- **Turn Ownership** is a belief state held by each party about who should be speaking presently, or next. In smooth conversation, turn is handed back and forth in accordance with conventions and protocol. We model turn as a probability distribution over three states, self, partner, either. At some points in a conversation, turn is clearly handed back and forth between the speaker and the conversation partner (1.0, 0.0, 0.0)/(0.0/1.0/0.0). At other times both parties may want to say something

(.8,.2,0)/(.2,.8.0) and must negotiate who gets to speak next. And a other points, a pause or quiescence may be reached whether each party is contemplating and either one is entitled to speak when ready (0.0, 0.0, .5)/(0.0, 0.0, .5).

- **Control** refers to conversation segments that address the goals or concerns of one party or the other. Control operates at a longer time scale than a single turn. For example, a set of back-and-forth turns may focus on answering a question that one participant asks. The party holding Control retains priority of Turn Ownership at potential segment ending points, and normally issues a Dialog Management level signal utterance to relinquish Control. [Walker and Whittaker, ACL '90]

- **Chunk** size is a parameter that governs how much topic data to convey per utterance. This assumes that one utterance is followed by a pause, into which the conversation partner may issue an utterance of their own such as a confirmation that the information was received. It is possible to rattle off all ten digits of a telephone number at once. If in a hurry, this might be done. If the receiver is taking down the number on a napkin, the pace might best slow down to one digit at a time.

- **Handshake** modulates the quantity of confirmations, checks, elaboration, and detail folded into the protocol. For example a low handshake conversation might have the sender speaking only the telephone number's digits, while a high-handshake level would have the sender articulating segment names or other indicators of each digit's data index, while the information receiver might speak back each digit to send confirmation that it was received and to check its correctness.

4.2 Roles & Interpersonal Level

The Roles & Interpersonal Level of conversation includes aspects of the participants' competencies, personalities, social status, relationship, and attitudes toward one another. For the purposes of building a naturalistic conversational agent, it may become important at some point to intentionally design aspects of personality and character into spoken words and intonation. But data communication is a routine, transactional, low emotion task. While a service agent certainly should be polite, we sidestep Roles & Interpersonal issues, with one major exception.

One aspect of the Roles & Interpersonal Level is critical to dialog management, namely, which party commands which knowledge. This is called *epistemic status* in the CA literature [Heritage, 2012a]. We conceive the two parties in a data communication task as the *Topic Information Sender* (**TIS**) and the *Topic Information Receiver* (**TIR**). At the outset, TIS holds high epistemic status by virtue of knowing the telephone number (K+), while the receiver presumably does not know the number (K-). As discussed below, this knowledge gradient gives rise to dialog patterns such as:

(1)	TIS: `six five zero`	TIS presents digits
(2)	TIR: `was that six five zero?`	TIR checks whether he/she understood them
(3)	TIS: `yes`	TIS confirms or corrects

During utterance-by-utterance interaction, dialog acts may assume localized assertions of knowledge possession and relative command, otherwise known as the utterance's *epistemic stance*. For example, the utterance, "The area code is six five zero" conveys confidence and superior knowledge on the part of the speaker, while "Was that six five zero?", and "Six five zero, right?", both convey knowledge deference to the listener.

We note a subtlety about epistemic status with regard to what knowledge is possessed by which party. While the actual data is presumably known by TIS, it is TIR who possesses knowledge of their own belief and confidence in that belief. Thus we may observe shifts in epistemic stance throughout the dialog as the Topic Data shifts from the telephone number as known to TIS, to TIR's belief *about* the telephone number. For example, in this dialog segment, TIS declares topic information with a K+ epistemic stance, then shifts to a knowledge-deferential K- epistemic stance with issuing a followup question:

(1) A: `the area code is six five zero`
(2) B: `okay`
(3) A: `did you get six five zero for the area code?`

Examples such as this are drawn out in more detail in Section 7 after the full vocabulary of DialogActs is introduced.

5 Natural Language Processing and Logical Forms

Note: This section is essential for understanding the mechanics of how the naturalistic dialog manager is implemented. But for those interested mainly in the underlying theory, the details can be skimmed or skipped.

Natural language is boundlessly rich and complex. Even for the relatively narrow universe of data communication tasks, there are many things to say, and there are myriad ways to express the same thing. To manage linguistic complexity, we borrow from the Otto Dialog Management System and use a Semantic Parser to convert between word sequences and semantic expressions called *LogicalForms*. The Otto semantic parser translates between text (both input and output) using a list of *Utterance Template Rules*. This is the syntax for Utterance Template Rules:

UtteranceDesignator DirectionArrow PhraseSequence

where

UtteranceDesignator:	UtteranceCategoryName[LogicalForm]
UtteranceCategoryName:	Predicate
Predicate:	alphanumeric string, usually capitalized
LogicalForm:	Predicate
LogicalForm:	Predicate(arguments)
arguments	argument, argument, argument...
argument:	LogicalForm
argument:	variable
variable:	$N
N:	any string of alphanumeric characters (no spaces), but customarily a natural number
DirectionArrow:	$\leftarrow$
DirectionArrow:	$\rightarrow$
DirectionArrow:	$\leftrightarrow$
PhraseSequence:	space-separted sequence of Word, UtteranceCategorySlot
Word:	alphabetic string
UtteranceCategorySlot:	UtteranceDesignator

A special form of this rule format is:

DialogAct[Intent(LogicalForm)] DirectionArrow PhraseSequence

where

DialogAct is a special value for UtteranceCategoryName, and Intent is one of a small set of predefined Predicates.

Because LogicalForms can contain other LogicalForms as arguments, the semantic representation allows nested structures.

Here's an example,

```
(1)  DialogAct[InformTopicInfo(InfoTypeCat($1), InfoValue(DigitString($2, $3, $4)))] ↔ \
     the {InfoType[$1]} is {DigitCat[$2]}{DigitCat[$3]}{DigitCat[$4]}
(2)  InfoTypeCat[area-code]                                                          ↔ \
     area code
(3)  InfoTypeCat[exchange]                                                           ↔ \
     exchange
(4)  InfoTypeCat[line-number]                                                        ↔ \
     line number
(5)  DigitCat[one]                                                                   ↔ \
     zero
(6)  DigitCat[one]                                                                   ↔ \
     one
(7)  DigitCat[two]                                                                   ↔ \
     two
...
(14) DigitCat[nine]                                                                  ↔ \
     nine
```

The backslash indicates line continuation. The numbers in parentheses at the beginning of each line, (1), etc, are not part of the syntax, but are written for reference in this discussion.

This rule translates between sentences like

```
the area code is six five zero
```

and its corresponding semantic representation,

InformTopicInfo(area-code, InfoValue(DigitString(six, five, zero)))

Line (1) presents a rule which is the special DialogAct type of UtteranceDesignator. This means that the rule applies to speech or text input or output, as opposed to being intermediary constructions.

The type of DialogAct in this example is one of the special Intent predicates, in this case InformTopicInfo. The complete list of Intents is covered below. The LogicalForm arguments to the Intent predicate include the UtteranceCategoryName, InfoTypeCat. InfoTypeCat is a label that includes the set of names of the segments of a U.S. telephone number. The segment names, area code, exchange and so on, are not actual values of the telephone number, but rather the names of "information type" fields, hence the name, InfoType. Cat refers to the fact that the label is an UtteranceCategoryName.

The rules at line (2) though (4) say that the UtteranceCategoryName can take any of three arguments, area-code, exchage, or line-number. Each of these arguments maps between the Utterance Category named InfoTypeCat and a corresponding text string.

Similarly, the rules (5) through (14) map words for digits to Utterance Categories taking an argument indicating the digit value.

The Utterance Categories, InfoTypeCat and DigitCat, are UtteranceCategorySlots in the Right Hand Side (RHS) of Rule (1), as indicated by curly braces, { and }. In the RHS,

they take take free variables, $1, $2, $3, and $4. Therefore, when the text string, "the area code is six five zero" is encountered, the corresponding argument values are filled in in the semantic representation, leading finally to,

InformTopicInfo(area-code, InfoValue(DigitString(six, five, zero)))

One advantage of this approach is that the combinatorically large numbers of segment names and possible digit values can be handled parsimoniously.

A second advantage is that it is easy to express linguistic variants that mean the same thing and hence map to the same semantic representation. For example, another way to tell someone an area code is to say, "well the area code happens to be six five zero".

(15) DialogAct[InformTopicInfo(InfoTypeCat($1), InfoValue(DigitString($2, $3, $4)))] ↔ \
`well the` {InfoType[$1]} `happens to be` {DigitCat[$2]}{DigitCat[$3]}{DigitCat[$4]}

Rule (1) is written to employ the LogicalForm, InfoValue(DigitString(...)) in order to make it easy for software in the processing engine of the dialog management system to parse the semantic expression to detect that some Topic Data is being presented, and that the Topic Data consists of a sequence of digit values.

In this particular way of representing InfoValue LogicalForms for multiple digits, each rule maps a precise number of digits. To support utterances with different numbers of digits, a different rule needs to be written for each number. Thus in the system's actual list of rules, versions of Rule (1) are written for one digit, two digits, on up to ten digits.

An Otto-type semantic parser is not the only way in which the conversion between text utterances and a semantic representation in terms of DialogActs and LogicalForms could be accomplished. But it has proven effective so far.

6 DialogActs for Communicating Structured Data

Under the Cognitive Belief Model framework, for the constrained data communication task, we can identify a limited number of DialogAct categories, and sequences of DialogActs, that capture a large proportion of natural interactions. The categories are obtained by composing five basic utterance purposes, called Intents, with the three Dialog Levels. Each of the five Intents embodies an epistemic stance, and they are chosen by speakers based on epistemic status, specifically the speaker's model of their own and the conversation partner's beliefs about data values, and the holder's confidence in those beliefs.

The regimes for use of these intents can be charted in a 2x2 graph, as shown in Figures 1 through 4. The chart reflects the speaker's model about the belief state of their conversation partner. For example the chart may reflect a Topic Information Sender's (TIS's) model of what the Topic Information Receiver (TIR's) believes a particular digit of a telephone number is, and how much confidence TIR holds in this belief. A related diagram is shown in Figure 5. The layout here follows Heritage's 2012a Figure 1, which illustrates how an utterance's epistemic stance portrays the speaker's assumption about the Knowledge Gradient between the speakers.

These figures are discussed further below. First we spell out definitions for DialogAct Intents.

The five Intents principally used in data communication dialogs are:

- **Inform** The Inform intent is issued by a sender of topic data who assumes they hold epistemic authority, K+. Their intent is to communicate that information to a partner receiver whom they believe either lacks knowledge of the topic data, or else possesses knowledge of the topic data with low confidence.

- **Request** The Request Intent is issued by a party who either acknowledges that they lack knowledge about topic data in question, or else possesses knowledge but with low confidence. A Request is issued by a Topic Information Receiver in order to request new information or solicit a repeat send, clarification or confirmation by the Topic Information Sender. Also, TIS may issue a Request to TIR to inquire about TIR's beliefs. Of course TIR holds knowledge authority over their own beliefs. Often, a Request uses some form of question syntax and intonation.

- **Check** The Check Intent is defined as a statement of high belief proffered for evaluation by the conversation partner. Most often, this is a statement of repetition or echoing of information content. It is issued by TIR to TIS, once TIR has relatively high confidence that their belief about topic knowledge is correct. A Check DialogAct serves two purposes. First, it informs TIS about what the TIR now believes. Second, it hands over the dialog Turn to TIS. After receiving a Check utterance, TIS is in a position to:

  - increase their confidence that TIR has correct topic data knowledge
  - decide that TIR has incorrect knowledge and issue a correction
  - issue a Confirmation Dialog Act to TIR to solidify their confidence in the topic data
  - advance the dialog and proceed to send new topic data

  Most often, a Check DialogAct uses declarative syntax and intonation.

- **Confirmation** The Confirmation Intent is issued by either TIS to or TIR as an indicator that information has been received. Normally, Confirmation carries an implication that the topic data has been understood and recorded with high confidence, conversational alignment is intact, and the dialog may proceed. It is still up to the hearer to decide what to do with that implication. Very often, a Confirmation DialogAct utterance is a brief positive affirmation.

- **Correction** The Correction Intent is issued when the speaker steps in to signal that the partner has said something incorrect and with high confidence. In data communication, a Correction therefore occurs in response to a Check DialogAct. A Correction DialogAct indicates a high information gradient, i.e. high epistemic stance (K+) on the part of the speaker and low knowledge authority (K-) on the part of the listener. (Other types of conversation not considered here might employ an "Objection" Intent, in which the epistemic gradient tilts less strongly in favor of the speaker.)

These intents are composed with the three levels of DialogAct (Topic Data, DialogManagement, and Roles & Interpersonal), to generate fifteen fundamental categories of Intent plus Dialog Level. For some Intents, the distinctions between these levels can be fine and somewhat blurry. By and large, an Intent is sub-classified as a Topic Data Level Intent if it directly carries topic data, or else directly refers to topic data through anaphor. By and large, an Intent is sub-classified as a Dialog Management Level Intent if it is primarily concerned with the conversation and its protocols. The distinction blurs a bit when we consider on the one hand between data values (Topic Data level), and on the other hand, the places, names for, and indices for those values in the data model (Dialog Management Level). DialogActs that include only data index information, but are not designed to communicate data values as such, are regarded as providing orientation information, which falls midway between TopicData and DialogManagement. See Table I for examples.

For now, with regard to Roles & Interpersonal Intents, we attend only to those that involve epistemic status.

Here are representative examples. A fuller list of DialogAct mappings to utterances is presented in Appendix A.

Table I. Examples of DialogAct Intents

InformTopicData

```
six five zero
the area code is six five zero
that digit is a six
the first digit is a six
```

InformDialogManagement/InformTopicData (borderline)

```
that was the area code
the exchange is next
after that comes the exchange
followed by...
the six is followed by...
```

InformDialogManagement

```
i'll first say the area code
i didn't get that
you are speaking too fast
i am not hearing you
```

InformRoles&Interpersonal

```
i think you are wrong about the area code
i would prefer speaking to a real person
i think you are playing games with me
i programmed to not respond to that kind of language
you are an idiot
```

RequestTopicData

```
what is the area code?
was that a six?
was that six five zero?
please tell me the area code
what is the next number?
```

RequestDialogManagement

```
start with the area code
was that the area code?
did you get that?
please repeat that
would you like me to slow down?
please speak more slowly
```

RequestRoles&Interpersonal

```
are you sure the area code you told me is correct?
you're a robot, right?
why are you speaking like a robot?
can you put a real person on the line?
may I ask why you asking me the same thing over and over?
```

CheckTopicData

six five zero
I've got the area code as six five zero

CheckDialogManagement
that's the area code... (spoken by TIR)
next the exchange... (spoken by TIR)

CheckRoles&Interpersonal
you're a robot

ConfirmationTopicData
that's correct
right
yes

ConfirmationDialogManagement
okay
go ahead
got it

ConfirmationRoles&Interpersonal
a robot, yes
you're doing fine

CorrectionTopicData
no (i.e. followed by InformTopicData: that was six not seven)
not six five zero
sorry that's not correct

CorrectionDialogMangement
sorry no (i.e. followed by InformDialogManagement: that was the
exchange not the area code)
not the area code

CorrectionRoles&Interpersonal
sorry for interrupting
I didn't mean to be rude (–not sure about this -ES)

Notes on the subcategorizations:

Note 1: Confirmation and Inform

A ConfirmationDialogManagement Intent is distinguished from an InformDialogManagement Intent in that a Confirmation Intent does not require further attention on the part of the hearer. The Confirmation indicates that the conversation is on track, and it provides a turn handoff such that the other party may proceed. For example, consider two dialog excerpts:

I.

(1) A: Did you get that? (RequestDialogManagement)
(2) B: Got it. (ConfirmationDialogManagement)

II.

(1) A: `Did you get that?` (RequestDialogManagement)
(2) B: `No I didn't get that.` (InformDialogManagement)

Turn (I.2) does provide information about the state of the dialog, but of such nature that A would be invited to move on. Whereas, turn (II.2) requires that A direct attention to repairing the communication breakdown.

Note 2: Check and Inform

A declarative statement of topic data, such a the utterance, "six five oh", can serve multiple DialogAct intents. When spoken by TIS in order to convey information to TIR, the utterance is labeled as Intent, InformTopicData. However, the same declarative syntax and identical or similar intonation may be used on the part of TIR to echo the information back, as a CheckTopicData Intent. A related phenomenon is observed in the CA literature. [2]

Figures 1 through 4, in conjunction with Figure 5, portray how different Intent/Dialog Layer DialogActs follow from one another to build typical dialog sequences based on belief models on the parts of the participants.

At Turn (1), (Figure 1) the topic information sender TIS simply presents a chunk of information through an InformTopicInfo DialogAct.

At Turn (2) (Figure 2), TIR's response will depend on their confidence in having understood the topic data. A Check DialogAct indicates confidence, but repeats the data back for A to confirm. A Confirmation DialogAct on the part of B indicates high confidence, without checking. Note that silence, or absence of taking a turn, in this context is normally interpreted as Confirmation. A Request DialogAct indicates lower confidence. These options are illustrated also in Figure 5b.

At Turn (3) (Figure 2, Figure 5a again), TIS evaluates both the confidence indicated by TIR's response, and the correctness of the topic information conveyed (in the event TIR issued a Check or Request DialogAct). An indication that TIR has incorrect information will require a repeat Inform DialogAct, but preceded by a Dialog Management utterance that addresses TIR's indicated confidence level. If TIR has reflected incorrect information with high confidence, then a Correction will be called for. In case TIR issued a Confirmation at Turn (2), TIS may elect to solicit an explicit recitation of TIR's belief in the topic data by issuing a RequestTopicInfo DialogAct. Representative dialog sequences for these patterns are shown in Figure 4.

6.1 Semantic Arguments to LogicalForms

The listing in Table I decomposes data communication dialog acts coarsely according to two factors, Intent, and Dialog Level. We take the semantic representation a level deeper by introducing LogicalForm arguments that name different aspects of purpose and types of data communicated. For example, while a Request Intent is a request for some form of communication on the part of the speaker, the intended communication could go in either direction. "Tell me the area code" and "Let me tell you the area code" are both Requests. With regard to type of information to transfer, "Was that a six?" is a Request about a topic data value, while "Was that the area code?" is a Request about data index, that is, the placement of data values in a data model.

[2] "...as first observed by Labov and Fanshel (1977:100), putatively shared knowledge of a recipient's superior epistemic access to some state of affairs permits a speaker to produces, and be heard to produce, a request for information." Note that a Check DialogAct invites the recipient TIS to issue a Confirmation or Correction.

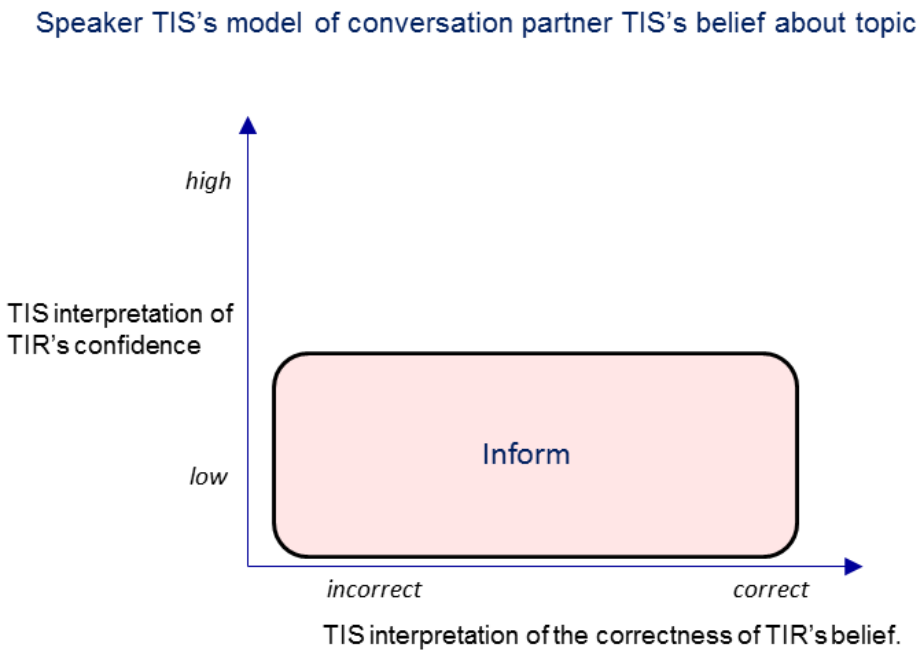

Figure 1: TIS (TopicInformation Sender) model of conversation partner TIR's (Topic Information Receiver's) belief about topic data of interest, depicted before Turn (1); Turn (1) is the time at which Speaker TIS first issues an Inform DialogAct conveying topic information (e.g. one or more digit values). The position of the pink box represents TIS's Inform DialogAct as assuming that before the utterance, TIR has low knowledge/confidence about the data value(s).

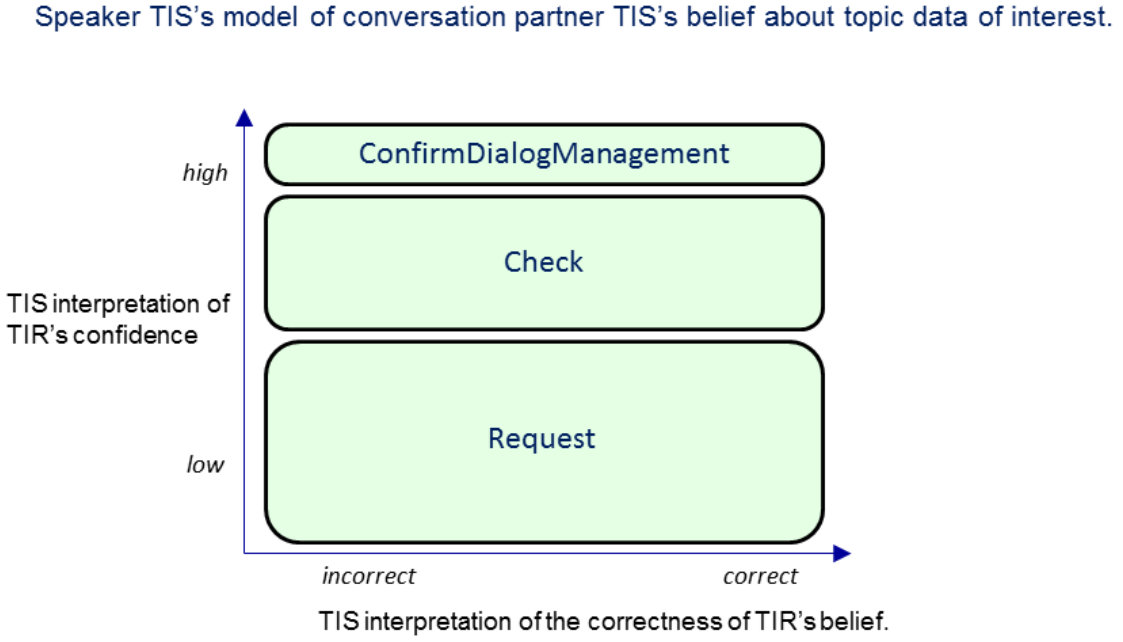

Figure 2: TIS (TopicInformation Sender) model of conversation partner TIR's (Topic Information Receiver's) belief about topic data of interest, depicted after Turn (2); Turn (2) is the time at which Speaker TIR replies to TIS. The green boxes represent different possible DialogAct Intents of TIR's response; their positionings represent different inferences TIS can make about about TIR's knowledge and confidence about the topic data.

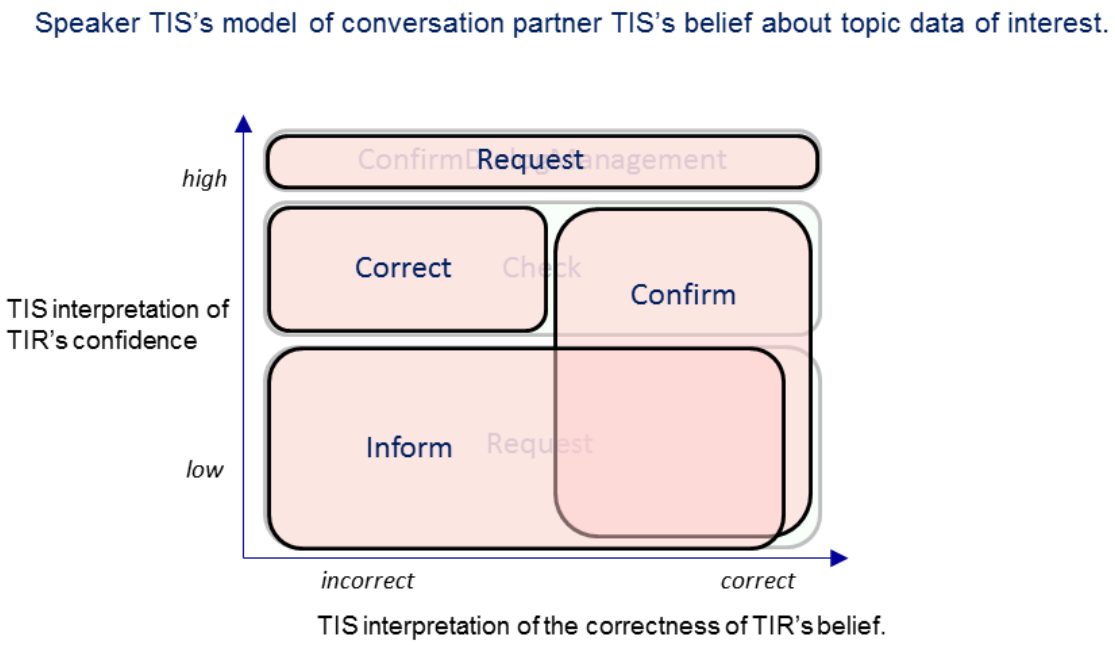

Figure 3: Pink boxes represent different possible Intents of TIS's DialogAct at Turn (3); their positionings reflect different replies that TIS may appropriately issue depending on their model of TIR's knowledge and confidence about the topic data, which was inferred from TIR's response at Turn (2).

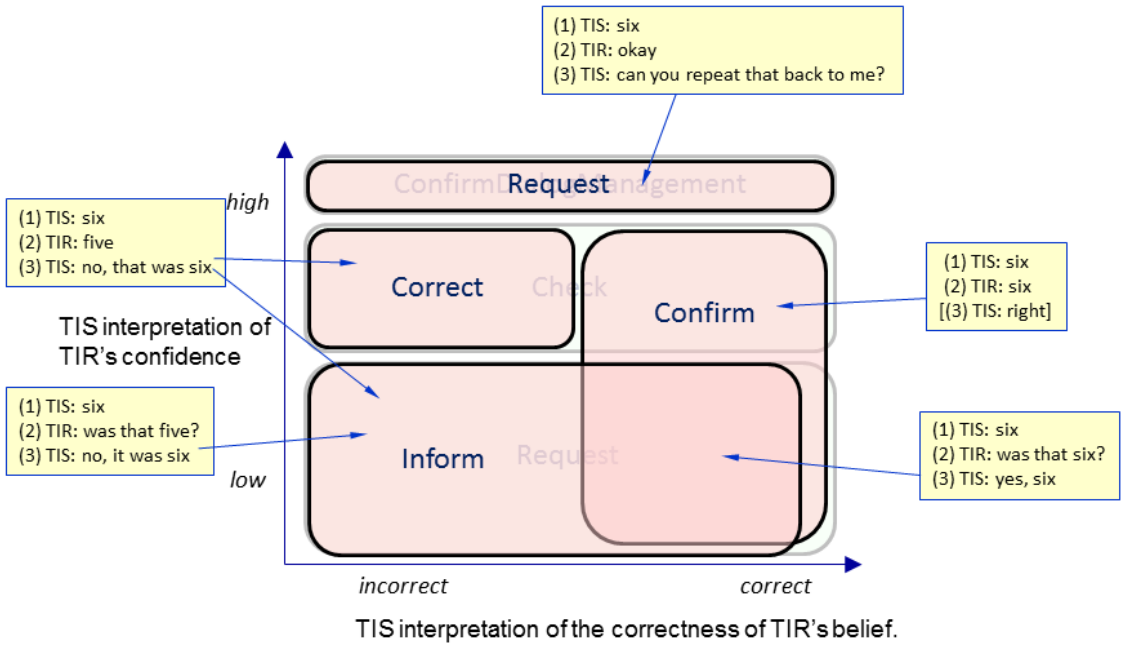

Figure 4: TIS belief model for three-turn sequence, annotated with representative utterances for the DialogActs issued by TIS (Topic Information Sender) and TIR (Topic Information Receiver).

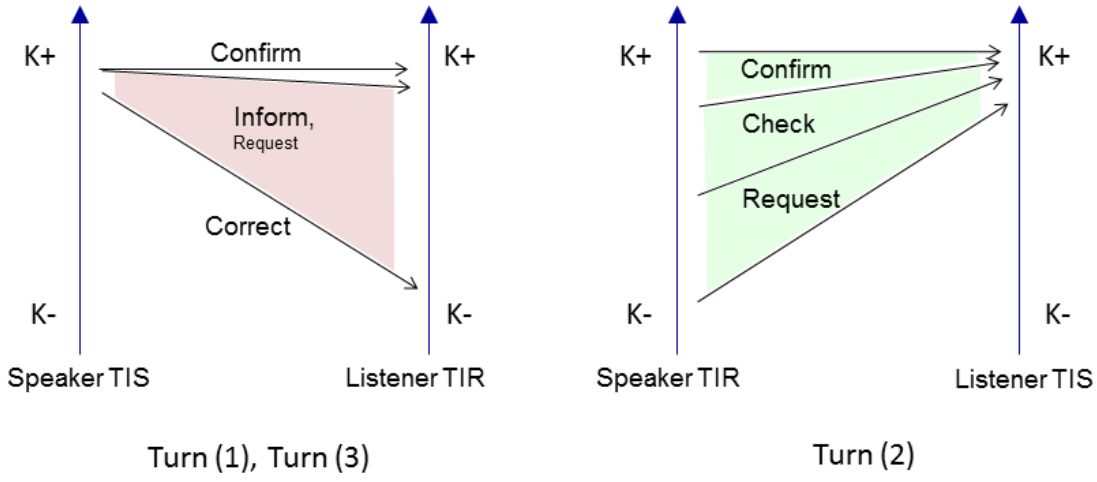

Figure 5: Relative epistemic status, or *epistemic gradient* of TIS (Topic Information Sender) and TIR (Topic Information Receiver) are reflected in the Intents of DialogActs issued by TIS and TIR.

To support deeper semantics like these, the LogicalForm arguments of the basic Intents is developed to include intention predicates and data types specifiers. Nesting is also allowed, to support hierarchical relationships.

Here are some representative examples of elaborating Intent semantics through the use of LogicalForm arguments. A larger listing is presented in Appendix A. The things that can be meaningfully communicated about even structured data such as telephone numbers is endless, so the list of possible LogicalForm arguments to Intents will forever be open-ended.

InformTopicInfo(ItemValue(DigitSequence($1, $2, $3)))
`six five zero`

InformTopicInfo(assertion, FieldName($1), ItemValue(DigitSequence($2, $3, $4)))
`the area code is six five zero`

RequestTopicInfo(request-confirmation, FieldName($1), ItemValue(DigitSequence($2, $3, $4)))
`is the area code six five zero`

InformDialogManagement(repeat-intention)
`I'll repeat that.`

RequestDialogManagement(misalignment-request-repeat, FieldName($1))
`repeat the area code`

FieldName refers to a segment of the data model such as telephone number, country code, area code, exchange, extension, and line number.

DigitSequence a logical form that can be filled in with digit names like, `one`, `two`, `three`, etc.

7 Prototypical DialogAct Sequences

Equipped these tools, we may examine prototypical utterance sequences of the data communication task. The Cognitive Belief Model provides grounds for conversation participants to work toward a common goal of TIR (the Topic Information Receiver) obtaining the information initially possessed by TIS (the Topic Information Sender). The vocabulary of Intents, which organize DialogActs, provides semantic instruments for information transfer and dialog maintenance via linguistic productions. The semantics encompasses multiple dimensions, including topic data itself, parties' beliefs about the topic data and confidence in these beliefs, knowledge authority (epistemic status), the parameters of dialog management (e.g. chunk size, handshaking), and the instantaneous state of the dialog (e.g. turn).

In Table II we list 32 prototypical utterance sequence patterns. In each case, the trajectory can be traced in terms of the speaker's model of their own and their partner's belief about topic data and epistemic status. In the commonplace sequences listed below, the TIS/TIR relationship is assumed to be established and clear at the outset, there are no subtleties or complexities about discovering which party knows what, and there are no disputes about the topic data facts. Also, these sequences reflect purely cooperative interactions. The same framework, however, readily accommodates these more challenging types of interaction. Some are explored in Section 10.

The common utterance sequence patterns are prone to grow in number due to branching options at each turn. Some branching options are data-based, such as the requirement to issue a correction when incorrect beliefs on the part of the conversation partner are detected.

Other options are governed by dialog management parameters, such as the handshaking parameter which prescribes the amount of turn back-and-forth takes place through Check and Confirmation DialogActs.

Table II employs two alternative representations for sequence patterns. First, each pattern has many or most of its branches spelled out in separate sequences. This is followed by a Factored Representation, in which sets of possible DialogActs are presented using brackets [ ], which each option enclosed by parentheses ( ).

Empty parentheses ($\rightarrow$) or ($\leftarrow$) indicates no DialogAct utterance on the part of the party whose turn it initially is; this is normally eventually interpreted as a Confirmation on the part of the other party, who then steps in to take the turn.

(sequence close) indicates that the turn-owning party offers no more DialogActs concerning the topic data item of concern in the sequence. Typically, this occurs when TIS decides to move on to the next topic data item, e.g. the next digit or segment in a telephone number.

Turn Ownership, Control, and Net Handshaking Alignment are discussed below the table of sequence patterns.

Table II. Protypical Sequence Patterns

TIS DialogAct	Utterance Direction	TIR DialogAct	Representative Utterance	Turn Ownership	Control	Net Handshake Alignment	
Inform/Check/X pattern							
S1:							
InformTD	→		six	→	←	+2	2
	←	CheckTD	six	↔	←	+2	4
(sequence close)							
S2:							
InformTD	→		six	→	←	+2	2
	←	CheckTD	six	←	←	+2	4
ConfirmationTD	→		yes	→	←	+1	5
S3: InformTD	→		six	→	←	+2	2
	←	CheckTD	six	←	←	+2	4
ConfirmationTD	→		yes	←	←	+1	5
InformTD	→		six	→	←	+2	7
S4:							
InformTD	→		six	→	←	+2	2
	←	CheckTD	five	←	←	+2	4
CorrectionTD	→		no	←	←	XX	0
InformTD	→		six	→	←	+2	2
Inform/Check/X pattern, Factored Representation:							
S5:							
InformTD	→		six	→	←	+2	2
	←	CheckTD	six (correct)	←	←	+2	4
[							
(sequence close)				↔	←		4
or							
(ConfirmationTD	→		yes)	→	←	+1	5
or							
(ConfirmationTD	→		yes	←	←	+1	5
InformTD	→		six)	→	←	+2	7
]							
S6:							
InformTD	→		six	→	←	+2	2

	←	CheckTD	five(incorrect)	←	←	+2	4
CorrectionTD	→		no	←	←	XX	0
InformTD	→		six	→	←	+2	2

Inform/Request/X pattern

S7:

InformTD	→		six	→	←	+2	2
	←	RequestTD	was that six?	←	→	+1	3
ConfirmationTD	→		yes	↔	→	+1	4

S8:

InformTD	→		six	→	←	+2	2
	←	RequestTD	was that six?	←	→	+1	3
ConfirmationTD	→		yes	↔	→	+1	4
	←	ConfirmationDM	okay got it	←	←	+1	5

S9:

InformTD	→		six	→	←	+2	2
	←	RequestTD	was that six?	←	→	+1	3
ConfirmationTD	→		yes	←	→	+1	4
InformTD	→		six	↔	→	+2	6

S10:

InformTD	→		six	→	←	+2	2
	←	RequestTD	was that six?	←	→	+1	3
ConfirmationTD	→		yes	←	→	+1	4
InformTD	→		six	↔	→	+2	6
	←	ConfirmationDM	okay got it	←	←	+1	7

S11:

InformTD	→		six	→	←	+2	2
	←	RequestTD	was that five?	←	→	+1	3
CorrectionTD	→		no	→	→	XX	0
InformTD	→		six	↔	→	+2	2

Inform/Request/X pattern, Factored Representation:

S12:

InformTD	→		six	→	←	+2
	←	RequestTD	was that six?(correct)	←	→	+1
[						
(ConfirmationTD	→		yes)	→	→	+1

or							
(ConfirmationTD	→		yes	←	→	+1	
InformTD	→		six)	→	→	+2	
]							
	←	ConfirmationDM	okay got it)	←	←	+1	

S13:							
InformTD	→		six	→	←	+2	
	←	RequestTD	was that five?(incorrect)	←	→	+1	
CorrectionTD	→		no	←	→	XX	
InformTD	→		six	↔	→	+2	

Inform/Confirmation pattern

S14:							
InformTD	→		six	→	←	+2	2
(sequence close)				↔	←	2	

(completion of sequence: no response from TIR, TIS moves on to next topic data item)

S15:							
InformTD	→		six	→	←	+2	2
	←	ConfirmationDM	okay	←	←	+1	3
(sequence close)				↔	←		3

(completion of sequence: TIS moves on to next topic data item)

Inform-Request/Inform/X pattern

S16:							
InformTD	→		six	→	←	+2	2
RequestTD	→		what do you have for that digit?	→	←		2
	←	InformTD	six	←	←	+2	4
(sequence close)				↔	←		4

S17:							
InformTD	→		six	→	←	+2	2
	←	ConfirmationDM	okay	←	←	+1	3
RequestTD	→		what do you have for that digit?	→	←		3
	←	InformTD	six	←	←	+2	5
(sequence close)				↔	←		5

S18:

InformTD	→		six	→	←	+2	2
RequestTD	→		what do you have for that digit?	→	←		2
	←	InformTD	six	←	←	+2	4
ConfirmationTD	→		right	←	←	+1	5

S19:

InformTD	→		six	→	←	+2	2
	←	ConfirmationDM	okay	←	←	+1	3
RequestTD	→		what do you have for that digit?	→	←		3
	←	InformTD	six	←	←	+2	5
ConfirmationTD	→		right	←	←	+1	6

S20:

InformTD	→		six	→	←	+2	2
RequestTD	→		what do you have for that digit?	→	←		2
	←	InformTD	six	←	←	+2	4
ConfirmationTD	→		right	←	←	+1	5
InformTD	→		six	←	←	+2	7

S21:

InformTD	→		six	→	←	+2	2
	←	ConfirmationDM	okay	←	←	+1	3
RequestTD	→		what do you have for that digit?	→	←		3
	←	InformTD	six	←	←	+2	5
ConfirmationTD	→		right	←	←	+1	6
InformTD	→		six	←	←	+2	8

S22:

InformTD	→		six	→	←	+2	2
RequestTD	→		what do you have for that digit?	→	←		2
	←	InformTD	five	←	←	+2	4
CorrectionTD	→		no	←	←	XX	0
InformTD	→		that's six	↔	←	+2	2

S23:

InformTD	→		six	→	←	+2	2
	←	ConfirmationDM	okay	←	←	+1	3
RequestTD	→		what do you have for that digit?	→	←		3

	←	InformTD	five	←	←	+2	5
CorrectionTD	→		no	←	←	XX	0
InformTD	→		that's six	↔	←	+2	2

Inform-Request/Inform/X pattern, Factored Representation:

S24:

InformTD	→		six	→	←	+2
[						
(	←		)	←		
or						
(	←	ConfirmationDM	okay)	←	←	+1
]						
RequestTD	→		what do you have for that digit?	→	←	
	←	InformTD	six(correct)	←	←	+2
[						
(	→		)			
or						
(ConfirmationTD	→		right)	←	←	+1
or						
(ConfirmationTD	→		right	←	←	+1
InformTD	→		six)	←	←	+2
]						

S25:

InformTD	→		six	→	←	+2
[						
(	←		)	←		
or						
(	←	ConfirmationDM	okay)	←	←	+1
]						
RequestTD	→		what do you have for that digit?	→	←	
	←	InformTD	five (incorrect)	←	←	+2
CorrectionTD	→		no	←	←	XX
InformTD	→		that's six	↔	←	+2

Inform-Check/Inform/X pattern

S26:

InformTD	→		six	→	←	+2 2
CheckTD	→		you got six for that, right?	→	←	+2 4
	←	ConfirmationTD	yes	←	←	+1 5

S27:

InformTD	→		six	→	←	+2 2
	←	ConfirmationDM	okay	←	←	+1 3
CheckTD	→		you got six for that, right?	→	←	+2 5
	←	ConfirmationTD	yes	←	←	+1 6

S28:

InformTD	→		six	→	←	+2 2
CheckTD	→		you got six for that, right?	→	←	+2 4
	←	ConfirmationTD	yes	→	←	+1 5
	←	InformTD	six	←	←	+2 7

S29:

InformTD	→		six	→	←	+2 2
	←	ConfirmationDM	okay	←	←	+1 3
CheckTD	→		you got six for that, right?	→	←	+2 5
	←	ConfirmationTD	yes	→	←	+1 6
	←	InformTD	six	←	←	+2 8

S30:

InformTD	→		six	→	←	+2 2
CheckTD	→		you got six for that, right?	→	←	+2 4
	←	ConfirmationTD	yes	→	←	+1 5
	←	InformTD	five	←	←	+2 7
CorrectionTD	→		no	←	←	XX 0
InformTD	→		it's six	↔	←	+2 2

S31:

InformTD	→		six	→	←	+2 2
CheckTD	→		you got six for that, right?	→	←	+2 4
	←	CorrectionTD	no	→	←	XX 0
	←	InformTD	I got five	←	←	+2 2
CorrectionTD	→		well no	←	←	XX 0
InformTD	→		it's six	↔	←	+2 2

S32:

InformTD	→		six	→	←	+2 2
	←	ConfirmationDM	okay	←	←	+1 3
CheckTD	→		you got six for that, right?	→	←	+2 5
	←	ConfirmationTD	yes	→	←	+1 6

	←	InformTD	five	←	←	+2 8
CorrectionTD	→		no	←	←	XX 0
InformTD	→		it's six	↔	←	+2 2

Inform-Check/Inform/X pattern, Factored representation

S33:

InformTD	→		six	→	←	+2
[						
(	←		)	←		
or						
(	←	ConfirmationDM	okay)	←	←	+1
]						
CheckTD	→		you got six for that, right?	→	←	+1
[						
(	←	ConfirmationTD	yes)	←	←	+1
or						
(	←	ConfirmationTD	yes	→	←	+1
	←	InformTD	six(correct))	←	←	+2
or						
(	←	InformTD	six(correct))	←	←	+2
]						

S34:

InformTD	→		six	→	←	+2
[						
(	←		)	←		
or						
(	←	ConfirmationDM	okay)	←	←	+1
]						
CheckTD	→		you got six for that, right?	→	←	+1
[						
(	←	CorrectionTD	no)	←	←	XX
or						
(	←	CorrectionTD	no	→	←	XX
	←	InformTD	five (incorrect))	←	←	+2
or						
(	←	InformTD	five (incorrect))	←	←	+2
]						
[						
(CorrectionTD	→		no	←	←	XX
InformTD	→		it's six)	↔	←	+2
or						
(CorrectionTD	→		well no	←	←	XX
InformTD	→		it's six)	↔	←	+2
]						

X/Request/Inform

S35:
...

	←	RequestTD	what is the area code?	←	→	
InformTD	→		six five zero	→	→	+2 2
	←	ConfirmationTD	six five zero	↔	→	+2 4

S36:
...

	←	RequestTD	what is the area code?	←	→	
InformTD	→		six five zero	→	→	+2 2
	←	ConfirmationDM	got it	←	←	+1 3

S37:
...

	←	RequestDM	was that the area code?	←	→	+1 1
InformDM	→		yes	→	→	+1 2

S38:
...

	←	RequestDM	was that the area code?	←	→	+1 1
InformDM	→		no	←	→	XX 0
InformDM	→		that was the exchange	↔	→	+1 1

S39:
...

	←	CheckDM	that was the area code...	←	→	+1 1
ConfirmationDM	→		right	→	→	+1 2

S40:
...

	←	CheckDM	that was the area code...	←	→	+2 2
CorrectionDM	→		no	→	→	XX 0
InformDM	→		that was the exchange	↔	→	+2 2

8 Analysis of Prototypical Patterns

Let us consider the columns, Turn Ownership, Control, and Net Handshake Alignment. Speaker, Turn Ownership, and Control are indicated by arrows. For the first three columns, the arrow points from utterance speaker toward utterance receiver, and the DialogAct Intent

form appears in the speaker's column, TIS or TIR. For Turn Ownership and Control, a left arrow indicates that TIS retains turn ownership or control, while a right arrow indicates TIR.

Turn Ownership refers to which party has priority in speaking next. Control refers to the party who will obtain Turn Ownership when the current dialog segment is complete, where *dialog segment* refers to a self-contained group of utterances that accomplish a subgoal or subtask within the larger task, such as establishing the correct communication of one item of topic data. Turn Ownership can shift back and forth on an utterance-by-utterance basis in order to execute a dialog segment, whereas Control operates at a somewhat longer time scale.

In normal conversation, when turn ownership is clearly established by virtue of the dialog following established protocol patterns, then the turn non-owning party refrains from speaking for sufficient time for the turn owner to speak next. We observe that the turn owner may elect not to speak. For example, in S14 and S15, TIS offers Topic Data in an InformTD utterance. Turn shifts to TIR. In S15 they reply with a ConfirmationTopicData utterance, while in S15 they remain silent and eventually implicit confirmation is assumed by TIS.

Turn ownership may be retained by a speaker to produce successive DialogActs, simply by appending them with no pause or minimal pause. This happens for example in S3 and S4, where a ConfirmationTopicData or CorrectionTopicData DialogAct is immediately followed up by an Inform DialogAct containing explicit topic data proper.

Certain DialogActs typically serve dual purposes of both conveying information and indicating turn ownership. The most prominent example is ConfirmDialogManagement, which usually both indicates agreement with the topic data conveyed, and hands turn ownership to the other party (this is variously referred to as a "Prompt" or a "Continuer" in the CA literature).

Under most protocol patterns, TIS assumes responsibility for communicating topic data by exercising Control in the dialog. However, Control can be claimed away by a party asking a question. The claim may be ceded when the issue that prompted the question is resolved, or it may be retained as the questioning party moves to ask a new question.

Violations of standard protocol patterns provoke common-sense repair efforts. For example, a variant on S4 is shown in S41. Here, TIS issues a CorrectionTopicData DialogAct, which retains Turn Ownership for TIS. But then they fail to follow up with the expected InformTopicData DialogAct. Eventually TIR steps in to ask about it. The "well," which might naturally preceed "what is it?", could be interpreted as a DialogAct of type RequestDialogManagement, indicating that protocol alignment is perceived to have broken down. But "well" also contains shades of Roles&Interpersonal, if TIR indicates impatience with TIS's non-standard behavior.

TIS DialogAct	Utterance Direction	TIR DialogAct	Representative Utterance	Turn Ownership	Control	Net Handshake Alignment
S41:						
InformTD	→		six	→	←	+2 2
	←	CheckTD	five	←	←	+2 4
CorrectionTD	→		no	←	←	XX 0
(pause)						
	←	RequestDM	well			
	←	RequestTD	what is it?			

The property, Net Handshake Alignment is concerned with the degree of dialog alignment maintenance that takes place in the course of a dialog segment. Alignment maintenance occurs when parties utter Check, Confirmation, Request, and Inform utterances that contribute redundancy and double-checking of agreement on topic data and dialog management dimensions. We assign an incremental score that each utterance contributes to Handshake Alignment, and a cumulative score for a dialog segment. The cumulative score is an indication of how confident the parties can be that they are in agreement about the topic data in question. The scoring is as follows:

+2 : For explicit communication of topic data through a DialogAct of type, InformTopicData, CheckTopicData and some instances of RequestTopicData

+1 : For confirmation of receipt of information (without echoing the content) through a DialogAct of type ConfirmationDialogManagement, InformDialogManagement, RequestDialogManagement, or CheckDialogManagement.

XX : Reset cumulative Net Handshake Alignment to 0 when a discrepancy has been discovered in belief about topic data. Reset corresponds to a DialogAct of type, CorrectionTopicData or CorrectionDialogManagement, or InformDialogManagement. Note that simply communicating different topic data values, as in S4, does not trigger a Net Handshake Alignment Reset. Because of communication channel noise, topic data values are not known to have been heard as intended by the speaker. Therefore Reset occurs only when the discrepancy is called out explicitly in the dialog.

One purpose for conversational actors to tally Net Handshake Alignment during the course of dialog is to provide guidance about whether their dialog contains the degree of checks, confirmation, and other handshaking protocol elements that their current prescriptive setting of their handshake parameters indicate. Many of the protocol patterns listed in Table II are slight variants on one another by virtue of tacking on or inserting Check and Confirmation DialogActs that boost Net Handshake Alignment.

9 Implementation in a Program

The Tell Phone Number experimental dialog application attempts to implement this approach to achieve naturalistic human-computer conveyance of a telephone number. The Cognitive Belief Model is implemented in terms the architecture shown in Figure 6. The computer agent holds two Dialog Models, one representing its knowledge about data values and parameters of the conversation, the other representing what the agent believes the conversation partner holds for these values.

The data values themselves are contained in a Data Model, which includes both data values themselves (e.g. telephone numbers), but also indices into the data in terms of fields or segment names. For example the Area Code field is data indices 0-2 (zero-based) within the entire telephone number.

All communication takes place in terms of Dialog Acts. Because utterances take place in physical space, the record of turn utterances and their corresponding DialogActs stands outside the agent's belief model.[3] Additionally, the program maintains a list of turns that represent pending questions issued by the computer agent or else the conversation partner. This is similar to what Otto does.

[3]It could be argued that the LogicalForm semantic representation is an *interpretation* of physical utterances, and therefore belongs inside one or both of the agent's Dialog Models.

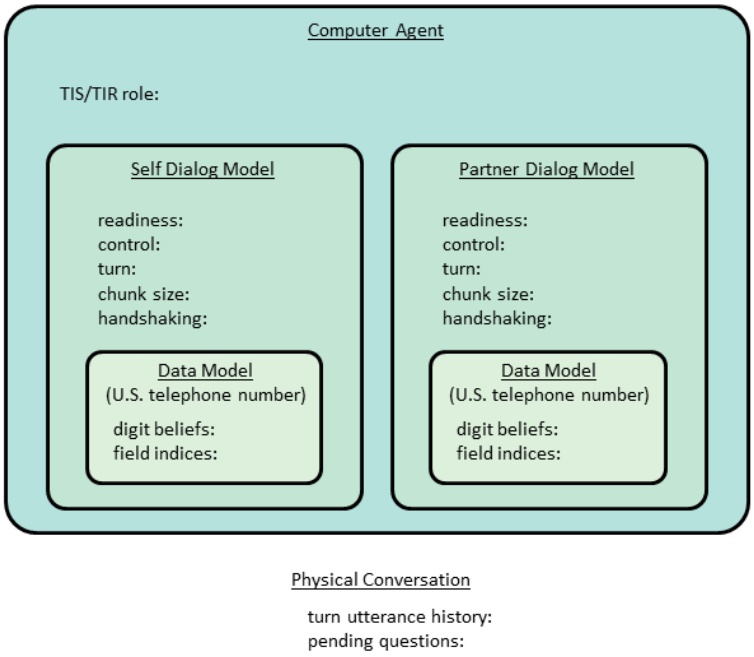

Figure 6: Organization of model parameters for a naturalistic computer dialog agent. The agent maintains a belief model of its own knowledge and conversation parameters, plus a probabilistic estimation of domain knowledge and parameters held by the conversation partner. TIR/TIS means Topic Information Sender/Receiver.

The program operates through hand-crafted rules. Input utterances are dispatched to subroutines based on their top level Intents. The computer agent is loosely goal-driven, in the sense that when the computer agent is has control, the rules are written for it to compare its knowledge with its model of the partner's data belief and confidence, and reduce discrepancy. Appendix A lists the most significant Intent Logical Forms used by the program so far.

An essential question about this approach is whether this approach will scale to more complex data models beyond telephone numbers, and more complex conversational interactions. I propose the following hypothesis:

Hypothesis: The variety and complexity of Topic Data type Logical Form Intents is proportional to that of data models so is essentially unbounded, but the variety and complexity of Dialog Management type Logical Form Intents is limited.

Motivation for this hypothesis is that the role of Dialog Management as such in structured data communication is circumscribed by a limited set of dialog management parameters such as described above and shown in Figure 6. If this is correct, then with sufficient effort, we should be able to capture the back-and-forth conversational aspects of structured data communication more-or-less completely. To scale to new domains would entail only extending the set of Topic Data Logical Forms to match the new domains' data models.

10 Interesting Sequences

11 Appendix A: Listing of Principal DialogAct Logical Forms

These are the principle DialogAct Logical Forms used so far for telephone number communication. The numbers following dollar signs (e.g. $30) are used to indicate logical arguments that are passed from the utterance parser to the dialog management engine.

format:
    LogicalForm
        example utterance

InformTopicData
    InformTopicData(ItemValue(DigitSequence($1, $2, $3)))
        six five zero
    InformTopicInfo(SayIs(FieldName($30))
        the area code is
    InformTopicInfo(Grammatical($100), FieldName($30))
        that is the area code
    InformTopicInfo(nothing-relative-to, Indexical($140), FieldName($30))
        there is nothing after the area code
    InformTopicInfo(nothing-relative-to-indicative, Indexical($140))
        there is nothing after that
    InformTopicInfo(SayIs(ItemTypeChar($25))
        the digit is
    InformTopicInfo(SayIs(ItemTypeChar($25), Indexical($140)))
        the first digit is
    InformTopicInfo(meaning-of, FieldName($30))
        the area code is the first three digits of the telephone number
    InformTopicInfo(GrammaticalBeIndicative($100), FieldName($30))
        that is the area code
    InformTopicInfo(all-done)
        we're all done

InformDialogManagement
    InformDialogManagement(self-readiness)
        i am ready
    InformDialogManagement(self-not-readiness)
        please wait a moment
    InformDialogManagement(declare-waiting-for-partner)
        i am waiting for you
    InformDialogManagement(repeat-intention)
        i'll repeat that
    InformDialogManagement(misalignment-confusion)
        i am confused
    InformDialogManagement(misalignment-self-hearing-or-understanding)
        sorry, i did not understand that
    InformDialogManagement(Inform(partner, self), Tense(past))
        i heard you say
    InformDialogManagement(Knowledge(possess, PersonRef($132)), Tense($101), FieldName($30))
        i have the area code already
    InformDialogManagement(Knowledge(possess, PersonRef($132)), Tense($101), GrammaticalIndicative($100))
        i have that already

InformRoles&Interpersonal

InformRoleInterpersonal(user-belief-unsure)
`i do not know`
InformRoleInterpersonal(youre-welcome)
`you're welcome`
InformRoleInterpersonal(believe, PersonRef($132))
`i believe`
InformRoleInterpersonal(not-believe, PersonRef($132))
`i do not believe`

RequestTopicData
RequestTopicInfo(SendReceive($50), FieldName($30))
`i would like to receive the telephone number`
RequestTopicInfo(SendReceive($50), FieldName($30))
`what is the area code`
RequestTopicInfo(SendReceive($50), Indexical($140), FieldName($30))
`tell me the entire the area code`
RequestTopicInfo(SendReceive($50), Indexical($140), ItemTypeChar($25))
`tell me the next number`
RequestTopicInfo(SendReceive($50), ItemTypeChar($25), Indexical($140), GrammaticalIndicative($100), FieldName($30)
`what is the third digit of the exchange`
RequestTopicInfo(request-affirmation)
`is that right`
RequestTopicInfo(meaning-of, FieldName($30))
`what does line number even mean`
RequestTopicInfo(request-confirmation, ItemValue(DigitSequence($1, $2, $3)), Grammar($100))
`was that six five zero`
RequestTopicInfo(request-confirmation, ItemValue(Digit($1, $2, $3)), FieldName($30), Tense($100)
`is six five zero the area code`

RequestDialogManagement
RequestDialogManagement(partner-desire, send-or-receive, telephone-number)
`Would you like to send or receive a phone number?`
RequestDialogManagement(proceed-with-next)
`please go on`
RequestDialogManagement(self-not-readiness)
`please wait`
RequestDialogManagement(are-you-waiting)
`are you waiting for me`
RequestDialogManagement(misalignment-start-again)
`start over again`
RequestDialogManagement(misalignment-request-repeat)
`please repeat that`
RequestDialogManagement(misalignment-request-repeat, FieldName($30))
`say the area code again`
RequestDialogManagement(clarification-utterance, Grammatical(past), FieldName($30))
`was that the area code`
RequestDialogManagement(proceed-to-completion, FieldName($30))
`tell me the rest of the area code`
RequestDialogManagement(clarification-utterance, ItemValue(DigitSequence($1, $2, $3)), Speaker($100))
`did you say six five zero`

RequestRoles&Interpersonal
RequestRoleInterpersonal(other-belief)
`do you believe`

CheckTopicInfo

CheckTopicInfo(ItemValue(DigitSequence($1, $2, $3)))

six five zero

ConfirmationDialogManagement

ConfirmDialogManagement(affirmation-okay)

okay

ConfirmDialogManagement(affirmation-yes)

yes

ConfirmDialogManagement(good)

good

CorrectionTopicData

CorrectionTopicData(negation)

no

CorrectionTopicInfo(negation-polite)

sorry no

CorrectionTopicInfo(partner-correction)

no it's

CorrectionTopicInfo(FieldName($30))

not the area code

CorrectionTopicInfo(ItemValue(DigitSequence($1, $2, $3)))

not six five zero

CorrectionDialogMangement

CorrectionDialogManagement(negation)

no

Notes on LogicalForm Arguments

The arguments in the LogicalForms representing DialogActs fall broadly into two classes, *content* arguments and *grammatical* arguments.

Content arguments are purely concerned with topic data, for example digit values, the names of data fields, and the purpose of the utterance (e.g. clarification, request-repeat).

Grammatical arguments carry generic semantic information that is often associated with the lexical and syntactic forms of utterances, such as indefinite vs. definite (the/a), singular vs. plural cardinality (the/those), past/present tense (is/was). For example, the argument, GrammaticalIndicative($100), is itself a LogicalForm having the predicate "GrammaticalIndicative", which itself takes an argument whose values include:

definite	the
definite-near	this
definite-far	that
indefinite	a

These and other arguments enable encoding of the following variations of the DialogAct, InformDialogManagement(Knowledge(possess, PersonRef($132)), Tense($101), GrammaticalIndicative($100)):

i have it already
you have it already
you have that already
i had that already

Elegant Natural Language Processing systems such as Lexical Functional Grammar have been developed to disentagle various aspects and subtlties of meaning that are encoded in lexical and syntactic structures. The LocalForm arguments of this work are intended to be a lightweight way of achieving some of this.