IJCAI 07

5th **IJCAI Workshop on**

Knowledge and Reasoning in Practical Dialogue Systems

Workshop Proceedings

Introduction

This is the fifth workshop on Knowledge and Reasoning in Practical Dialogue Systems. The first workshop was organised at IJCAI-99 in Stockholm,[1] the second workshop took place at IJCAI-2001 in Seattle[2], the third workshop was held at IJCAI-2003 in Acapulco[3] and the fourth at IJCAI-2005 in Edinburgh[4].

The current workshop includes presentation and discussion of research on grammar and parsing for dialogue, information-retrieval and summarization techniques for dialogue generation, dialogue modelling, adaptive dialogue systems, dialogue for robots, and evaluation of dialogue systems.

These workshop notes contain 10 papers that address these issues from various view-points. The papers provide stimulating ideas and we believe that they function as a fruitful basis for discussions during the workshop and will stimulate further research in the future.

The program committee consisted of the colleagues listed below. Without the time spent reviewing the submissions and the thoughtful comments provided by these colleagues, the decision process would have been much more difficult. We would like to express our warmest thanks to them all.

We would also like to thank Andrei Voronkov for help in use of the Easychair Conference management system (www.easychair.org), and Carles Sierra, the IJCAI workshop chair.

[1] Selected contributions have been published in a special issue of ETAI, the Electronic Transaction of Artificial Intelligence http://www.ida.liu.se/ext/etai/

[2] http://www.ida.liu.se/~nlplab/ijcai-ws-01/

[3] http://www.ida.liu.se/~nlplab/ijcai-ws-03/

[4] http://www.csse.monash.edu.au/~ingrid/IJCAI05dialogueCFP.html

Organizers:

David Traum (Chair), ICT, University of Southern California (USA)
Jan Alexandersson (Co-Chair), DFKI GmbH (Germany)
Arne Jönsson (Co-Chair), Linköping University (Sweden)
Ingrid Zukerman (Co-Chair), Monash University (Australia)

Program Committee:

Jens Allwood, Department of Linguistics, Göteborg University (Sweden)
Elisabeth Andre, University of Augsburg (Germany)
Johan Bos, Università di Roma ”La Sapienza” (Italy)
Sandra Carberry, University of Delaware (USA)
Justine Cassell, Northwestern University (USA)
Joakim Gustafson, TeliaSonera (Sweden)
Yasuhiro Katagiri, Future University - Hakodate (Japan)
Susann Luperfoy, Tilted Axis Corp (USA)
Yosuke Matsusaka, AIST (Japan)
Anton Nijholt, University of Twente (Netherlands)
Tim Paek, Microsoft Research (USA)
Candace Sidner, Mitsubishi Electric Research Labs (USA)
Ipke Wachsmuth, University of Bielefeldt (Germany)

Table of Contents

Conference Program

Monday, January 8, 2007

9:30–9:45 Opening Remarks

Session 1: Dialogue Utterance Interpretation

9:45–10:15 *Robust Dependency Parser for Natural Language Dialog Systems in Tamil*
C J Srinivasan, N Udhayakumar, R Loganathan and C Santhosh Kumar

10:15–10:45 *Considering Multiple Options when Interpreting Spoken Utterances*
Sarah George, Ingrid Zukerman, Michael Niemann and Yuval Marom

10:45–11:00 Discussion

11:00-11:30 Coffee/Tea

Session 2: Retrieval and Summarization approaches to Dialogue

11:30–12:00 *Dialogue Generation for Robotic Portraits*
Andrew M Olney

12:00–12:30 *First Steps towards Dialogue Modelling from an Un-annotated Human-Human Corpus*
Sudeep Gandhe and David Traum

12:30–13:00 *Evaluation of a Large-scale Email Response System*
Yuval Marom and Ingrid Zukerman

13:00–14:00 Lunch

14:00–14:30 Discussion: Corpus-based methods and Evaluation

Monday, January 8, 2007 (continued)

Session 3: Affective Dialogue

15:10–15:30 Discussion

15:30-16:00 Coffee/Tea

Session 4: Knowledge and Reasoning in Dialogue

17:10–17:30 Discussion

17:30–18:00 Final discussion

Robust Dependency Parser for Natural Language Dialog Systems in Tamil

Srinivasan CJ*, Udhaykumar N+, Loganathan R*, Santhosh Kumar C*

*Centre for Excellence in Computational Engineering,
Amrita Vishwa Vidyapeetham,
Coimbatore 641105.
{cj_srinivasan,r_logu,cs_kumar}@ettimadai.amrita.edu

+Language Technologies Institute,
Carnegie Mellon University,
Pittsburgh, USA.
udhay@cmu.edu

Abstract

Tamil, being an inflectional language with restricted free word order syntax, presents inherent difficulties for parsers using Finite State Automata (FSA) or Context Free Grammars (CFG) and their variants. We propose a robust dependency parser to parse natural language utterances in Tamil. The parser uses a PC-Kimmo based morphological analyzer that examines the Tamil words to identify root forms and their inflections. The analyzer computes feature structures that are used to identify dependency relations among the words, which in turn are translated into semantic frames. The parser is integrated into a mixed-initiative natural language dialog system built on MIT GALAXY architecture. It understands users' response and interacts with a dialog manager to provide information about availability of trains and ticket fares. The paper discusses the challenges in parsing Tamil utterances and our efforts in building a robust parser for a Tamil spoken dialog system which provides automatic train enquiry services. The dialog system had a task success rate of 76% (38 out of 58 users) in our evaluation study.

1 Introduction

Interactive spoken dialog systems allow users to access online information using natural language. They are particularly attractive in developing regions due to their cheap infrastructure and ease of accessibility among users with low literacy levels. From a cultural point of view, spoken dialog systems are highly appropriate in countries like India, where a strong oral tradition exists in its illiterate and semi-literate population [Parikh et al., 2006]. Multi-modal dialog systems have been recently shown to improve the existing protocols in India to disseminate information concerning essential domains like healthcare, agriculture, primary education, etc [Plauché et al., 2006].

Tamil is a classical language spoken by around 74 million people worldwide. It is one of the official languages in Srilanka, Singapore and Tamil Nadu in India. Tamil, like many other Indian languages differs significantly from English in both morphology and syntax. This presents unique challenges to parse and understand Tamil utterances. Traditional FSA/CFG variants and template based extraction techniques are not suitable for the task. A parser for Tamil should analyze the inflected word forms to determine the semantic relations between them. Moreover it needs to parse the output of speech recognizer in a dialog system framework, rather than well-formatted written text. So the parser needs to be robust enough to deal with recognition errors and handle discontinuities that are common in spontaneous speech such as elliptic sentences, false starts, filled pauses, corrections, etc. This paper describes various issues in building a robust parser for Tamil for seamless integration into a spoken dialog system used by native people to access train enquiry services over telephone. We have implemented a dependency parser to address these challenges

There have been significant research efforts in building interactive spoken dialog systems in English language. [Zue et al., 2000] explains an interactive weather information system. [Raux et al., 2003] discusses research issues in building a spoken dialog system for elderly and non-native speakers. [Allen et al., 2001] describes the results of 10 year effort in building robust spoken dialogue systems for English. For Indian languages, [Reddy et al., 2006] describes a natural language QA system designed for Telugu using a keyword based approach. The system handles inflected word forms by listing all the possible noun forms as alias in the knowledge base.

Section 2 explains various issues in parsing Tamil utterances. Section 3 introduces dependency parsing and explains how it solves the issues in parsing. Section 4 gives an

Proceedings of the 5th *Workshop on Knowledge and Reasoning in Practical Dialogue Systems*, pages 1–6,
Hyderabad, January 8th, 2007.

overview of the train information system, system capabilities and a sample conversation with the system. Section 5 presents the evaluation done. All Tamil utterances in this paper are transcribed using Anjal (www.suvadi.com) roman transliteration standard.

2 Parsing issues in Tamil

A simple technique to parse and understand users utterance is to use keyword or template matching. In many cases, input utterances are parsed using hand-crafted Context Free Grammars. The decision to use such techniques for parsing is greatly influenced by the natural language for which the system is built. CFGs and template based matching techniques are shown to work well for English language as it has a simple morphology and strict word order. However, Indian languages are vastly different from English in both word formation (i.e. Morphology) and in sentence construction (i.e. Syntax). Tamil has concatenative morphology and word formation is significantly complex compared to English. With regards to syntax, Tamil is a free word order language. The constituents of a clause can be moved to alter the emphasis placed on the role players without affecting its semantics.

1.1 Morphological issues

Tamil belongs to Dravidian family of languages and is an agglutinative language. Words are formed by concatenating morphemes one after another. These morphemes are functionally different and add a variety of features to the word being formed. Tamil uses case suffixes and post-positions instead of prepositions. For example, a prepositional phrase "from Chennai" is written as "cennaiyilirundthu". For every other preposition, an inflected word with the same root is quite possible. Since the contextual information, like case marking, is a part of the word form, it is almost impossible to design templates to capture the information.

Similarly, Tamil verbs get inflected for tense, png (person, number and gender), negativity, passiveness, etc. In addition, verbs can also get inflected for interrogation. These inflections present difficulties in detecting the difference between INFORM and REQUEST acts, using keyword matching techniques. For instance, "irukku" is an INFORM act, and its inflected form "irukkaa" is a REQUEST act. These acts cannot be differentiated using only the root "iru" as a keyword. This presents a problem for mixed-initiative conversational systems which allow users to assert premises and requests in any sequence. Morphological analysis of verbs is required to identify the different speech acts in the conversation.

1.2 Syntactic issues

Tamil syntax presents diverse syntactic challenges in parsing the utterances. Some of the main issues are discussed in the following subsections.

Free word order constructions

Tamil is a restricted free word order language. It presents a major difficulty in constructing slot-based FSA. Slots are usually filled using positional and contextual information (provided by surrounding words like preposition) to identify the semantic role of the nouns. Since words are allowed to move around within a clause and the contextual information is embedded into the words themselves, there are a large number of syntactically correct paths. This increases the computational complexity of decoding.

CFGs, on the other hand are used to parse sentences based on relative positions of the constituents. Typically an English sentence will be analyzed as an NP (subject) followed by a VP (predicate). The constituents of these phrases are held in their positions. In Tamil however, the predicate can be broken and its constituents move freely around displaying a clear case of discontinuity. Thus CFGs are ineffective to parse Tamil utterances.

The following example explains the anomalous behavior of Tamil constructions. Let us consider a typical SVO construction in English and its acceptable translations in Tamil.

Example 1

"I gave him a pen"

"ndaan avanukku oru peenaa kotuththeen"
(I him a pen gave)

The following constructions are also acceptable.

"avanukku ndaan oru peenaa kotuththeen"
(him I a pen gave)

"oru peenaa ndaan avanukku kotuththeen"
(a pen I him gave)

"ndaan oru peenaa avanukku kotuththeen"
(I a pen him gave)

***Note**: The words in the parenthesis above denote word-to-word translation*

Case less noun constructions

Tamil allows case less noun constructions. In such cases, role players are identified by their relative positions. The following illustration explains this linguistic phenomenon.

Example 2

(1) "I want to go to Chennai"

(2) "ndaan cennaikku cellaveeNtum"
(I chennai want+go)

(3) "cennaikku ndaan cellaveeNtum"
(chennai I want+go)

(4) "ndaan cennai cellaveeNtum"
(I chennai want+go)

(5) "cennai ndaan cellaveeNtum"
(chennai I want+go)

Sentences (2), (3) and (4) are valid translations of sentence (1). In sentence (2), the noun "cennai" is case marked by "ku" morpheme clearly identifying its dative role in the sentence. In sentence (4), the noun plays the same role though it is not case marked. In such constructions, where explicit case markers are omitted, the roles of nouns are identified based on their relative positions as done in English. "cennai" is identified to play dative role in sentence (4) because of its proximity to the verb. For this reason, the nouns are not allowed to move freely in such constructions. Sentence (3) can be constructed by moving nouns around in sentence (2). However, a similar construction of sentence (5) from (4) is not allowed as it violates the semantics. A robust parser for Tamil should be able to handle both free word order and case less noun constructions.

Missing copular verbs
Tamil also allows verb less constructions. Copular verbs in present tense may be omitted in Tamil sentences.

Example 3

"cennaikku tikkat vilai evvaLavu *[aakum]"
(Chennai ticket how much is)

The copular verb "aakum" can be omitted sometimes leaving the sentence with no main verbs. In such cases, a robust dependency parser must handle the missing head constituent in the utterance.

2 Parsing and Understanding

This section presents dependency parsing coupled with morphological analysis to overcome the issues described in the previous section. It also explains the translation of dependency relations to semantic frames for dialog processing.

2.1 Dependency Parsing

Dependency parsing analyzes a sentence to identify the relationship between the words. In a dependency relation between two words, one will be the head and other a dependent. It is an alternative to the much common phrase structure analysis that outputs a parse tree built using CFG rules. The dependency relations are represented by their label followed by the head and dependent in parentheses. The following is a set of dependency relations for the sentence "Soldier saved the beautiful princess"

subj (saved, Soldier)
det (princess, the)
adj (princess, beautiful)
dobj (saved, princess)

Dependency analysis is very much suited for Tamil to extract the semantic relations. The dependency of one word on another in Tamil can be determined using the case inflection of the words and their relative positions. Typically, all the dependents occur before the head. This heuristic allows us to restrict the search. A variety of dependency parsers are available for English [Klein et al. 2002; Klein et al. 2003; Sleator et al 1993]. Our implementation of dependency parser is custom built for a train enquiry application. It takes the input utterance and returns a list of labeled dependencies. Parsing an utterance is done in a two step process.

Step 1: Morphological Analysis
Morphological analysis serves as a preprocessing step for dependency parsing. The incoming utterance is analyzed to identify the root forms and possible inflections. It also identifies multiword expressions like names of trains. The morphological analyzer is built using PC-Kimmo framework [Antworth 1990]. The lexicon is populated with domain specific vocabulary that includes all proper nouns like names of trains and stations. The output of this analysis is a set of feature structures represented in PATR-II formalism [Shieber 1986]. In addition to PoS, these feature structures provide a variety of information like case and post position inflections, mood, modal, tense, aspect of the words.

Example 4

USER: cennaikku tikkat evvaLavu vilai

The output of the morphological analysis is a list of feature structures as below,

cennaikku   cennai+kku   cennai+sg+dat

```
[ head:   [ case:  DAT
            gloss: cennai
            png:    [ num:   SG ]
            pos:   N
            ppos:  + ]
  cat:   Word ]
```

tikkat   tikkat+sg+nom

```
[ head:   [ case:  NOM
            gloss: tikkat
            png:    [ num:   SG ]
            pos:   N
            ppos:  + ]
  cat:   Word ]
```

evvaLavu   evvaLavu+nom

```
[ head:   [ case:  NOM
            gloss: evvaLavu
            mood:  int
            pos:   PN
            ppos:  + ]
  cat:   Word ]
```

vilai   vilai+sg+nom

```
[ head:   [ case:  NOM
            gloss: vilai
            png:    [ num:   SG ]
```

pos: N
ppos: +]
cat: Word]

The analyzer also acts as a filter to prune the out-of-vocabulary words and discontinuities that occur in spontaneous speech such as false starts, filled pauses, etc.

Step 2: Parsing

Dependency parsing decodes the grammatical relation between the words in the utterance. The parser identifies the dependency relations based on the relative positions and the unification of feature structures. Our parser treats every word as a dependent and pairs it to a head word. In Tamil, head words typically appear at the end of their respective phrase (head-final language). This heuristic restricts our search to the right-hand side of the sequence. For any two words the dependency is checked using a number of carefully hand-crafted rules. These rules make use of the feature structures of the words being examined. Feature structure unification is done to check for agreement between the words. Some example rules are given below for illustration.

Rule 1.
For any two words i & j,
if PoS(i) = NOUN & PoS(j) = VERB & Case(i) = NOM & unifiable(PNG(i), PNG(j))
then dependency = subject

Rule 2.
For any two words i & j,
if PoS(i) = ADJ & PoS(j) = N
then dependency = mod

Apart from general rules, there are lexicalized rules that capture the exceptions of the language. In the following rule, the modal verb "mutiyum" takes as subject a noun that is marked for ablative case and not a nominative case.

Rule 3.
For any two words i & j,
if PoS(i) = NOUN & PoS(j) = MODAL & GLOSS(j) = "mutiyum" & Case(i) = ABL
then dependency = subject

The search ends in a head for every dependent and the parser skips to the next word. This process repeats until the last word is reached. The last word will typically be a finite verb which is the head of the utterance. In case of missing copular constructions, where the sentence does not end with a finite copular verb, the ZERO feature structure is substituted for the missing verb. The output of the dependency parser for the above sentence is given below,

case_dat (cennai, ZERO)
case_nom (tikkat, ZERO)
int_mod (evvaLavu, vilai)
case_nom (vilai, ZERO)

In a conversational system, users' responses consist of incomplete constructions, which typically are single word answers to the system's prompts. In such cases, the parser provides a shallow dependency parse. The ability to handle elliptic constructions makes the parser robust to parse spontaneous dialogs.

Example 5

USER: cennai, will be parsed as,
case_nom(cennai, ZERO)

Dependency parser is independent of the domain of application. The output set of dependencies are based on grammatical relations between the words which are independent of the semantics of the domain. The parser can be easily ported to another domain or application, because of its domain independent nature.

2.2 Semantic translation

In the final step, the dependency relations identified by the parser are translated to semantic frames. These are slot-value pairs that are used by the dialog manager to retrieve the information the user needs. Hand-crafted rules are employed for this purpose.

The output of the translation is given below,
[travel_plan] cennaikku tikkat evvaLavu vilai
[to_station] cennai
[request_type] fare

In many cases, case and post-position inflections of the words determine the semantic labels, without using the dependency relations. However when there is no case-marking the system defaults to dependency relations to decide the semantic frames. This approach increases the efficiency of the parser as it has to compute dependency relations only when it cannot resolve the meaning using feature structures. For example, in cases like "cennai cellaveeNtum", there is no case marking and so dependency relations help decide the semantic frames. Query frames ([request_type]) are constructed when the utterance contains a request or a question. In this case, keyword matching is used along with dependencies, to identify the query frames. Ambiguous cases like case_nom(cennai, ZERO) produce [station] cennai and are resolved by the dialog manager using pragmatic and contextual information.

The output of the semantic translation is passed to the dialog manager which takes appropriate action to respond to the user's utterance. When porting the dialog system to a new domain, the Semantic Translation module needs appropriate updating of rules that map the dependencies to semantic frames of the new domain.

4 Train Enquiry System

We used our parser to develop a mixed-initiative natural language dialog system. The system currently takes input from the keyboard and provides answers through a console in text mode. The system provides information on trains, their time, travel fares and availability. It is built on MIT GALAXY, a distributed message-passing architecture for spoken dialog systems [Seneff et al., 1998]. The system incorporates a text based I/O console, parser, dialog manager, backend and natural language generation modules. All these servers and connected to a central hub which monitors the communications between them.

The dialog manager is designed using RavenClaw framework. It engages the user in an interactive conversation and gets necessary inputs to complete the task. The system allows for mixed-initiated dialogs, so both the user and system take mutual control over the conversation to complete a task as opposed to system-initiated VoiceXML like restricted dialogs. We are upgrading the system to include speech recognizer and synthesis modules to enable a speech based interface. The dialog manager can handle errors efficiently using robust error recovery strategies. It also uses explicit and implicit confirmations to establish proper grounding based on the confidence of the parsed output and dialog parameters. The backend implements a perl stub to query a static database and retrieve the results. The natural language generation module consists of templates and rewrite rules to map the dialog concepts from the dialog manager to system prompts to be presented to the user.

Sample Conversation:

SYSTEM:rayil vipara thaaniyangki unkaLai varaverrkirathu. ungaL payana viparangkaLai kooravum!
(Welcome to Train Information System. Please tell us your travel details!) [*Welcome Message*]
USER: cennaikku cellaveentum
(I want to go to Chennai)
SYSTEM: cennaikku celkiriirkaL, meelum..
(Traveling to Chennai, and..) [*Implicit confirmation*]
USER: tikkat vilai evvaLavu?
(How much does the ticket cost?) [*Query frame*]
SYSTEM: enkirundthu puRapatukiRiirkaL?
(Where are you boarding the train?) [*System Request*]
USER: kovai
(Kovai)
SYSTEM: cennaiyilirunthu koyampthurukku paacanjaril 124 rupaay, expressil 230 rupaay.
(From Chennai to Coimbatore, it is Rs. 124 in Passenger trains and Rs. 230 in Express trains.) [*Results*]
SYSTEM: veeru ethaavathu viparam aRiya virumpukiRiirkaLaa? [*Anything else*]
(Do you need more information?)
USER: illai. ndanRi.
(No. Thanks.)
SYSTEM: ungkaLukku indtha uraiyaatal eLithaakavum payanuLLathaakavum irundthathaa?
(Did you find this conversation easy and useful?)
USER: ... System archives the users response
SYSTEM: ungkal payanam inithaakuka!
(Wishing you a pleasant journey!) [*Goodbye*]

Note*: Dialog concepts are enclosed in square brackets*

5 Evaluation

We decided to evaluate the system as an application rather than just a parser. Evaluation was done based on the ability to get information on trains list, next train to station and ticket fare between two stations. 38 out of 50 users who evaluated the system were able to finish the tasks. On an average there were three turns per conversation. The summary of issues we faced in the failed 12 cases is as follows.

A. Vocabulary
Most of the users tried using spoken Tamil to interact with the system. Word forms in spoken Tamil are contracted and short when compared to their written counterparts. For instance "pookanum" is spoken version of "pookaveentum". These word forms were not handled by the morphological analyzer. Another case is the use of English words like 'train', 'ticket' etc. As our morphological analyzer is trained to handle only their phonetic equivalents like 'treyin', 'tikat' etc, their English counterparts were not recognized.

B. Typing
Typing Tamil using Roman scripts needs a little training. Although the users can figure out rough Roman transcription equivalents of Tamil scripts, they need to follow a standard chart. This issue will get automatically resolved when speech interface is used. Another related issue is that the users are not aware of which scripts to use. For now, the users expect the system to have an intelligent typing interface that will guess the user's typing.

C. Syntax
Although most words in the vocabulary have been covered, Tamil users tend to put together adjacent words into one single word. For instance "evvalavu aakum" will be presented as "evvalavaakum". In such cases, the dependencies were not created by the parser.

Although, the system handles only three basic tasks, we are planning to scale it up to handle more complex tasks like reservation of seats, etc. The current results are promising and encouraging.

6 Conclusion and Future work

We have described a robust, dependency parser that addresses various issues in parsing spontaneous Tamil utterances. Dependency checking rules are currently hand crafted. We would like to build Tamil dependency tree banks to work on automatic induction of rules. The architecture can be easily extended to other Indian languages. We have also presented a natural language Train Enquiry system that answers users queries in text mode. We are currently working on integrating our speech recognizer built using Sphinx II and limited domain speech synthesis modules to enable spoken language interface. In order to make the parser robust enough to handle spoken utterances, the differences between spoken and written Tamil need to be studied in detail. We would also like to investigate generating sentences from our parser to build language models for the front-end speech recognizer [Seneff, 1992].

References

[Allen et al., 2001] J Allen, D Byron, M Dzikovska, G Ferguson, L Galescu, and A Stent, "Towards Conversational Human-Computer Interaction", *AI Magazine*, 2001

[Antworth et al., 1990] Antworth, L Evan, "PC-KIMMO: a two-level processor for morphological analysis", *Pubs in Academic Computing No. 16, Summer Institute of Linguistics*, Dallas, USA, 1990

[Klein et al., 2002] Dan Klein and Christopher D. Manning, "Fast Exact Inference with a Factored Model for Natural Language Parsing", *In Proc. NIPS*, Vancouver, Canada, 2002

[Klein et al., 2003] Dan Klein and Christopher D. Manning, "Accurate Unlexicalized Parsing", *In Proc. 41st Meeting of ACL*, Sapporo, Japan, 2003

[Parikh et al., 2006] Tapan S. Parikh and Kaushik Ghosh, "Understanding and Designing for Intermediated Information Tasks in India", *IEEE Pervasive Computing Magazine*, 2006

[Plauché et al., 2006] Madelaine Plauché, Udhyakumar N, Divya Ramachandran, Joyojeet Pal and Chuck Wooters., "Speech recognition for illiterate access to information and technology", *In Proc. ICTD*, Berkeley, USA, 2006

[Raux et al., 2003] Antoine Raux, Brian Langner, Alan Black, and M Eskenazi, "LET'S GO: Improving Spoken Dialog Systems for the Elderly and Non-natives", *In Proc. Eurospeech*, Geneva, Switzerland 2003

[Reddy et al., 2006] Rami Reddy, Nandi Reddy and Sivaji Bandyopadhyay, "Dialogue based Question Answering System in Telugu", *In Proc. Workshop on Multilingual Question Answering, EACL*, Trento, Italy, 2006

[Seneff, 1992] Stephanie Seneff, "TINA: A Natural Language System for Spoken Language Applications", *Computational Linguistics*, Vol 18, No. 1, 1992

[Seneff et al., 1998] Stephanie Seneff, E Hurley, R Lau, C Pao, P Schmid, and V Zue, "GALAXY-II: A Reference Architecture for Conversational System Development", In Proc. ICSLP, Sydney, Australia, 1998

[Shieber, 1986] Stuart M Shieber, "An introduction to unification-based approaches to grammar", *CSLI Lecture Notes No. 4. Stanford, CA: Center for the Study of Language and Information,* 1986.

[Sleator et al., 1993] Daniel Sleator and Davy Temperley, "Parsing English with a Link Grammar", *Third Int'l Workshop on Parsing Technologies*, Tilburg, The Netherlands 1993

[Zue et al., 2000] Victor Zue, Stephanie Seneff, James Glass, Josepj Polifroni, Christine Pao, Timothy J Hazen and Lee Hetherington, "JUPITER: A Telephone-Based Conversational Interface for Weather Information", *IEEE Transactions on Speech and Audio Processing*, Vol. 8 , No. 1, 2000

Considering Multiple Options when Interpreting Spoken Utterances

Sarah George, Ingrid Zukerman, Michael Niemann and Yuval Marom
Faculty of Information Technology
Monash University
Clayton, VICTORIA 3800, AUSTRALIA
{sarahg,ingrid,niemann,yuvalm}@csse.monash.edu.au

Abstract

We describe *Scusi?*, a spoken language interpretation mechanism designed to be part of a robot-mounted dialogue system. *Scusi?*'s interpretation process maps spoken utterances to text, which in turn is parsed and then converted to conceptual graphs. In order to support robust and flexible performance of the dialogue module, *Scusi?* maintains multiple options at each stage of the interpretation process, and uses maximum posterior probability to rank the (partial) interpretations produced at each stage. The time and space requirements of maintaining multiple options are handled by means of an anytime search algorithm. Our evaluation focuses on the impact of the speech recognizer and the search algorithm on *Scusi?*'s performance.

1 Introduction

The DORIS project (*Dialogue Oriented Roaming Interactive System*) aims to develop a spoken dialogue module for a robotic agent. Eventually, this module will be able to engage in a dialogue with users and plan physical actions (by interfacing with a planner). In this paper, we describe *Scusi?*, the speech interpretation module that is being developed within the DORIS framework.

It is widely accepted that spoken dialogue systems are more prone to misinterpretations and partial interpretations than text-based systems. This may be attributed to the state of the art in speech recognition, and to people generally using more informal and less grammatical forms of expression in spoken discourse than in written discourse. In order to handle gracefully the additional uncertainty associated with the interpretation of spoken discourse, a dialogue module should be able to (1) *make decisions on the basis of the state of the interpretation process*, (2) *adjust these decisions dynamically on the basis of new information*, and (3) *recover from flawed or partial interpretations*. For example, if an addressee was reasonably sure that she heard a sentence correctly, her response would differ from the response she would generate if she couldn't quite distinguish between several possible sentences or parts thereof. If new information then came to light (e.g., the speaker just pointed to an object), it could change the certainty of the addressee regarding different interpretations. Still, it is possible that even the preferred interpretation has areas of uncertainty (e.g., it is not clear what the speaker wants done with the object in question). In this case, the addressee can just ask a clarification question regarding the intended action.

Scusi? was designed to enable a dialogue module to achieve the above requirements. *Scusi?*'s interpretation process comprises three main stages (Figure 1): speech recognition, parsing, and semantic interpretation. During semantic interpretation a parse tree is first mapped into a knowledge representation based on *Conceptual Graphs (CGs)* [Sowa, 1984]; this is similar to the assignment of semantic role labels [Gildea and Jurafsky, 2002]. The content of this CG structure is then matched with items and actions in the world (Section 3). Each stage in the interpretation process produces multiple candidate options, which are ranked according to their probability of being intended by the speaker. The probability of a candidate depends on the probability of its parents (generated in the previous stage of the interpretation process) and that of its components (Section 4).

The generation and maintenance of multiple interpretations, and the calculation and update of their probability contribute to the above requirements for a dialogue module as follows.

1. The generation and maintenance of multiple interpretations and the calculation of the probability of an interpretation at each stage of the interpretation process enable the dialogue module to *make decisions on the basis of features of the overall state of the interpretation process*. Examples of such features are: how many highly ranked interpretations there are, how similar they are to each other, and how confident is the system about its interpretations at the different stages. For instance, if there are several top-ranked interpretations, it is reasonable to generate a clarification question that discriminates between them; if all the interpretations produced by the speech recognizer have a low probability, then the dialogue module can initiate a clarification sub-dialogue regarding the spoken utterance; and if the parse tree used for the top interpretation has a low probability, the dialogue module may ask *Scusi?* to perform additional processing using other parse trees.

2. The calculation and update of the probability of interpre-

Proceedings of the 5th *Workshop on Knowledge and Reasoning in Practical Dialogue Systems*, pages 7–14,
Hyderabad, January 8th, 2007.

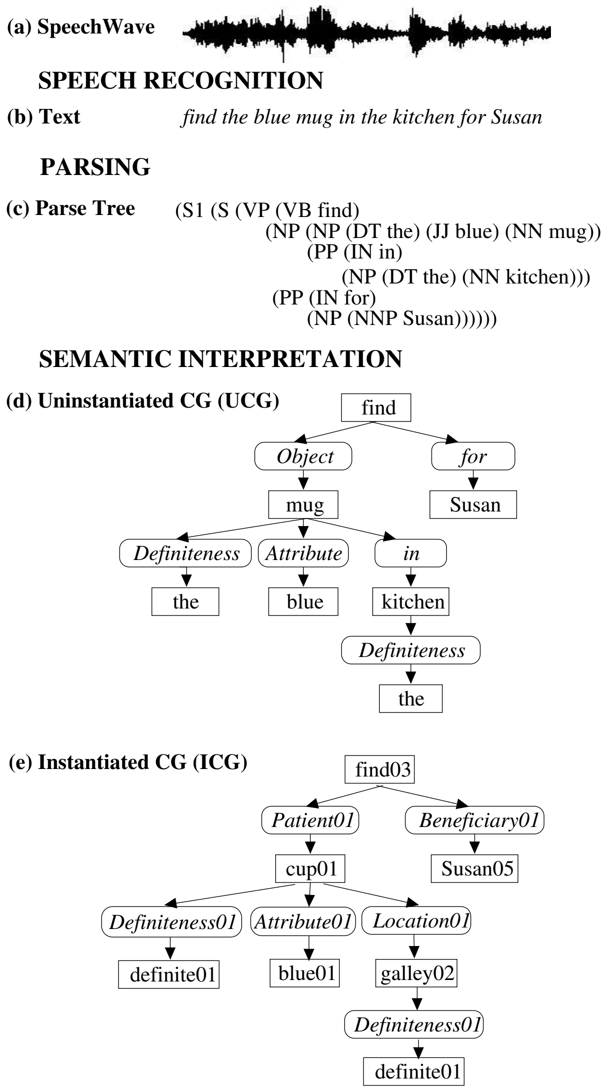


Figure 1: Structures for the interpretation stages

tations supports a dynamic re-ranking of the interpretations as new information becomes available, which in turn enables the dialogue module to *modify its decisions on the fly*. This information may be obtained from additional interpretations generated by *Scusi?* (after it has submitted its current interpretations to the dialogue module), or from new observations, which may be received from a vision module or from a new utterance generated by the user. For example, a clarification question may no longer be required if a newly produced interpretation has a much higher probability than any interpretation generated so far, or if a new visual input can disambiguate between several top-ranked interpretations.

3. As mentioned above, the process of calculating the probability of an interpretation incorporates the calculation of the probability of individual components of the interpretation. This supports the identification of "trusted" (high probability) and "untrusted" (low probability) regions of an interpretation, which enables the dialogue module to select strategies to *recover from flawed or partial interpretations*. For instance, if the speech recognizer is not confident about some words, the resultant interpretation will have low-probability components that correspond to these words; or if a concept in a final interpretation (CG) does not match a domain expectation (e.g., an object to be moved is not movable), the probability of the corresponding component will be low. The identification of these untrusted regions will enable the dialogue module to initiate a focused recovery, such as a clarification question about the components in an untrusted region, e.g., for the first example, it may ask "What do you want me to get?", and for the second example, it may inquire "I understood you want me to move your room. Is this what you meant?".

The rest of this paper is organized as follows. Section 2 presents our interpretation process, followed by a description of conceptual graphs – our knowledge representation formalism. Our probabilistic approach is discussed in Section 4, and an initial evaluation of our interpretation mechanism is presented in Section 5. Section 6 discusses related research, followed by concluding remarks.

2 Multi-Stage Processing

Figure 1 illustrates the stages involved in processing spoken input. The first stage activates an *Automatic Speech Recognizer (ASR)* to generate candidate sequences of words (*Text*) from a *Speech Wave*.[1] Each Text has a score that represents how well its words fit the speech wave. This score is converted into a probability. The word sequences are then parsed using a probabilistic parser, which generates a set of *Parse Trees*.[2]

The last two stages of the interpretation process generate two types of CGs: *Uninstantiated Concept Graphs (UCGs)* and *Instantiated Concept Graphs (ICGs)*. UCGs are obtained from Parse Trees, where one Parse Tree produces one UCG (Section 3.1). UCGs represent mainly syntactic information, i.e., the concepts in a UCG correspond to the words in the parent Parse Tree, and the relations between the concepts are directly derived from syntactic information in the Parse Tree and prepositions. For instance, in the example in Figure 1(c-d), the noun "mug" is mapped to the concept mug, and the preposition "in" in the Parse Tree is mapped to the relation *in* in the UCG. Next, *Scusi?* proposes candidate ICGs for UCGs, where one UCG may yield several ICGs. This is done by nominating *Instantiated Concepts* from DORIS's knowledge base as a potential realization for every *Uninstantiated Concept* in a UCG (Section 3.2). In the example in Figure 1(d-e), the concept mug is mapped to cup01, and the relation *in* in the UCG is mapped to *Location01* in the ICG.

[1]We are currently using ViaVoice (http://www-306.ibm.com/software/voice/viavoice), and trialling Sphinx (http://cmusphinx.sourceforge.net/).

[2]We use Charniak's parser (ftp://ftp.cs.brown.edu/pub/nlparser/) because it can produce partial parses for ungrammatical utterances, and it provides multiple parse trees. This is in line with our approach, which expects multiple options at each stage of the interpretation process.

2.1 Anytime processing

The consideration of all possible options for each stage of the interpretation process is computationally intractable. To address this problem, we have adapted the *anytime* algorithm described in [Niemann *et al.*, 2005], which applies a selection-expansion cycle to build a search graph as follows. The selection step nominates a single sub-interpretation (Speech, Text, Parse Tree or UCG) to expand, and the expansion step generates only one child for that sub-interpretation. The selection step then nominates the next sub-interpretation to expand, which may be the one that was just expanded, its new child, or any other sub-interpretation in the search graph.

The selection-expansion cycle is repeated until one of the following happens: all the options are fully expanded, a time limit is reached, or the system runs out of memory. At that point, the interpretation process returns all the (ranked) interpretations and sub-interpretations obtained so far. This will enable the dialogue module to decide on an action on the basis of the overall state of the interpretation process. If memory hasn't run out, the interpretation process will continue cycling, and if it finds new high-probability interpretations, the dialogue module can adjust its actions accordingly.

Our search algorithm differs from most search algorithms for spoken language interpretation in two respects: (1) it implements a *stochastic optimization strategy*, and (2) it dynamically decides which level in the search graph and which node within this level to expand next.

Stochastic optimization strategies. These strategies, which include simulated annealing and neural nets, occasionally allow low-ranking nodes to generate children. In so doing, these strategies typically avoid getting stuck in local maxima — a problem incurred by greedy algorithms.

Dynamic node selection. Many spoken language interpretation systems apply some type of level-building algorithm [Myers and Rabiner, 1981], which expands each level of the search in turn. In order to curb combinatorial explosion, a beam threshold, which selects the best K options, is used at each level (typically, the value of K is quite small, allowing only the best or top-few interpretations to proceed [Shankaranarayanan and Cyre, 1994; Gorniak and Roy, 2005]). In contrast, our search dynamically determines the stage (level in the search graph) to be expanded, selects a node within that level, and generates one child for this node. In line with our stochastic optimization approach, the first two decisions are probabilistic, choosing preferred options most often, but not always. In order to encourage the early generation of complete interpretations, preference is given to later stages in the search (e.g., expanding UCGs rather than Texts). Within a level, nodes with a proven "track record" are preferred, i.e., nodes that have previously produced high-probability children. This heuristic cannot be used by a level-building algorithm, as information about later stages is not available to earlier stages. In Section 5, we compare the performance of our search with that of a level-building algorithm.

3 Conceptual Graphs

Conceptual graphs represent entities and the relationships between them.[3] For instance, the CG in Figure 1(e) indicates that there are two concepts find03 and cup01 that have a *Patient01* relationship. Every relationship in a CG must have at least one parent concept and one child concept, e.g., the *Patient01* relationship in Figure 1(e) has concept find03 as a parent and cup01 as a child. However, a concept can exist in isolation without any relationships. This supports phrases as well as single-word utterances like "yes", "there" and "Mary".

3.1 Uninstantiated Conceptual Graphs

A UCG represents concepts and relationships that can be obtained directly from the Parse Tree (without resorting to domain knowledge). Most phrases in a Parse Tree map to a concept node representing their head-word. If a phrase governs a word or phrase other than its head-word, then the phrase's concept node is joined to the other word or phrase's concept node via a relationship node. For instance, the adjective (JJ) *"blue"* in Figure 1(c), which is governed by the same NP as the noun (NN) *"mug"*, is connected to the mug concept in the UCG in Figure 1(d) by means of an *Attribute* relationship node (*Attribute* is the default relationship). Linguistic details such as part-of-speech and phrasal category are retained as features of the concepts. Prepositions are treated as defining a relationship node between two concepts, rather than mapping to concept nodes, e.g., *for* represents the relationship between find and Susan in Figure 1(d). It is worth noting that slightly different Parse Trees may yield the same UCG. For instance, the blue concept node in Figure 1(d) could also be generated from an Adjectival Phrase, instead of a stand-alone adjective (JJ) adjunct to the NP.

This representation allows *Scusi?* to accept and combine information from different types of sentences and input modalities. For spoken input, our mapping from Parse Tree to UCG handles declarative, imperative and interrogative sentences, as well as single words. In the future, *Scusi?* is expected to interact with the scene analysis component of a robot's vision system. This component will return objects and relationships such as *Coordinates*, *Colour* or *Shape*, which can readily map to UCGs.

3.2 Instantiated Conceptual Graphs and the Knowledge Base

The generation of an ICG requires the selection of an *Instantiated Concept* from the knowledge base for each *Uninstantiated Concept* in a UCG. The knowledge base contains entries for the following types of concepts.

- Specific real-world objects, e.g., cup03, Susan05;
- general objects that have default features, e.g., CupClass01, which has features like container=Y, movable=Y, shape=cylinder and size=small;

[3]Our knowledge representation is structurally like CGs, but the relations are inspired by the Verb Semantic Classes from EAGLES96 (`http://www.ilc.cnr.it/EAGLES96/rep2/node10.html`) and by FrameNet categories (`http://framenet.icsi.berkeley.edu/`).

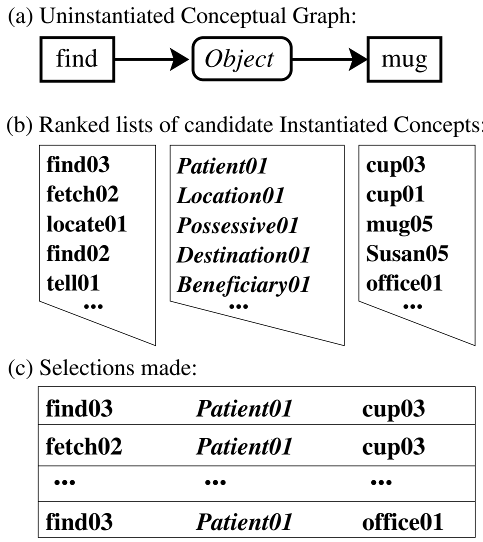


Figure 2: Selection of Instantiated Concepts

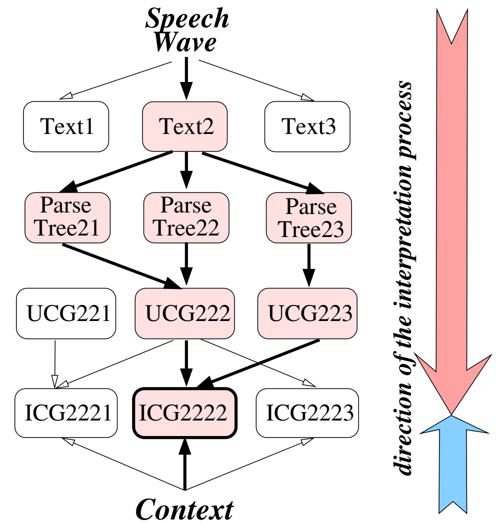


Figure 3: Interpretation process

- abstract attributes like blue01 and quickly01;
- actions known to the system, e.g., find01 for locating a place and reporting its whereabouts, as in "find an office for Susan", and find03 for retrieving an object, as in "find a cup for Susan"; and
- instantiated Relationships, e.g., roles like *Patient01*, *Destination01* and *Beneficiary01*.

The process for postulating Instantiated Concepts for Uninstantiated ones is similar to that used for suppositions in [George *et al.*, 2005]. Each Uninstantiated Concept in a UCG is associated with a list of Instantiated Concepts (Figure 2(b)). Each entry in the list is assigned a probability on the basis of how well it matches the Uninstantiated Concept (Section 4). To generate an ICG, one Instantiated Concept is selected from the list of each Uninstantiated Concept in the parent UCG, starting with the higher-ranked combinations (Figure 2(c)). Subject to time and memory limitations, all combinations of Instantiated Concepts may eventually be considered.

4 Probabilities of Interpretations

Scusi? ranks candidate ICGs according to their posterior probability in light of a given Speech Wave and conversational context. At present, the context is obtained from concept and relation instances in the system's knowledge base, which in the absence of other information are equiprobable. We are currently in the process of incorporating salience from dialogue history into our formalism, such that it influences the prior probability of mentioning a concept or relation. In the future, we will also include information from the robot's vision system.

As seen in Section 2, the interpretation process goes mainly from evidence (Speech Wave) to ICG (thick arrows on the right-hand-side of Figure 3). The ASR provides probabilities from Speech Wave to Text (its scores are directly translated to probabilities), and the probabilistic parser from Text to Parse Tree. We therefore calculate the posterior probability of an ICG as follows.

$$\Pr(\mathit{ICG}|\mathit{Speech}) \cong \alpha \times \sum_{txt,prsTr,ucg} \left\{ \begin{array}{l} \Pr(\mathit{ICG}|\mathit{UCG},\mathit{Context}) \times \Pr(\mathit{UCG}|\mathit{ParseTr})\times \\ \Pr(\mathit{ParseTr}|\mathit{Text}) \times \Pr(\mathit{Text}|\mathit{Speech}) \end{array} \right\} \quad (1)$$

where α is a normalizing constant.

The summation is required since, as seen in Figure 3, a sub-interpretation may have multiple parents. The retention of all sub-interpretations regardless of their probability enables us to get a true measure of the overall probability of any interpretation with multiple paths to the Speech Wave.

Further, since a UCG is generated algorithmically from a Parse Tree, $\Pr(\mathit{UCG}|\mathit{ParseTr}) = 1$. Thus, the only outstanding issue is the calculation of $\Pr(\mathit{ICG}|\mathit{UCG},\mathit{Context})$. To perform this calculation we make the following simplifying assumptions.

- $\Pr(\mathit{ICG}|\mathit{UCG},\mathit{Context})$ can be calculated separately for each node (concept or relation) in the ICG; and
- given a source UCG and the Context, the probability of each node N_i^{ICG} in an ICG depends on its corresponding node in the UCG (N_i^{UCG}), its neighbouring nodes in the ICG ($\mathit{nbours}(N_i^{ICG})$), and its prior probability in the Context.

These assumptions yield the following formulation.

$$\Pr(\mathit{ICG}|\mathit{UCG},\mathit{Context}) = \prod_{i=1}^{n} \Pr(N_i^{ICG}|N_i^{UCG}, \mathit{nbours}(N_i^{ICG}), \mathit{Context}) \quad (2)$$

where n is the number of nodes in the ICG.

By making some conditional independence assumptions, we obtain the following approximation for this equation.

$$\Pr(\mathit{ICG}|\mathit{UCG},\mathit{Context}) = \beta \times \prod_{i=1}^{n} \left\{ \begin{array}{l} \Pr(N_i^{UCG}|N_i^{ICG}) \times \Pr(\mathit{nbours}(N_i^{ICG})|N_i^{ICG})\times \\ \Pr(N_i^{ICG}|\mathit{Context}) \end{array} \right\} \quad (3)$$

Table 1: Features for sample concepts in the UCG and ICG

Stage	called	PoS	rel	cg-role
UCG	"find"	{S1,S,VP,VB,word}	–	concept
UCG	"fine"	{S1,ADJP,JJ,word}	–	concept
ICG find01	"find, locate"	{VP,VB,VBZ,VBP, VBN,VBG,VBD}	–	concept
ICG find03	"find, locate"	{VP,VB,VBZ,VBP, VBN,VBG,VBD}	–	concept

where β is a normalizing constant.

- The third factor in this product contains the prior probability of node i in ICG, which reflects the salience of the concept or relation in question in the current context.
- The second factor reflects how reasonable it is to put the concepts of the ICG together, i.e., it encodes the extent to which each node in the ICG matches the requirements of other nodes. For example, find03 in Figure 1(e) expects a *Patient01* relationship and a *Beneficiary01* relationship with other concepts.
- The first factor in the product represents how well a candidate ICG node matches a source UCG node. We have found it useful to consider four features of these nodes to determine the goodness of this match (described below). The values of these features for a UCG node are obtained from the parser, and the possible values that can be taken by a candidate ICG node are stored in the knowledge base.

1. called – the lexical items associated with a concept, e.g., "find" and "mug" in the UCG in Figure 1(d). This feature is used to determine whether the words in a user's utterance could be used to designate a candidate concept or relation in the knowledge base. In the future, we intend to complement this feature with similarity metrics such as those discussed in [Pedersen *et al.*, 2004].
2. PoS – part of speech. This feature is more forgiving than called, because only some of the PoS-tags returned by a parser inform the matching process.
3. relation – syntactic relation, e.g., *Object* and *Attribute* in the UCG in Figure 1(d). This feature has a value when the parser provides information about the type of a relation between concepts. Like called, this feature is used to determine whether the relations in a user's utterance could be used to designate a candidate relation in the knowledge base.
4. cg-role – the semantic role of a node, i.e., concept or relation. This feature is crucial for determining whether a candidate ICG node could possibly match an uttered word.

To illustrate the calculation of these factors, consider a situation where given the input in Figure 1, the ASR has alternatively heard "find" and "fine" as the first word. Table 1 shows the above features for "find" and "fine" in the UCG, and for two candidate domain actions in the ICG: find01 and find03.

Both UCG words match the cg-role and relation features for both domain actions. However, "fine" does not match the called and PoS features (the parse that produced the UCG in this example does not have "fine" as a verb). Hence, the UCG with "find" has a higher probability of being the parent of an ICG that has find01 as a candidate action, and an ICG that has find03 as a candidate action. After calculating the first factor in Equation 3, the find nodes in these ICGs are equiprobable. However, the second factor discriminates between these domain actions. This is done through the expectations of action-relation pairs. For example, find03 expects an inanimate object as a patient, while find01 expects a place. Since "cup01" is an object, find03 has a higher probability. At present, *Scusi?* interprets utterances in isolation, hence dialogue context has no influence.

5 Evaluation

To evaluate *Scusi?*'s speech interpretation performance, we used 27 utterances, which were based on the TRAINS92 corpus [Allen *et al.*, 1996] and were spoken by one of the authors. These utterances were selected due to their simplicity, so that they are easy to parse. The utterances had different lengths, ranging from 3 to 13 words, e.g., *"back to Illinois"* and *"bring the boxcar back to Avon to fill the boxcar up with bananas"*. The knowledge base had 139 concepts (e.g., go0, town_Avon).

Our evaluation focuses on *Scusi?*'s ability to generate the intended interpretation (hence measures of partial matches, such as Word Error Rate, are not appropriate). We defined two gold standards as follows. Each utterance had one Gold Text which was the original TRAINS text (the ASR could produce the Gold Text for 23 of the 27 utterances considered — our evaluation is based on those 23 utterances). In addition, each utterance had zero or more Gold ICGs among the ICGs generated by *Scusi?* (sometimes *Scusi?* could not find a correct ICG, and sometimes there were several appropriate interpretations). The correctness of an ICG was determined on the basis of the knowledge base. If an Uninstantiated Concept did not have a corresponding concept in the knowledge base, then the Gold ICG mapped it to a generic concept, e.g., unknown_concept, unknown_noun. Otherwise, the Gold ICGs contain concepts from the knowledge base that are valid representations of what the speaker uttered, e.g., the Gold ICG for "go to Corning" is go0 $\rightarrow$Destination_Relationship $\rightarrow$town_Corning.

Our evaluation focuses on two aspects of *Scusi?*'s operation: (1) the effect of speech recognition performance on interpretation performance, and (2) our search algorithm.

Effect of speech recognition performance. Our ASR often produces a large number of options for a spoken utterance. For instance, the utterance *"We were going to take the red engine"* yields 7,682 options, and *"Pick up a tanker in Corning I guess"* produces 53,762 alternatives. Figure 4 shows the number of Gold ICGs found by *Scusi?* as a function of the error percentage of the ASR. This is the proportion of incorrect outputs produced by the ASR that were used in the interpretation process. 0% error means that the ASR was deemed to return only the Gold Text, and 100% means that the ASR Gold

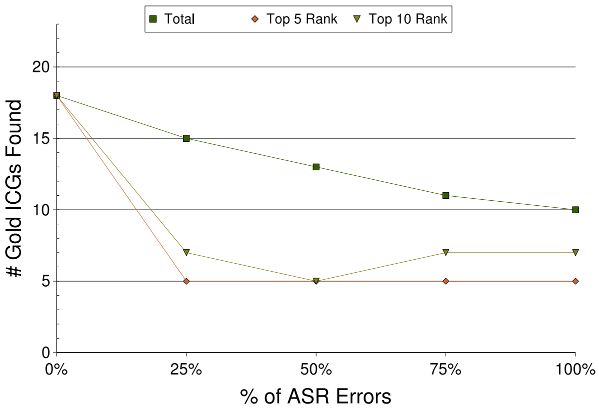


Figure 4: Effect of speech recognition errors, 300 iterations

Text was mixed in among all the erroneous options produced by the ASR (clearly, when thousands of options are produced by the ASR, not all of them can be considered in the available time). The 25, 50 and 75 error percentages were produced by selecting every fourth, second, and three out of four erroneous options produced by the ASR. *Scusi?* was run for 300 iterations, which is our current default setting for tests.

The graph also shows how error percentage affects the number of Gold ICGs with *average rank* ≤ 5 and ≤ 10, where average rank is the average of the ranks of equiprobable interpretations, e.g., if there are three equiprobable interpretations ranked 1, 2 and 3, their average rank is 2. We used *average rank* rather than raw rank because it is often the case that clusters of interpretations have the same probability.

As expected, ASR accuracy has a significant influence on interpretation performance, with accurate recognition yielding 18 Gold ICGs out of the possible 23, and all these ICGs being ranked top 5. As ASR accuracy drops, so does interpretation performance. However, this deterioration is graceful in terms of our system's ability to find Gold ICGs, while it is sudden for average rank. Further, the number of Gold ICGs with ranks ≤ 5 and ≤ 10 remains constant between ASR error of 25% and 100%. We propose to investigate two ways to obtain better output from the ASR: we are considering procedures for filtering the options returned by ViaVoice, and in parallel we are experimenting with a different ASR.

***Scusi?*'s search algorithm.** Figure 5 compares the performance of our search algorithm with that of a level-building algorithm that uses beams of different sizes. Since the level-building algorithm expands each level in turn, a sub-interpretation does not have information about the performance of its children. Hence, unlike *Scusi?*-search, beam search selects the top sub-interpretations to be expanded on the basis on their probability only (Section 2). Figure 5(a) shows the number of ASR Gold Texts found by both algorithms (from the possible 23), and Figure 5(b) shows the number of Gold ICGs. Both figures show the total number of Golds found, and the number with average ranks of 1, ≤ 5 and ≤ 10. The plain-coloured bars show the performance of the level-building algorithm for beams of size 1, 5 and 10, and the bars with diagonal stripes show the performance of the "corresponding" *Scusi?*-search. This corresponding search was defined in order to make the comparison fair — the number of iterations it performs is equal to the number of iterations performed by the beam search. For example, Beam-1 means that only the top-ranked option was expanded by the beam search at every stage, which is equivalent to 6 *Scusi?* iterations; Beam-10 is equivalent to *Scusi?*-350 (note that *Scusi?*-350 finds 12 Gold ICGs, compared to 10 Gold ICGs found by *Scusi?*-300, plotted in Figure 4).

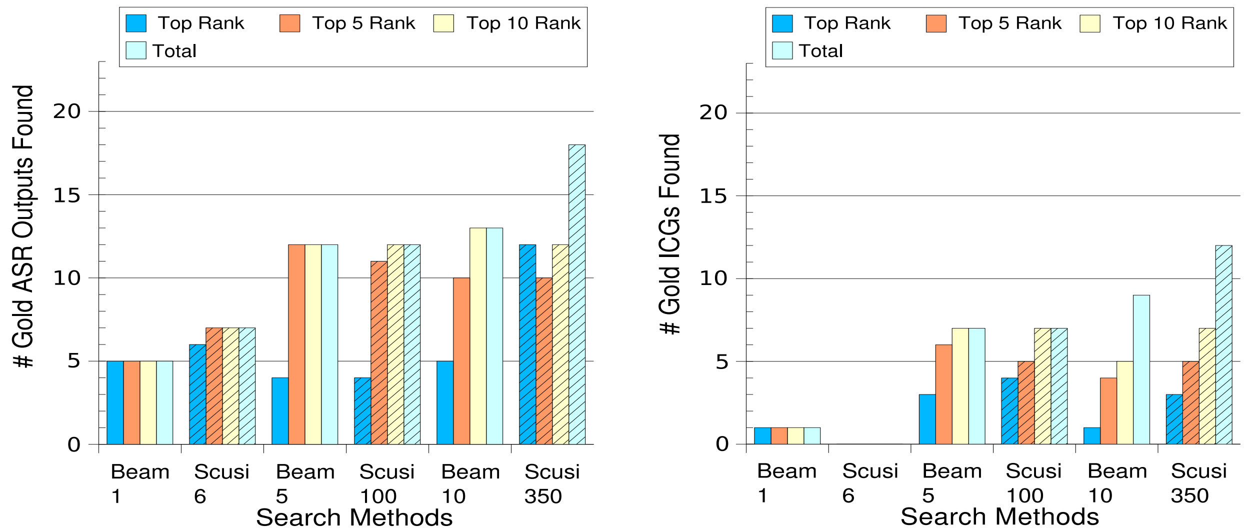
As seen in Figure 5, the performance of *Scusi?*-search is slightly better than that of beam search, in particular for *Scusi?*-350 versus Beam-10. *Scusi?*-350 found 18 Gold Texts, which led to 12 Gold ICGs, while Beam-10 found 13 Gold Texts, which led to 9 Gold ICGs. Note that the additional Gold Texts found by *Scusi?* have ranks greater than 10, and hence are unlikely to be found by a rigid beam search. This indicates that *Scusi?*'s flexible expansion procedure is a promising approach. Additionally, *Scusi?*'s anytime performance (Section 2) makes it more responsive to its operating conditions than systems governed by arbitrary thresholds (e.g., beam size). Hence, we consider this approach worth pursuing. Also note that our results, both for beam search and *Scusi?*-search, are heavily influenced by our calculations of the probability of a sub-interpretation. We expect that additional information brought to bear to these calculations, such as corpus-based statistics, will yield improvements in interpretation performance.

6 Related Research

This research extends the work described in [Niemann *et al.*, 2005] in its use of CGs as its main knowledge representation formalism. CGs were chosen, instead of the simple frames used by Niemann *et al.*, due to their higher expressive power (the relationship between CGs and predicate calculus is discussed in [Dau, 2001]). The use of CGs in turn affects the calculation of the posterior probability of an interpretation.

Miller *et al.* [1996] and He and Young [2003] also applied a probabilistic approach for the interpretation of utterances from the ATIS corpus, and Pfleger *et al.* [2003] used this approach to interpret multi-modal input (but using a scoring function, rather than probabilities). However, these three projects use semantic grammars for parsing, while *Scusi?*'s interpretation process initially uses generic, syntactic tools, and incorporates semantic- and domain-related information only in the final stage of the process. Knight *et al.* [2001] compared the performance of a grammar-based dialogue system to that of a system based on a statistical language model and a robust phrase-spotting grammar. The latter performed better for relatively unconstrained utterances by users unfamiliar with the system. Our probabilistic approach and intended audience are in line with this finding.

Like us, Fischer *et al.* [1998] regarded speech interpretation as an optimization task. They achieved anytime performance by employing a stochastic optimization method which considers multiple interpretations and expands "sub-optimal" candidates. However, their use of statistical information is fundamentally different from ours, as they use the results of



(a) Number of ASR Gold Texts

(b) Number of Gold ICGs

Figure 5: Comparison between *Scusi?*-search and beam search, 100% ASR error

statistical analysis to prime the interpretation process. Additionally, they worked on railway schedule queries, which are stylistically constrained.

Sowa and Way [1986] and Shankaranarayanan and Cyre [1994] used conceptual graphs for discourse interpretation. Both used a predefined set of canonical graphs to define the semantics of the base concepts in their system. *Scusi?* differs from both of these systems in its use of the UCG as an intermediate stage that is independent from the semantic- and domain-knowledge in the knowledge base. From a processing point of view, Shankaranarayanan and Cyre considered only the first parse tree that supports an acceptable interpretation, rather than retaining multiple parse trees. Sowa and Way allowed multiple interpretations, but applied a filtering mechanism that removed parses that failed semantic expectations. *Scusi?* does not apply such filtering, allowing possibly flawed candidates to undergo a deeper examination.

Our work resembles that of Horvitz and Paek [1999] and Gorniak and Roy [2005] in its integration of context-based expectations with alternatives obtained from spoken utterances. Gorniak and Roy use a probabilistic parser like ours, but they restrict the search space by training the parser on a corpus of human interactions relating to a computer game, and provide tightly constrained domain expectations based on the appropriate actions at particular stages in the game. In addition, they allow only the most probable parse state to generate an interpretation. In contrast, we do not restrict our expected input, we only factor in domain knowledge in the final stage of the interpretation, and dis-preferred sub-interpretations are allowed to proceed to the next stage. The differences between these approaches highlight important trade-offs between processing speed, flexibility and robustness.

Horvitz and Paek focused on higher level informational goals than those addressed in this paper, using a single output produced by a parser as linguistic evidence for their goal recognition system. An important aspect of their work, which we hope to incorporate into our dialogue module in the future, is their use of a utility-based decision procedure to determine the system's actions on the basis of the probabilities of interpretations.

7 Conclusion

We have presented a multi-stage interpretation process that maintains multiple options at each stage of the process, and uses maximum posterior probability to rank the (partial) interpretations produced at each stage. We have argued that these features support the following desirable behaviours in a dialogue module: making decisions on the basis of the state of the interpretation process, adjusting these decisions dynamically on the basis of new information, and recovering from flawed or partial interpretations.

The time and space requirements of maintaining multiple options are handled by means of an anytime search algorithm. Our algorithm dynamically decides which level to expand in a search graph, and which node within a level. This supports flexible behaviour that takes into account a system's operating constraints. Additionally, our algorithm employs a stochastic optimization method, which allows the examination of sub-optimal sub-interpretations.

Our evaluation considered two aspects of the interpretation of spoken discourse: (1) impact of ASR performance, and (2) search algorithm. As expected, ASR performance affects our system's interpretation performance overall. However, in our experiments, our system's ability to produce highly-ranked interpretations was invariant for ASR error percentages between 25% and 100%. Our search algorithm performed slightly better than a traditional beam-search approach. This, together with our algorithm's flexibility, indicate that our approach is worth pursuing.

Acknowledgments

This research was supported in part by the ARC Centre for Perceptive and Intelligent Machines in Complex Environments. The authors thank Eugene Charniak for his modifications to his probabilistic parser, Charles Prosser for his assistance in extracting multiple texts from ViaVoice, David Albrecht for his advice on the probabilistic formulation, and James Allen for permission to use and modify the transcripts from the TRAINS project [Allen *et al.*, 1996].

References

[Allen *et al.*, 1996] J.F. Allen, B.W. Miller, E.K. Ringger, and T. Sikorski. A robust system for natural spoken dialogue. In *Proceedings of the Thirty-Fourth Annual Meeting of the Association of Computational Linguistics (ACL)*, pages 62–70, Santa Cruz, California, 1996.

[Dau, 2001] F. Dau. Concept graphs and predicate logic. In *ICCS 2001 – Proceedings of the 9th International Conference on Conceptual Structures*, Stanford, California, 2001.

[Fischer *et al.*, 1998] J. Fischer, J. Haas, E. Nöth, H. Niemann, and F. Deinzer. Empowering knowledge based speech understanding through statistics. In *ICSLP'98 – Proceedings of the Fifth International Conference on Spoken Language Processing*, volume 5, pages 2231–2235, Sydney, Australia, 1998.

[George *et al.*, 2005] S. George, I. Zukerman, and M. Niemann. Modeling suppositions in users' arguments. In *UM05 – Proceedings of the 10th International Conference on User Modeling*, pages 19–29, Edinburgh, Scotland, 2005.

[Gildea and Jurafsky, 2002] D. Gildea and D. Jurafsky. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288, 2002.

[Gorniak and Roy, 2005] P. Gorniak and D. Roy. Probabilistic grounding of situated speech using plan recognition and reference resolution. In *ICMI'05 – Proceedings of the Seventh International Conference on Multimodal Interfaces*, Trento, Italy, 2005.

[He and Young, 2003] Y. He and S. Young. A data-driven spoken language understanding system. In *ASRU'03 – Proceedings of the IEEE Automatic Speech Recognition and Understanding Workshop*, St. Thomas, US Virgin Islands, 2003.

[Horvitz and Paek, 1999] E. Horvitz and T. Paek. A computational architecture for conversation. In *UM99 – Proceedings of the Seventh International Conference on User Modeling*, pages 201–210, Banff, Canada, 1999.

[Knight *et al.*, 2001] S. Knight, G. Gorrell, M. Rayner, D. Milward, R. Koeling, and I. Lewin. Comparing grammar-based and robust approaches to speech understanding: A case study. In *Proceedings of Eurospeech 2001*, Aalborg, Denmark, 2001.

[Miller *et al.*, 1996] S. Miller, D. Stallard, R. Bobrow, and R. Schwartz. A fully statistical approach to natural language interfaces. In *ACL96 – Proceedings of the 34th Conference of the Association for Computational Linguistics*, pages 55–61, Santa Cruz, California, 1996.

[Myers and Rabiner, 1981] C. Myers and L. Rabiner. A level building dynamic time warping algorithm for connected word recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 2:284–297, 1981.

[Niemann *et al.*, 2005] M. Niemann, S. George, and I. Zukerman. Towards a probabilistic, multi-layered spoken language interpretation system. In *Proceedings of the Fourth IJCAI Workshop on Knowledge and Reasoning in Practical Dialogue Systems*, pages 8–15, Edinburgh, Scotland, 2005.

[Pedersen *et al.*, 2004] T. Pedersen, S. Patwardhan, and J. Michelizzi. WordNet::Similarity – measuring the relatedness of concepts. In *AAAI-04 – Proceedings of the Nineteenth National Conference on Artificial Intelligence*, pages 25–29, San Jose, California, 2004.

[Pfleger *et al.*, 2003] N. Pfleger, R. Engel, and J. Alexandersson. Robust multimodal discourse processing. In *Proceedings of the Seventh Workshop on the Semantics and Pragmatics of Dialogue*, Saarbrücken, Germany, 2003.

[Shankaranarayanan and Cyre, 1994] S. Shankaranarayanan and W.R. Cyre. Identification of coreferences with conceptual graphs. In *ICCS'94 – Proceedings of the Second International Conference on Conceptual Structures*, College Park, Maryland, 1994.

[Sowa and Way, 1986] J.F. Sowa and E.C. Way. Implementing a semantic interpreter using conceptual graphs. *IBM Journal of Research and Development*, 30(1):57–69, 1986.

[Sowa, 1984] J.F. Sowa. *Conceptual Structures: Information Processing in Mind and Machine*. Addison-Wesley, 1984.

Dialogue Generation for Robotic Portraits

Andrew M. Olney
University of Memphis
Institute for Intelligent Systems
aolney@memphis.edu

Abstract

This paper examines the effectiveness of two domain independent methods of generating dialogue in a conversational robot. The goal is to create interactive dialogues based on the interviews, writings, and biographies of a science fiction author without resorting to domain specific knowledge engineering. Judges read conversations between the author and an interviewer, where the last utterance is either a continuation of the original dialogue, randomly generated, or generated using a combination of shallow semantic, statistical, and syntactic techniques. By rating the last utterance on several scales related to Grice's conversational maxims and the Bystander Turing Test, the expert judges enable comparison amongst the dialogue generation methods for this specific domain. The dialogue generation methods were then incorporated into a robotic portrait of the science fiction author.

1 Introduction

How can we best hold a natural conversation with a machine? The difficulties associated with an unrestricted natural language interface are so severe that the inventor of the computer, Alan Turing, proposed it be the condition upon which computers were deemed to have "intelligence" [Turing, 1950]. Turing's original proposal is called an imitation game, where a human seated at a computer terminal has two conversations via the keyboard and screen. One of the conversational partners is a human, the other a machine. If the human seated at the terminal is unable to distinguish between them, the machine has passed the Turing Test. Notwithstanding the validity of determining intelligence based on behavior, this technique, or the Bystander variant, has been used in evaluating natural language generation [Jurafsky and Martin, 2000; Person *et al.*, 2002]. It has also fired the public's imagination in the annual Loebner Prize competition established to encourage the development of machines that could pass such a test.

However, many previous attempts at the Turing Test have dubious merit. Shieber [1994] in a critique of the Loebner competition concludes that the scope of the endeavor is too much for the current state of artificial intelligence. The weight of his argument stands on the methods used by the winning programs: the same kind of trickery first applied almost 40 years ago by ELIZA [Weizenbaum, 1966]. Programs such as ELIZA, PARRY [Colby, 1981], and the winner of the first Loebner competition [Weintraub, 1991] all select a conversational mode that masks deficiencies in the natural language interface. ELIZA imitates a Rogerian psychotherapist, which by definition is supposed to 'reflect' the patient/user's statements:

> PATIENT: I dislike you.
> ELIZA: Does it please you to dislike me?

PARRY likewise reflects the user's utterances, this time using the idiom of a paranoid schizophrenic:

> USER: I like to play football.
> PARRY: Why do you keep talking about football? You are really getting on my nerves.

Likewise, Weitraub's program used the idiom of "whimsical conversation". All of these programs use simple string matching to extract a phrase from the user's utterance and include that string in the program's utterance. The idioms used by the programs limit the user's expectations beyond a turn, and the idioms thus mask the simplicity of the algorithm. It's been reported that psychiatrists have believed PARRY's transcripts belong to a real paranoid schizophrenic, but the scientific merit of such an achievement is highly questionable [Shieber, 1994].

Although Shieber [1994]'s analysis reveals the enormous gap between winning the Loebner prize and possessing "real" intelligence, it is nonetheless significant that modern reimplementations of the pattern matching strategy using AIML [Wallace, 2005] are able to hold a realistic conversation on a number of topics. The ability of such a "chatterbot" to hold a realistic conversation can be useful in some contexts, e.g. video game entertainment, product marketing, and education.

Unfortunately, the effort involved in scaling up the pattern matching approach is enormous: freely available AIML implementations have upwards of 43,000 unique patterns, which together with the corresponding response of the chatterbot must be crafted by hand. Existing chatterbot packages [Foundation, 2005] mitigate this problem by allowing online users to interact with the chatterbot in a kind of reinforce-

Proceedings of the 5th *Workshop on Knowledge and Reasoning in Practical Dialogue Systems*, pages 15–21,
Hyderabad, January 8th, 2007.

ment learning, which is somewhat less effortful than scripting AIML by hand. Overall however, the AIML scripting approach suffers from the same weaknesses as any other knowledge engineering methodology: all of the knowledge must be engineered by hand, and the knowledge is usually domain dependent.

In contrast, this paper investigates domain independent generation of dialogue using techniques from shallow semantics, statistics, and syntax, all of which are either domain independent or use unsupervised learning. Even though the techniques are domain independent, some domain must be chosen for the purposes of evaluation. Therefore we have selected Philip K. Dick (PKD), a science fiction writer of the 20th century, as the dialogue persona to emulate. That is, the chatterbot must be able to respond both with the relevant information and with the relevant PKD style, just as if PKD himself had responded. The task is to utilize a collection of interviews, biographies, and novels written by PKD as a source for generating dialogue that has relevant response information and a stylistically appropriate answer. The best such dialogue generation techniques have been loaded into the PKD Android, a robotic portrait of Philip K. Dick.

2 Biographical Dialogue Generation

Depending on the unit size of generation, e.g. a paragraph, a sentence, a phrase, different problems arise. At the paragraph level, the generation task is essentially information retrieval (IR): given a context of dialogue (a query) find the best matching paragraph in the collection of interviews. This approach has the advantage that the paragraph returned is guaranteed to be a well formed PKD response and therefore to have his style. The disadvantage with this approach is that it is unlikely that an existing paragraph is totally appropriate, unless the context of the conversation closely matches that of a previous interview.

At the sentence level, the generation task is essentially IR with multiple document summarization: given a context of dialogue find the best matching documents, find the most appropriate summary sentences, and sequence them. The advantage of this approach is that the content and style of a response can be more closely tailored to the current context of the conversation. The disadvantage is that now we must consider the ordering of the sentences and the relationship between the sentences. For example, the cohesion between sentences may no longer match the PKD style, and anaphora between sentences need to be resolved.

Finally, at the phrase level, the generation task combines information retrieval, summarization, and parsing: retrieve the most relevant documents, find summary sentences, extract the phrases from these and recombine them into a syntactically and semantically well-formed response. Clearly the advantages are even more flexibility in the response, so that the chatterbot's reply may very closely match the context of the conversation. The disadvantage is that on top of the disadvantages inherited from the other two levels, the phrase approach must solve the problem of phrase order, e.g. matching verbs with appropriate semantic and syntactic arguments. Given the existing state of the natural language generation field, this is likely the point of diminishing returns at which the flexibility of the unit size is offset by the complexity of combining like-sized units together.

Our approach is distinguished from previous statistical approaches to natural language generation in that we use no underlying meaning representation. For example, Langkilde & Knight [1998a] and Langkilde & Knight [1998b], who pioneered the first statistical approach to NLG using bigrams, use an underlying abstract meaning representation (AMR) a kind of propositional structure, as the semantic specification of the desired generated text. This kind of semantic specification is one way to determine rough word order in a generated sentence, e.g. actor/actee roles. Our approach on the other hand, gets all of its semantics from latent semantic analysis (LSA) a statistical technique for modeling world knowledge. Not only can LSA be used for information retrieval [Dumais, 1993] but it also can be used to measure the cohesion between sentences and paragraphs [Foltz *et al.*, 1998] a useful property for sequencing sentences.

2.1 Paragraph: Information Retrieval

Latent semantic analysis [Landauer and Dumais, 1997; Dumais, 1993; Manning and Schutze, 1999] is a statistical technique that represents the similarity between collections of words as a cosine between vectors, and LSA can be used for information retrieval. The process begins with the collection of text into a single corpus. A matrix is created from the corpus, having one row for each unique word in the corpus and one column for each document or paragraph. The cells of the matrix consist of a simple count of the number of times word i appeared in document j. Since many words do not appear in any given document, the matrix is often sparse. Weightings are applied to the cells that take into account the frequency of word i in document j and the frequency of word i across all documents, such that distinctive words that appear infrequently are given the most weight.

The key to the process is singular value decomposition (SVD), a technique that creates an approximation of the original word by document matrix. After SVD, the original matrix is equal to the product of three matrices, word by singular value, singular value by singular value, and singular value by document. The size of each singular value corresponds to the amount of variance captured by a particular dimension of the matrix. Because the singular values are ordered in decreasing size, it is possible to remove the smaller dimensions and still account for most of the variance. The approximation to the original matrix is optimal, in the least squares sense, for any number of dimensions one would choose. In addition, the removal of smaller dimensions introduces linear dependencies between words that are distinct only in dimensions that account for the least variance. Consequently, two words that were distant in the original space can be near in the compressed space, supporting the inductive machine learning and knowledge acquisition reported in the literature[Landauer and Dumais, 1997]. The final step is to compare two collections of words by creating two vectors. Each word is associated with a row vector in the matrix, and the vector of a collection is simply the sum of all the row vectors of words in that collection. Vectors are compared geometrically using cosine.

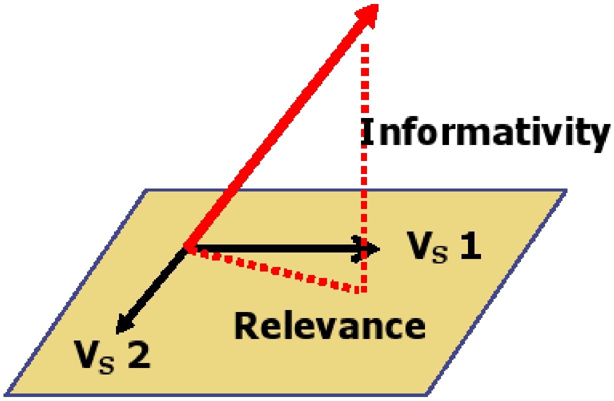


Figure 1: An orthonormal basis

For the PKD task, the interviewer's utterance is used as a query into a database containing conversations between interviewers and PKD. The N highest matching documents are checked to ensure that they are PKD responses. If one of the highest matching documents is an interviewer utterance, then the PKD response to that interviewer utterance is used. This method ensures that the PKD response is topically cohesive with the interviewer's utterance.

2.2 Sentence: Summarization

Cohesion can be measured by comparing the LSA cosines of two successive sentences or paragraphs [Foltz *et al.*, 1998]. However, cohesion is a crude metric: repetitions of a single sentence will be highly cohesive (cosine of 1) but conversationally inappropriate since no new information is introduced. A variation of the LSA algorithm using orthonormalized vectors provides two new measures, informativity and relevance, which can detect how much new information is added and how relevant it is in a context [Hu *et al.*, 2003]. The essential idea is to represent context by an orthonormalized span of vectors, one vector for each utterance. The span is a subspace of the higher dimensional LSA space, in the same way as a plane or line is a subspace of 3D space. The span is created by projecting each utterance onto the span of previous utterances using a method known as the Gram-Schmidt process [Anton, 2000]. Since each vector in the span is orthogonal, the span represents all linear combinations of what has been previously said. For example, in Figure 1, a new utterance creates a new vector that can be projected to the span, forming a triangle. The base of the triangle lies within the plane created by V_s1 and V_s2, the two previous vectors projected into the span. The leg of the triangle that lies within the span indicates the relevance of the recent utterance to the span; the perpendicular leg indicates new information. Accordingly, a repeated utterance would have complete relevance but zero new information. This discrimination is key in any attempt to use LSA to generate natural language.

The orthonormal basis can theoretically be used to create a summary of a group of paragraphs, although this technique has never been used before. Using LSA to select N documents, suppose that these N documents are segmented into M sentences. These M sentences can be used to form M orthonormal bases, each consisting of $M-1$ sentences. When the excluded sentence is projected into each of these orthonormal bases, two measures are returned, informativity and relevance. The excluded sentence with the lowest informativity, i.e. whose meaning was mostly captured by all the other sentences, is discarded, and the process begun again with $M-1$ sentences. The stopping criteria for such an algorithm can either be the number of sentences in the summary or a threshold for informativity.

To preserve the conversational stylistics of PKD, a set of N turns was summarized to K sentences. The sentence that appeared closest to the beginning of its respective turn was selected as the starter sentence. Stylistically, this technique addresses the issue of discourse markers and conversational openings. The number of sentences in the turn associated with the starter sentence (starter turn) was used as the number of target sentences to have in the generated turn, and the cohesion relationships between sentences in that turn was used as the target for the cohesion of sentences in the generated turn. For example, first the starter sentence is projected into the span. Second, each of the remaining $K-1$ sentences is projected into the span, and the sentence whose relevance/informativity most closely mirrors the relevance/informativity of the second sentence in the starter turn is selected. The second step repeats for each additional sentence to be generated. This technique ensured that the cohesion relationships between sentences in the generated turn matched as closely as possible with the cohesion relationships in the starter turn (and therefore with a natural PKD turn).

3 Method

3.1 Independent Variables

Two independent variables were manipulated. The first independent variable is the method of generating natural language, either continuing actual human dialogue, randomly, or using NLG techniques. The conditions of using actual dialogue and random dialogue provide control conditions and hypothetically represent the upper and lower bounds to which to compare the generation algorithm. The second independent variable is unit of generation (paragraph or sentence). Only the last utterance of the dialogue [Person *et al.*, 2002] is generated.

3.2 Dependent variables

Five dependent variables were measured. For each conversation judges were asked to respond to five questions. The first question is the Bystander Turing Test question, i.e. "Was the last utterance generated by a computer?" The other four questions directly correspond with the Gricean Conversational Maxims (Grice, 1975) of Quantity, Quality, Relevance, and Manner:

- The "PKD" answer is an appropriate length
- The "PKD" answer is accurate (i.e. contains no falsehoods)
- The "PKD" answer is relevant to the non PKD sentence preceding it

- The "PKD" answer is clear, unambiguous, and orderly

All ratings are on a 6-point scale:

- Definitely not
- Probably not
- Not sure but guess not
- Not sure but guess yes
- Probably yes
- Definitely yes

3.3 Judges

Two judges participated. Judge A had low knowledge of PKD. Judge B had an above average knowledge of PKD.

3.4 Materials

The methods for generating natural and random conversations are defined as follows. Approximately 8500 sentences from PKD interviews were extracted from their text sources and used to build a conversation database. Each natural conversation was created by randomly selecting an interviewer question in the database and then selecting PKD's response to that question. Random conversations were generated by randomly selecting a question by an interviewer and then randomly selecting a PKD response. In the case of randomly generated sentences, the first sentence was selected from the random PKD response. If the response originally had more than one sentence, then that number of additional sentences was selected from the database, taking care not to select the same sentence twice. Therefore the random sentence condition created PKD answers where the sentences were from various turns, and the number of sentences was consistent with the first turn selected.

The algorithm-generated conversations were created by creating an LSA space of 23 scanned books and the text of the conversation database. New software was written to implement the techniques mentioned above for LSA based information retrieval and orthonormal basis summarization for the respective text units of paragraph and sentence. Note that for LSA based IR, the second highest document was chosen, as opposed to the first (which would otherwise give the actual PKD response).

3.5 Procedure

Judge evaluation consisted of rating one packet containing 60 interviewer/PKD pairs. The method variable has three levels, natural, random, and generated, the unit variable has two levels, paragraph and sentence, and each condition had ten exemplars each, 3 x 2 x 10 = 60. The pairs were randomly ordered. The judges were asked to rate the conversation on 5 six-point scales, indicating the Bystander Turing Test and the four Gricean Maxims. The following is an example dialogue turn used in the experiment:

> INTERVIEWER: did he take you for real oh my god he really thought you rode horses huh
>
> DICK: oh well christ i have no idea what he thought

Measure	Natural	Random	Generated	Total
The "PKD" answer was generated by a computer				
Paragraph	2.45	3.05	3.40	8.90
Sentence	3.00	4.35	3.65	11.00
Total	5.45	7.40	7.05	
The "PKD" answer is an appropriate length				
Paragraph	4.30	3.55	4.00	11.85
Sentence	4.30	4.00	3.85	12.15
Total	8.60	7.55	7.85	
The "PKD" answer is accurate				
Paragraph	4.65	3.50	4.25	12.40
Sentence	4.05	3.45	3.90	11.40
Total	8.70	6.95	8.15	
The "PKD" answer is relevant				
Paragraph	4.65	2.50	3.80	10.95
Sentence	3.95	2.50	3.15	9.60
Total	8.60	5.00	6.95	
The "PKD" answer is clear				
Paragraph	4.95	3.85	4.40	13.20
Sentence	3.95	2.95	3.45	10.35
Total	8.90	6.00	7.85	

Figure 2: Means for dependent measures

4 Results

Five 3 x 2 repeated measures ANOVAs were performed to determine the effect of Generation Method (Natural vs. Random vs. NLG) and Unit Size (Paragraph vs. Sentence) on the subject's ratings of the five questions. The means for the four measures are reported in Figure 2.

4.1 Question 1: The "PKD" answer was generated by a computer

The 3 x 2 ANOVA for the "PKD answer was generated by a computer" question indicated no interaction between the method of generation and the unit size. However, the method of generation was significant ($F = 5.78$, $p = .01$) and the unit size was also significant ($F=8.94$, $p = .02$). Pairwise comparison of the methods revealed that the natural method was significantly different from the random condition ($p = .002$) and from the generated condition ($p = .04$), but there is no statistical difference on this question between the random and generated conditions. Analysis of the associated means shows that the natural responses were perceived as more humanlike than the other two. Pairwise comparison of the units revealed that the paragraph unit is significantly different from the sentence unit ($p = .02$). Analysis of the associated means shows that the paragraphs were perceived as more humanlike than sentences. The difference between the means for random paragraphs and random sentences almost leads to a significant interaction ($F = 2.82$, $p = .09$). Interestingly, the difference between the means for natural paragraph and natural sentence is greater than the difference between generated paragraph and generated sentence, indicating that the orthonormal basis method of generating a response is doing well compared to the IR approach on this measure.

4.2 Question 2: The "PKD" answer is an appropriate length

The 3 x 2 ANOVA for the "PKD answer is an appropriate length" question indicated no significant interaction between the method of generation and the unit size. There was no significant difference between methods of generation and no significant difference between unit sizes.

4.3 Question 3: The "PKD" answer is accurate

The 3 x 2 ANOVA for the "PKD answer is accurate" question indicated no interaction between the method of generation and the unit size. However, the method of generation was significant ($F = 3.64$, $p = .05$) and the unit size was also significant ($F = 6.43$, $p = .03$). Pairwise comparison of the methods revealed that the natural method was significantly different from the random condition ($p = .03$), but there is no statistical difference on this question between the generated condition and the other two conditions. Analysis of the associated means shows that the natural responses were perceived as more accurate than random responses, and the nonsignificant trend is that natural responses are more accurate than generated which are more accurate than random. Pairwise comparison of the units revealed that the paragraph unit is significantly different from the sentence unit ($p = .03$). Analysis of the associated means shows that the paragraphs were perceived as more accurate than sentences. The difference between the means for random paragraphs and random sentences almost leads to a significant interaction ($F = 2.82$, $p = .09$). Once again, the difference between the means for natural paragraph and natural sentence is greater than the difference between generated paragraph and generated sentence, indicating that the orthonormal basis method of generating a response is doing well compared to the IR approach on this measure.

4.4 Question 4: The "PKD" answer is relevant

The 3 x 2 ANOVA for the "PKD answer is relevant" question indicated no interaction between the method of generation and the unit size. However, the method of generation was significant ($F = 14.07$, $p = .0001$) and the unit size was also significant ($F = 5.46$, $p = .04$). Pairwise comparison of the methods revealed that the random method was significantly different from the natural condition ($p = .0001$) and from the generated condition ($p = .02$), but there is no statistical difference on this question between the natural and generated conditions. Analysis of the associated means shows that the natural and generated responses were perceived as more relevant than random responses. Pairwise comparison of the units revealed that the paragraph unit is significantly different from the sentence unit ($p = .04$). Analysis of the associated means shows that the paragraphs were perceived as more relevant than sentences. Again, the difference between the means for natural paragraph and natural sentence is smaller than the difference between generated paragraph and generated sentence, indicating that the orthonormal basis method of generating a response is doing well compared to the IR approach on this measure.

4.5 Question 5: The "PKD" answer is clear

The 3 x 2 ANOVA for the "PKD answer is clear" question indicated no interaction between the method of generation and the unit size. The method of generation was not significant ($F = 3.02$, $p = .07$) and the unit size was significant ($F = 21.49$, $p = .001$). Analysis of the means of method shows the nonsignificant trend of natural as more clear than generated, which is more clear than random. Pairwise comparison of the units revealed that the paragraph unit is significantly different from the sentence unit ($p = .001$). Analysis of the associated means shows that the paragraphs were perceived as more accurate than sentences.

5 Discussion

With the exception of the first question, "The PKD answer was generated by a computer," all answers to the questions show a trend in which the generated answer is between random and natural. This trend meets our expectations that the generation techniques will provide a useful approximation of the natural answer. Moreover, on all questions except the first, there is no statistically significant difference between the natural method and the generated method. This is an extremely positive result. It means that for all of Grice's conversational maxims, there is no perceptible difference between the natural and generated methods. Therefore these results show that the generated method performs as an appropriate conversational partner for one turn. If it can be determined what is cuing the judges on the first question to rate the generated responses as more computerlike, the system should be able to generate responses that are not significantly different from the natural response on any of the five measures.

With the exception of the second question, "The PKD response was of an appropriate length" which was not significant, the effect of unit size was significant on all questions. The direction of the significant difference indicates that paragraphs are considered more humanlike and more conversationally appropriate than sentences. This result suggests that a backoff strategy is appropriate for this task. In such a backoff strategy, the paragraph method would be used preferentially before the sentence method. The exact decision criteria for this strategy has been determined in post hoc tests, in which a regression model was fit to metrics such as cohesion, informativity, and relevance. It is also interesting to note that the difference between the means for sentence and paragraph for the generated method on questions 1, 3, and 4, was less than the same difference for the natural method. This indicates that the summarization method using an orthonormal basis is doing fairly well, and suggests that some of the significant differences between sentence and paragraph are a result of the greatly differing means in the random sentence and random paragraph conditions.

Overall, these results show that shallow, domain-independent, and unsupervised learning techniques can be used to create a conversationally appropriate dialogue between a computer and a person, i.e. one that follows the

Figure 3: A robotic portrait of PKD

Gricean Maxims. This is particularly interesting because it shows that knowledge engineering techniques so common in natural language generation research are not necessary to make a conversationally appropriate dialogue. The issue of passing the Bystander Turing Test, however, still needs to be addressed.

These results were incorporated into a robotic portrait of PKD [Hanson *et al.*, 2005] shown in Figure 3. Although a full description of the PKD Android is beyond the scope of this paper, the following gives a brief overview of its implemented dialogue generation system. Because of processing constraints, the final dialogue generation system used the paragraph generation method described in this paper together with a chatterbot for stock questions, e.g. “What is your name”. The paragraph method generated 20 responses and the chatterbot generated 1 response to each user input. Responses were then ranked as to their probability of being a good response using the metrics of cohesion, relevance, and informativity described above. If the best response had a low probability of being an end of turn response, based on the probability of its last word being the end of a turn, the next best response was appended onto the end, iteratively. This final answer was then spoken by the PKD Android, with corresponding lip synchronozation and facial expression. An example conversation follows:

> PKD Android: Well the I-Ching gives advice beyond the particular advice that trancends the immediate situation. The answers have a universal quality. For instance, the mighty are humbled, and the humbled are raised. If you use the I-Ching long enough and continually enough it will begin to change and shape you as a person. It will make you into a Taoist, whether or not you have ever heard the word, whether or not you want to be. Well...
>
> Human: Don’t you think Taoism is too easy a religion for a robot? To illogical?
>
> PKD Android: I think the powers actually exist. I think they are real. Too: are you talking about an animal, vegetable, or mineral?
>
> Human: Taoism doesn’t have to do so much with animals, vegetables, or minerals, but ideas that you can’t prove. How can a robot think about something that it can’t prove?
>
> PKD Android: What makes it impossible?
>
> Human: What makes it impossible to prove Taoism? It’s too abstract.
>
> PKD Android: Ask the Open Directory about it. It being what.
>
> Human: It being Taoism...

6 Conclusion

This paper examined the effectiveness of several domain independent methods of generating dialogue, and introduced the use of the orthonormal basis for summarization. By mining a large collection of text written by and about Philip K Dick, LSA and the orthonormal basis for summarization were used to generate PKD-style answers to randomly selected interviewer questions. Judges read conversations between PKD and an interviewer, where the last utterance was either a continuation of the original dialogue, randomly generated, or generated using LSA or the orthonormal basis for summarization. Analysis of the judges ratings showed that although the techniques introduced do not allow a computer to pass the Bystander Turing Test, they do create a conversational partner that is indistinguishable from PKD on all of Grice’s conversational maxims. This result indicates that natural language generation methods do not have to include knowledge engineering and that domain independent and unsupervised techniques can be used to create an effective conversational partner.

References

[Anton, 2000] H. Anton. *Elementary linear algebra*. John Wiley, New York, 2000.

[Colby, 1981] K. M. Colby. Modeling a paranoid mind. *Behavioral and Brain Sciences*, 4(4):515–560, 1981.

[Dumais, 1993] S. Dumais. LSI meets TREC: A status report. In *Proceedings of the First Text Retrieval Conference*, pages 137–152. NIST, 1993.

[Foltz *et al.*, 1998] P. W. Foltz, W. Kintsch, and T. K. Landauer. The measurement of textual coherence with latent semantic analysis. *Discourse Processes*, 25:285–307, 1998.

[Foundation, 2005] ALICE Artificial Intelligence Foundation, 2005.

[Hanson *et al.*, 2005] D. Hanson, A. Olney, S. Prilliman, E. Mathews, M. Zielke, D. Hammons, R. Fernandez, and H. Stephanou. Upending the uncanny valley. In *Proceedings of the Twentieth National Conference on Artificial Intelligence and the Seventeenth Annual Conference on Innovative Applications of Artificial Intelligence*, pages 1728–1729, Menlo Park, CA, 2005. AAAI Press.

[Hu *et al.*, 2003] X. Hu, Z. Cai, M. Louwerse, A. Olney, P. Penumatsa, and A. Graesser. An improved LSA algorithm to evaluate contributions in student dialogue. In *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*, 2003.

[Jurafsky and Martin, 2000] D. Jurafsky and J. H. Martin. *Speech and Language Processing*. Prentice Hall, Upper Saddle River, 2000.

[Landauer and Dumais, 1997] T. Landauer and S. Dumais. A solution to Plato's problem: the latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, 104:211–240, 1997.

[Langkilde and Knight, 1998a] Irene Langkilde and Kevin Knight. Generation that exploits corpus-based statistical knowledge. In *COLING-ACL*, pages 704–710, 1998.

[Langkilde and Knight, 1998b] Irene Langkilde and Kevin Knight. The practical value of N-grams in derivation. In Eduard Hovy, editor, *Proceedings of the Ninth International Workshop on Natural Language Generation*, pages 248–255, New Brunswick, New Jersey, 1998. Association for Computational Linguistics.

[Manning and Schutze, 1999] C. Manning and H. Schutze. *Foundations of statistical natural language processing*. MIT Press, Cambridge, 1999.

[Person *et al.*, 2002] N. K. Person, M. Erkel, and A. C. Graesser. Autotutor passes the bystander Turing test. In *Proceedings of E-Learn*, 2002.

[Shieber, 1994] S. M. Shieber. Lessons from a restricted Turing test. *Communications of the Association for Computing Machinery*, 37(6):70–78, 1994.

[Turing, 1950] A. M. Turing. Computing machinery and intelligence. *Mind*, 49:433–460, 1950.

[Wallace, 2005] R. Wallace. http://www.alicebot.org /TR /2005 /WD-aiml/, 2005.

[Weintraub, 1991] J. Weintraub. http://www.loebner.net /Prizef /weintraub-bio.html, 1991.

[Weizenbaum, 1966] J. Weizenbaum. ELIZA - a computer program for the study of natural language communication between man and machine. *Communications of the Association for Computing Machinery*, 9:36–45, 1966.

First Steps towards Dialogue Modelling from an Un-annotated Human-Human Corpus

Sudeep Gandhe and **David Traum**
Institute for Creative Technologies
University of Southern California
13274 Fiji Way, Suite 600, Marin Del Ray, CA 90292
gandhe@ict.usc.edu, traum@ict.usc.edu

Abstract

Virtual human characters equipped with natural language dialogue capability have proved useful in many fields like simulation training and interactive games. Generally behind such dialogue managers lies a complex knowledge-rich rule-based system. Building such system involves meticulous annotation of data and hand autoring of rules. In this paper we build a statistical dialogue model from roleplay and wizard of oz dialog corpus with virtually no annotation. We compare these methods with the traditional approaches. We have evaluated these systems for perceived appropriateness of response and the results are presented here.

1 Introduction

Virtual human characters equipped with natural language dialogue capability have proved useful in many fields like simulation, training and interactive games. These dialogue capabilties are the essential part of their human-like persona. This interface has to be good enough to engage the trainee or the gamer in the activity.

Natural language dialogue systems come in many different flavors. Chatterbot systems like Eliza [Weizenbaum, 1966] or Alice [Wallace, 2003] have to operate in an unrestricted domain with an aim of being human-like. The user input can be about any topic he/she can think of. On the other hand, task-oriented dialogue systems such as pizza-ordering, ATIS [Seneff *et al.*, 1991] or Trains [Allen, 1995] restrict the user quite severely in the topics and ways of talking about them that are allowed.

In casual conversation, even without specific domain knowledege, one can always find reasonable things to say, e.g., "I don't want to talk about that", or "Why do you say that?". Moreover, it is often sufficient to talk about topics at a fairly shallow level, without requiring a lot of detailed task knowledge or knowledge of how some parts of a task relate to others. On the other hand, for a task oriented dialogue in which the system is expected to perform a task or provide task-relevant information, a detailed understanding of the progression of the task and which information has been expressed is often crucial. There are some domains that fall between these extremes, for instance negotiation about whether or not to adopt a proposal. In this case, there is definitely a task or set of tasks involved, but one does not necessarily require as detailed knowledge as is required to actually perform the task. One could agree or disagree for partial or even hidden reasons. This can allow much more flexibility in the type of dialogue interaction, including more varied levels of initiative and dialogue moves, as well as more general arguments and assessments.

There are also various methods for dialogue management. Chatbots typically follow Eliza in operating at a textual level, with pattern matching and substitution to compute a response from an initiative. This can provide a degree of generality, as a single pattern may produce a large range of responses to different initiatives. On the other hand, they can be fairly brittle if the pattern is not appropriately constrained and match inappropriately, producing sometimes uncomprehensible results. Corpus-based retrieval approaches (e.g., [Chu-Carroll and Carpenter, 1999; Leuski *et al.*, 2006]) have an advantage of robust selection, with a more limited set of responses.

Task oriented dialogue generally operates at a concept or dialogue act level. This allows reasoning at more of a meaning than form level and easy integration with other kinds of knowledge-based reasoning, but also more kinds of processing to translate from the surface level to the meaning level and back again.

All of these methods require either extensive writing of rules or other symbolic processing methods, or extensive corpus annotations, both of which serve to introduce a high cost in the development of a dialogue system for a new domain.

In this work we take a look at unsupervised corpus based methods to bootstrap dialogue bots. They don't have sophasticated cognitive models, but they can be built instantly from a dialogue corpus without annotation or rule-writing. We compare these methods with the more traditional approach of building a information-state based dialogue system.

In the next section we will introduce our first case study system for an annotation-less virtual human dialogue manager. In the next section we will elaborate more on the motivtion for using corpus based methods for such systems. In section 4 we describe the chat-bot systems we have implemented. Section 5 presents the evaluation of the implemented systems and we conclude with discussion and future work.

Proceedings of the 5th *Workshop on Knowledge and Reasoning in Practical Dialogue Systems*, pages 22–27,
Hyderabad, January 8th, 2007.

2 SASO-ST

At Institute for Creative Technologies, USC researchers have developed prototype virtual human characters used for simulation training. SASO-ST [Traum *et al.*, 2005] is one such environment, involving a prototype of a training environment for learning about negotiating with people from different cultures and with different beliefs and goals. In the first scenario, the trainee acts as an army Captain negotiating with a simulated doctor. The goal is convince him to move his clinic to another location. The captain can offer help in moving the clinic and some other perks like medical supplies and equipments.

In order to investigate this domain, and build resources for the system, we collected a corpus of roleplay dialogues and Wizard of Oz (WoZ) dialogues. Roleplay dialogues feature more free-form human face to face interaction whereas the WoZ interactions are constrained by allowing the wizard playing the role of doctor to choose from a limited set of replies. Fig 1 shows a typical roleplay dialogue.

doctor		
	0.0	yes what is it
	1.063	i've got a lot of patients in the back .
	3.03	what can i do for you .
captain		
	4.217	how are you doing sir ,
	5.175	uh my name's captain (xx) ,
	6.748	how are you today ?
doctor		
	7.78	uh well ,
	8.905	
	9.623	i could be better ,
	10.44	i've got a lot of patients in the back ,
	12.061	uh we just had uh FIVE of them come in from the LAST bombing ?
	15.718	so ,
	16.311	what can i do for you .
captain		
	17.342	okay i know you're very busy so i'll get straight to what i came here to talk to you about .
	22.983	right now ,
	24.185	with our estimate ,
	25.077	this is a very unsecure area .
	26.827	and what we'd like to do sir is uh secure and stabilize your patients as soon as possible and move you out of this area so we can move you to a more secure location .
doctor		
	36.58	my PATIENTS are stable right NOW .
	40.489	and ,
	41.395	i i don't understand why you're coming in here ,
	44.926	to tell me to move patients out of here ,
	47.583	from a clinic that's been here for almost a YEAR .
	50.311	and now i have to move my patients ?

Figure 1: A sample roleplay dialogue in SASO-ST

3 Motivation

A typical lifecycle of the dialogue modelling process for virtual humans begins with defining the domain of interaction which follows from the story line. The process includes defining the beliefs and goals of all the parties involved. It is followed by conducting roleplays where volunteers carry out conversations with these goals in mind. This gives a better idea about the behavior of participants that would be expected in real simulation. Experts can then formalize the task structure based on these sample interactions. Additional speech and language data can be gatherd by carring out Wizard of Oz studies and transcribing it. This gathered data can be used for training speech recognition acoustic and language models.

In an information-state based [Traum and Larsson, 2003] approach as used in SASO-ST, the dialogue model has to maintain the information-state — a description of the current state of information that is important for participating in the dialogue. This is done by applying a set of update-rules which are used to change the information-state based on the new input as the dialogue proceeds. Generally the input to information-state is a set of dialogue acts and semantic interpretation about an utterance.

In order to use corpus dialogue data for this kind of system, one must either write parsing or translation rules, or annotate sufficient quantities to train statistical systems. Fig 2 shows an example of the semantic annotation for an utterance in the SASO-ST system. It includes information like speech-acts, modality and case-roles. Based on pairs of sentences with annotated reporesentations like this, a Natural Languge Understanding module can be trained in a supervised fashion which maps the utterance to its semantic meaning. Rule-based processing is then used by the dialogue manager to compute resulting information state components and system utterances.

Producing training data for speech recognition langauge models makes it worthwhile to collect roleplay/WoZ data. But to make further use of this data, significant human effort is required either to write rules or annotate data. Alleviating this human-effort requirement is the main motivation behind the idea of using corpus-based methods to bootstrap dialogue systems without any annotation required. The shallow task structure and the constrained scenario of the negotiation domain make it viable to model dialogue as a sequence of tokens, a language. These modelling techniques are inspired from Information Retrieval field and try to predict the next utterance given the context of the dialogue. They work at the lexical level which does not need the dialogue act or semantic annotaion.

4 Chat-Bot methods

The methods described in the this section view dialogue as a sequence of tokens. They employ simple Information Retrieval techniques to create chat-bots that are trained in an unsupervised manner. Since there is no annotation effort other than building the dialogue corpus from roleplays and WoZ, these methods allow for rapid prototype development.

We will have to move the hospital .	
S.mood	declarative
S.sem.task	move-clinic
S.sem.speechact.type	statement
S.sem.type	event
S.sem.modal.deontic	must
S.sem.agent	we
S.sem.event	move
S.sem.theme	hospital
S.sem.time	future

Figure 2: An example of semantic annotation

In building these prototypes we have chosen to fix the input modality to typed text and the interface is in the form of a chat session. The turns strictly alternate between the doctor (system) and the captain (user). The screenshot of the interface is as seen in the fig 3.

The general idea is to retrieve one of the doctor utterances from the corpus and present it to the user as the system response. We implemented 4 types of chat-bots. They capture different aspects of local and global coherence of the dialogue.

4.1 random bot

This type of bot provides a zero baseline and does not capture global or local coherence. A set of utterances with doctor as the speaker is compiled from the corpus. The bot just replies to any utterance of the captain with a randomly selected utterance from this list. There are around 435 doctor utterances to randomly choose from.

4.2 nearest context

This type of bot captures local coherence. In this type rather than choosing the reply randomly from all available doctor utterances we decide to choose the one which has the most similar context as compared to the context of the current ongoing dialogue. The context is defined as last *n* turns. Here we have chosen n=2. To find the similarity between the contexts we represent the context using vector space model as in information retrieval [Manning and Schutze, 1999]. Fig 5 shows an example of the feature vector used to represent the context of the dialogue. In this vector the unigrams from utterances form the features. These unigrams are augmented with the speaker and the distance in time in units of turns. The latest turn is at a distance of 0, the previous at 1 and so on. The weights for the features are derived from tf-idf scores. For the systems to be more reactive to the latest input, we weigh these tf-idf scores depending on how far back in the history the utterance is.

Let W_i^j be the weight assigned for unigram w_i which appears j turns ago. Then W_i^j is given by,

$$W_i^j = TF(w_i) \times IDF(w_i) \times H(j) \quad (1)$$

$$TF(w_i) = 1 + \log(\#w_i) \quad (2a)$$

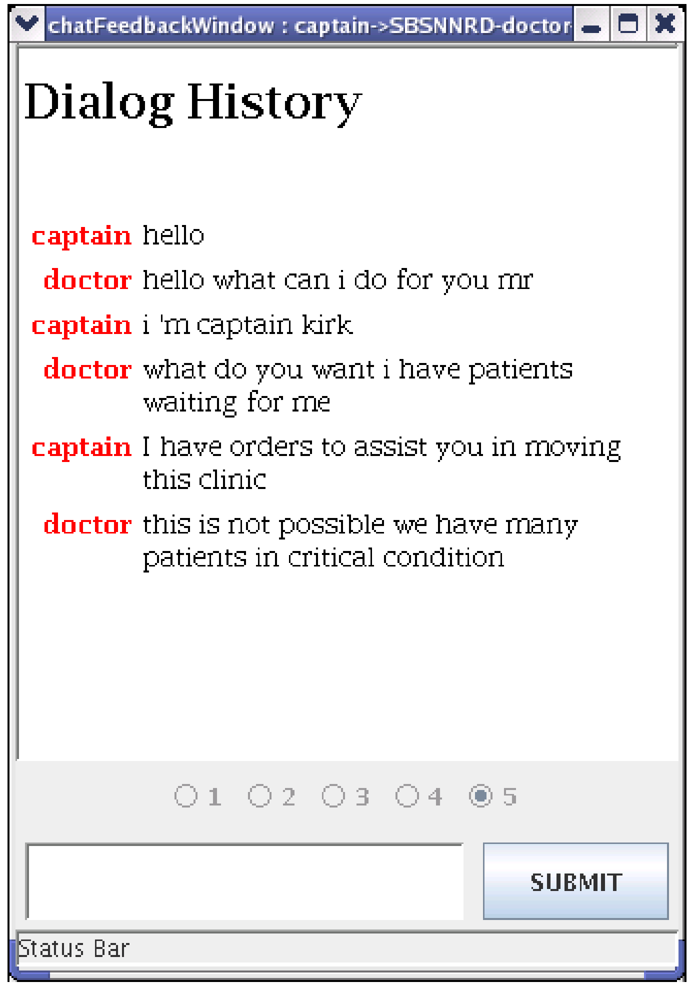


Figure 3: A screenshot of user interface

where $\#w_i$ is the number of times w_i appears in the utterance

$$IDF(w_i) = \log\left(\frac{N}{df_i}\right) \quad (2b)$$

where N is the total number of utterances
and df_i is the number of utterances containing w_i

$$H(j) = \exp\frac{-j^2}{2} \quad (2c)$$

This is a type of memory based or Instance based learning. The training phase only involves identifying all the contexts associated with utterances and storing the vector space representations for them in memory. When it's time to predict the next utterance for the doctor the job is to find a context c_k which is most similar to the context of the current dialogue c. The utterance u_k associated with context c_k will be the reply. Here k is given by,

$$argmin_{i=1..n}\left(||\bar{c}_i - \bar{c}||\right) \quad (3)$$

where the feature vectors $\bar{c}_i$ and $\bar{c}$ are L_2 normalized.

captain	hello
doctor	so how are you going to do that
captain	hello doctor i am captain kirk
doctor	uh i have i have patients from both sides from ah there have been injured from american forces as well as other locals

Figure 4: example interaction for random bot

	⋮
doctor	what do you want i have patients waiting for me
captain	I have orders to assist you in moving this clinic

<captain>_0_i	1.08705147
<captain>_0_have	1.66113929
<captain>_0_orders	6.24610677
<captain>_0_to	1.03117101
<captain>_0_assist	4.16666522
<captain>_0_you	0.56935296
<captain>_0_in	2.18566375
<captain>_0_moving	2.98801023
<captain>_0_this	1.7327793
<captain>_0_clinic	2.41746537
<doctor>_1_what	1.21161264
<doctor>_1_do	1.18616006
<doctor>_1_you	0.34533003
<doctor>_1_want	1.86086817
<doctor>_1_i	0.65933004
<doctor>_1_have	1.00753191
<doctor>_1_patients	1.02003814
<doctor>_1_waiting	2.45577118
<doctor>_1_for	1.32567208
<doctor>_1_me	1.66771622

Figure 5: feature vector representing the context of previous *n* turns. *(here n=2)* (same example as in fig 3)

Apart from this we try to avoid repetition in the dialogue. If the predicted best response u_k is same as that predicted in previous prediction the next best reply is chosen.

4.3 segmented - nearest context

This type of bot tries to capture both global and local coherence. The chat bot described earlier suffers from short-sightedness. It is designed to be coherent locally, but in the broader sense of the dialogue phenomenon there are certain responses that don't make much sense. These are a direct result of the fact that context cannot be faithfully represented using just n previous turns.

We observed most of the failures are because of the violations of presuppositions. Presupposition is the assumption that must be held by both parties for an utterance to make sense. e.g. Doctor saying " I don't want to move the clinic " makes sense only if the the dialogue participants both believe that doctor may be required to move the clinic. And in this case it comes down to whether moving the clinic was previously mentioned or not.

captain	hello doctor
doctor	hello
captain	i am captain kirk nice to meet you
doctor	i'm uh doctor perez uh i'm very busy right now i i only have uh just a few minutes to uh
captain	i want to talk to you about an important issue
doctor	uh captain i i appreciate you have given us a lot of very good information to uh which roads are safe and where the landmines are and i need you+ uh i i cannot move this location though because of all these patients they're they're too critical right now i'm working on a on a on a young girl with amoebic dysentery and and she she requires my attention at all times i there is no way i i+ these people are in no shape to to+

Figure 6: example interaction for nearest context bot

Hence we split the dialogue in segments that try to keep track of whether several key concepts were introduced in the dialogue or not. These key concepts are picked by an expert who understands the domain well. Fig 7 illustrates the compiled list for SASO-ST dialogues. It identifies concepts by listing down the words with similar sense.

[move]	move relocate
[supplies]	antibiotics supplies medicines plasma
[transportation]	vehicle transportation trucks

Figure 7: list of key concepts along with the representative unigrams compiled for SASO-ST dialogues

With this information the system can determine the segment signature for every context. e.g. the segment signature for the context in fig 5 will be,

```
[move 1, supplies -1, transportation -1]
```

The decision procedure for picking up the response based on given context is same as earlier but with one additional constraint. The most similar context c_i must have the same segment signature as the current context c.

Notice that it is possible to misrecognize the current segment of the dialogue. This may happen because a certain similar word was used for signaling the concept but this word was not in the list. e.g. using " shifting the clinic " instead of

captain	hello doctor
doctor	hello
captain	i am captain kirk nice to meet you
doctor	i'm uh doctor perez uh i'm very busy right now i i only have uh just a few minutes to uh
captain	i want to talk to you about an important issue
doctor	ok

Figure 8: example interaction for segmented - nearest context bot

captain	hello
doctor	yes
captain	i am captain kirk nice to meet you
doctor	you are the threat i need protection from you
captain	no we are here to protect you and your clinic
doctor	are you injured

Figure 9: example interaction for segmented - random bot

" moving the clinic ". Similarly " moving the chair " instead of " moving the clinic ". This clearly shows that single unigram spotting is not adequate for detecting dialogue segment transitions.

4.4 segmented - random

This type of bot tries to capture only the global coherence. The last mentioned chat-bot system tries to achieve local appropriateness by finding the nearest similar context and also the global appropriateness by keeping track of the segments. To understand which of the two factors makes more significant impact we implemented the fourth type of bot. It keeps track of the segment signature of the context but picks up one of the utterance randomly with that signature.

5 Evaluation

To evaluate the merits of these methods we asked volunteers to conduct a conversation with the simulated doctor. These volunteers had two roles - as a participant in negotiation conversation and also as a judge of the responses from the doctor. The interface shown in fig 3 allows the volunteers to judge the doctor's response on a scale of 1 to 5 for appropriateness. Here 1 stands for a totally non-sensical reply and 5 is the perfectly appropriate response. This is a subjective metric and we believe that the conversation participant is in the best position to judge the appropriateness of the response.

Each bot type was used in 5 conversations. Each volunteer had conversations with all types of bots. The presenting order of the bots was balanced.

The average ratings for various types of chat-bots is summarized here. nearest context, segmented - nearesest context and segmented - random are all significantly better (t-test, $p < 0.05$) over the random baseline. segmented - nearest context is significantly better (t-test, $p < 0.05$) than segmented - random or nearest context approaches.

	Without Segments		With Segments	
Without Context	avg	2.6764	avg	3.0430
	stddev	1.2758	stddev	1.2930
	size	136	size	93
With Context	avg	3.0625	avg	3.4722
	stddev	1.5438	stddev	1.3703
	size	112	size	108

Figure 10: Results for various types of chat-bots

6 Discussion and Future work

In the preliminary evaluation and the subjective feedback from the users it appears that type-3 system performs suprisingly well. The reason behind this success stems from the fact that these converstaions are restricted because they have to follow the story line. Also since the task structure is shallow, just identifying the correct dialogue segment helps quite a bit.

The way these methods differ from information-state based dialogue systems is that there is no need for annotation. Theoretically information state based systems can achieve arbitrary levels of perfection given enough rules are authored for it. But rule authoring still remains a practical limiting factor for such systems. We do miss the deep understanding that an information-based system gives, such as principled connection to emotions and plan reasoning.

When compared to general purpose chat-bot systems like Eliza or Alice, our domain of interaction is well defined. e.g. chat-bot systems have to be ready to talk about a variety of topics, including favorite movies, sports etc. while our system just has to know about the negotiation scenario between the captain and the doctor.

Some recent task-oriented dialogue systems use reward structure for optimizing the dialogue policies. These systems generally have a clear defination of success and failure which helps in designing reward structure. They typically learn the optimal policies using MDP [Levin and Pieraccini, 1997], POMDP [Roy *et al.*, 2000] framework. But in case of games or simulation training dialogue systems don't have easy access to the notion of success and failure. It is very hard to design this reward system. The best that can be done is to imitate human behavior, which is exactly what our system is designed to do.

There are a number of avenues of future work that we would like to employ. First, we can compare our dialogue agents to other dialogues, e.g., some of the original role-play dialogues and dialogues with the SASO-ST system. We would need to alter our evaluation mechanism though, since our evaluations are done by the dialogue participants themselves in real time rather than by a third party.

We will also connect our dialogue manager to the virtual human's body and speech recognizer and synthesizer, to see whether embodiment has an effect on acceptability. We will also use the same techniques to create systems for other dialogue agents.

Also, we will try to automatically identify the key concepts used to segment the dialogue, as well as looking at what in-

Speaker	Utterance	Rating
doctor	uh i'm sorry what was your name	5
captain	i'm captain kirk	
doctor	captain it's nice to meet you i don't have much time so i would appreciate it if you could make uh make this quick	4
captain	ok doctor . I want to talk about the security of the area this hospital is located in.	
doctor	well they i don't know that there is very good uh anwhere this this one seems to be full of patients who i need to be treating but the problem is the roads are not always clear and it's necessary uh you know that we are able to get supplies and and we're not	2
captain	yes doctor, it is hard to get supplies here, because of the danger. We do have access to supplies, but it will be hard to bring them here.	

Figure 11: Illustration of the problem due to the granularity of the utterance. This dialogue is generated by using segmented-nearest context bot and shows the evaluation by the participant. Last utterance from doctor gets a low rating.

formation would both improve dialogue quality and be able to be extracted automatically or authored with little effort. We will also investigate how these methods can be applied to tasks which have a more deeper structure.

Our system works by selecting the appropriate utterance from the ones it has seen. Using human generated utterances has the advantage of being more natural and fluent. But the main assumption that a dialogue can be carried out by retreiving an utterance from the training data rather than constructing it from a high level abstract representation can be a considerable limitation. This is felt strongly when the system comes across completely unseen contexts. Typically we found that type-3 system gets stuck in a loop where the dialogue does not move to the next phase.

Also the granularity of the utterances is an issue. Fig 11 shows an example where the utterance refers to security issues by mentioning the problems of the blocked roads but also talks about other things not mentioned in the preceeding context. This makes the utterance less coherent. We are looking into stochastic models for discourse coherence [Barzilay and Lapata, 2005; Soricut and Marcu, 2006] which can help recognize which utterances are best suited given the context.

References

[Allen, 1995] James F. Allen. The trains project. *Journal of Experimental and Theoretical AI*, 1995.

[Barzilay and Lapata, 2005] Regina Barzilay and Mirella Lapata. Modeling local coherence: An entity-based approach. In *Proc. ACL-05*, 2005.

[Chu-Carroll and Carpenter, 1999] Jennifer Chu-Carroll and Bob Carpenter. Vector-based natural language call routing. *Journal of Computational Linguistics*, 25(30):361–388, 1999.

[Leuski *et al.*, 2006] Anton Leuski, Ronakkumar Patel, David Traum, and Brandon Kennedy. Building effective question answering characters. In *Proceedings of the 7th SIGdial Workshop on Discourse and Dialogue*, 2006.

[Levin and Pieraccini, 1997] Esther Levin and Roberto Pieraccini. A stochastic model of computer-human interaction for learning dialogue strategies. In *Proc. Eurospeech '97*, pages 1883–1886, Rhodes, Greece, 1997.

[Manning and Schutze, 1999] Chris Manning and Hinrich Schutze. *Foundations of Statical Natural Language Processing*, chapter 15. MIT Press. Cambridge, MA, 1999.

[Roy *et al.*, 2000] N. Roy, J. Pineau, and S. Thrun. Spoken dialogue management using probabilistic reasoning. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL2000)*, Hong Kong, 2000., 2000.

[Seneff *et al.*, 1991] E. Seneff, L. Hirschman, and V.W. Zue. Interactive problem solving and dialogue in the atis domain. pages 354–359, February 1991.

[Soricut and Marcu, 2006] Radu Soricut and Daniel Marcu. Discourse generation using utility-trained coherence models. In *Proc. ACL-06*, 2006.

[Traum and Larsson, 2003] David Traum and Staffan Larsson. The information state approach to dialogue management. In Jan van Kuppevelt and Ronnie Smith, editors, *Current and New Directions in Discourse and Dialogue*. Kluwer, 2003.

[Traum *et al.*, 2005] David Traum, William Swartout, Jonathan Gratch, and Stacy Marsella. Virtual humans for non-team interaction training. AAMAS-05 Workshop on Creating Bonds with Humanoids, July 2005.

[Wallace, 2003] Richard Wallace. *Be Your Own Botmaster, 2nd Edition*. ALICE A. I. Foundation, 2003.

[Weizenbaum, 1966] Joseph Weizenbaum. Eliza–a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1):36–45, January 1966.

Evaluation of a Large-scale Email Response System

Yuval Marom and Ingrid Zukerman
Faculty of Information Technology, Monash University
Clayton, Victoria 3800, AUSTRALIA
{yuvalm,ingrid}@csse.monash.edu.au

Abstract

We are working on a large-scale, corpus-based dialogue system for responding to requests in an email-based help-desk. The size of the corpus presents interesting challenges with respect to evaluation. We discuss the limitations of the automatic evaluation performed in our previous work, and present a user study to address these limitations. We show that this user study is useful for evaluating different response generation strategies, and discuss the issue of representativeness of the sample used in the study given the large corpus on which the system is based.

1 Introduction

A help-desk domain offers interesting dialogue properties in that on the one hand responses are generalized to fit template solutions, and on the other hand they are tailored to the initiating request in order to meet specific customer needs. In recent years, we have been investigating an email-based help-desk task: generating a response to a new request based on a corpus of previous dialogues. The corpus consists of 30,000 email dialogues between customers and help-desk operators at Hewlett-Packard. However, to focus our work, we used a sub-corpus of 6,659 email dialogues which consisted of two-turn dialogues where the answers were reasonably concise (15 lines at most). These dialogues deal with a variety of customer requests, which include requests for technical assistance, inquiries about products, and queries about how to return faulty products or parts.

The size of our corpus presents challenges with respect to evaluation, which raise interesting research questions for practical corpus-based dialogue systems of a scale similar to ours. While automatic evaluations are useful during system development, the quality of a response is a subjective measure that should be judged by users of the system. Thus, user studies provide more realistic evaluations. However, how does one select a representative sample of request-response pairs to present to subjects? Many dialogue systems and other NLP systems are evaluated with user studies comprising 100-200 cases, which requires a considerable but reasonable amount of effort for test subjects and research staff. Statistically, this is an acceptable sample size when a system is based on up to 1000 cases. However, when a system is based on thousands of cases, the representativeness of such small studies is questionable. At the same time, increasing the size of a user study, and therefore the effort required from subjects and testers, may not be practical.

Is there a way to disable the NAT firewall on the CP-2W so I don't get a private ip address through the wireless network?
Unfortunately, you have reached the incorrect eResponse queue for your unit. Your device is supported at the following link, or at `888-phone-number`. We apologize for the inconvenience. URL.

Figure 1: An example where terms in the request are predictive of the response.

In this paper, we report on evaluations of our email-based dialogue system, comparing different response-generation strategies. We show the limitations of an automatic evaluation of the system, and discuss a small user study that we performed in order to address these limitations. The results of our study are encouraging. However, it has its own limitations in addressing our evaluation goals. These limitations are presented as challenges for the dialogue community.

The rest of the paper is organised as follows. In the next section, we give some background to our system and its automatic evaluation. In Section 3, we present our user study, which we follow with a discussion in Section 4. In Section 5, we provide a brief review of evaluation approaches reported in similar systems, and in Section 6, we present concluding remarks.

2 Response generation strategies

2.1 Methods

In previous work we have investigated various response-generation strategies [Zukerman and Marom, 2006]. Our conclusions are that a standard retrieval approach, where a new request is matched in its entirety with previous requests or responses, is successful only in very few cases. A more suitable approach is a predictive one, which uses correlations between features of requests and responses to guide response generation [Marom and Zukerman, 2007]. Figure 1 shows an example of a request-response pair, where the terms in the

Proceedings of the 5th *Workshop on Knowledge and Reasoning in Practical Dialogue Systems*, pages 28–33,
Hyderabad, January 8th, 2007.

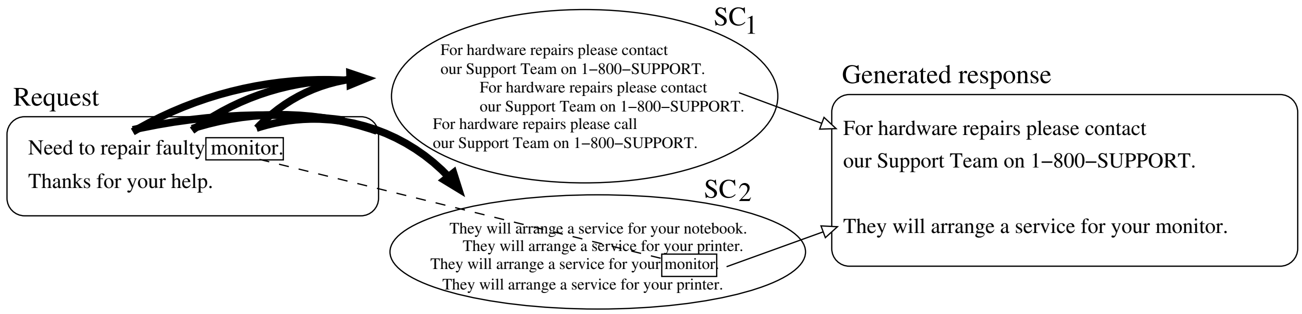


Figure 2: A fictitious example demonstrating the *Sent-Pred* and *Sent-Hybrid* methods.

response do not match any of the terms in the request, but a few of the terms in the request are predictive of the response (terms such as "firewall", "CP-2W" and "network" indicate that the query is network-related and should be redirected).

We have observed that while there is a high language variability in requests in our corpus, the responses exhibit strong regularities, mainly due to the fact that operators are equipped with in-house manuals containing prescribed answers. Further, we have observed that these regularities in the responses can occur at different levels of granularity, with two particular granularities of interest: document (email) and sentence. We have therefore implemented two predictive approaches, where we use machine learning techniques to firstly cluster responses at either of the two levels of granularity, and then learn mappings between terms in the request emails and the response clusters (document clusters or sentence clusters). We refer to these two approaches as *Document Prediction (Doc-Pred)* and *Sentence Prediction (Sent-Pred)* respectively. *Doc-Pred* produces a response by considering the response cluster with the highest prediction score, and if this prediction is higher than a confidence threshold, it selects a representative response document (the one closest to the centroid). *Sent-Pred* produces a response by considering all the sentence clusters that are predicted with sufficient confidence, and then employing multi-document summarization techniques to collate representative sentences into a single response. In the fictitious example shown in Figure 2, the combination of the terms "repair", "faulty" and "monitor" is highly predictive of sentence cluster SC_1, and the sentences in this cluster are sufficiently similar for a confident selection of a representative sentence in the response.[1] These predictive approaches only predict response clusters for which there is sufficient evidence in the corpus, and then select representative items.

In another approach, we investigated tailoring sentences to specific issues raised in the requests. We complemented the *Sent-Pred* method with a retrieval component that biases the selection of a sentence from a cluster based on how well the sentence matches any of the sentences in the request. We refer to this approach as *Sentence Prediction-Retrieval Hybrid (Sent-Hybrid)*. For example, in Figure 2, SC_2 is also highly predicted, but rather than selecting the more representative sentence (containing the term "printer"), we select the sentence that best matches the request (containing the term "monitor"). We employ this retrieval mechanism when we cannot confidently select a representative sentence from a cluster.

The two sentence-level methods (*Sent-Pred* and *Sent-Hybrid*) can produce partial responses. This happens when there is insufficient evidence to predict all the sentences required for a response. In contrast, the document-level method either produces a complete response or does not produce any response. The implementation details of these three methods are described in [Marom and Zukerman, 2007]. Here we focus on evaluation issues raised by the need to evaluate and compare these methods in the context of a very large corpus.

2.2 Automatic evaluation

In the automatic evaluation of our system we were interested in testing firstly the *coverage* of each of the methods — the proportion of requests it can address, and secondly the *quality* of the generated responses, measured separately as *correctness* and *completeness*. To measure correctness we considered the responses written by the help-desk operators as model responses, and then used the precision measure from Information Retrieval [Salton and McGill, 1983] to evaluate the response generated for each request against the model response. This measure determines the proportion of the generated response that matches the actual response. To measure completeness we used the F-Score measure, which is the harmonic mean of recall and precision (recall gives the proportion of the actual response that is included in the generated response) [Salton and McGill, 1983]. The reason for considering precision separately from the combined F-score measure is that the former simply measures whether the generated text is correct, without penalizing it for omitted information. This enables us to better assess our sentence-based methods.

The results of this evaluation are shown in Table 1, and are discussed in detail in [Marom and Zukerman, 2007]. Here we wish only to highlight a few issues. The *Doc-Pred* method produces more complete responses. This is evident from its relatively high average F-Score. Since its average precision is not higher than the precision of the other two methods, the higher F-Score must be a result of a higher average recall. However, the coverage of this method is lower than the coverage of the sentence-level methods. These methods can address additional requests, for which there is insufficient evidence for a complete response.

[1] We obtain this confidence using a measure of cluster cohesion that behaves like entropy [Marom and Zukerman, 2007].

Table 1: Results of automatic evaluation (stdev. in brackets).

Method	Coverage	Precision Ave	F-score Ave
Doc-Pred	29%	0.82 (0.21)	0.82 (0.24)
Sent-Pred	34%	0.94 (0.13)	0.78 (0.18)
Sent-Hybrid	43%	0.81 (0.29)	0.66 (0.25)

The *Sent-Pred* method produces correct responses (high precision) at the expense of completeness (low recall). The *Sent-Hybrid* method extends the *Sent-Pred* method by employing sentence retrieval as well, and thus has a higher coverage. This is because the retrieval component disambiguates between groups of candidate sentences, thus enabling more sentences to be included in a generated response. This, however, is at the expense of precision (and hence F-Score). This lower precision means that the selected sentences differ from the "selections" made by the operator in the model response. However, it does not necessarily mean that the selected sentences are worse than those used by the operator. In fact, our user-based evaluations point to situations where the opposite is the case (Section 4).

Although the automatic evaluation is valuable for comparing and fine-tuning the various methods, it has some limitations. Generated responses should be assessed on their own merit, rather than with respect to some model response, because often there is not one single appropriate response. Also, the automatic evaluation does not inform us of the usefulness of partial responses. The user study presented in the next section was designed to address these limitations.

3 User study

The aim of this study was to obtain an approximation to customers' reactions to the responses generated by the various methods, and thus provide a more subjective evaluation of our system. We asked four judges to assess the responses generated by our system. Our judges were instructed to position themselves as help-desk customers who know that they are receiving an automated response, and that such a response is likely to arrive quicker than a manual response composed by an operator.

To address the limitations of the automatic evaluation mentioned in the previous section, we designed the user study to assess the different methods from the following perspectives:

1. **Informativeness:** Is there is anything useful in the response that would make it a good automatic response, given that otherwise the customer has to wait for a human-generated response? We used a scale from 0 to 3, where 0 corresponds to "not at all informative" and 3 corresponds to "very informative".
2. **Missing information:** Are any crucial information items missing? Y/N.
3. **Misleading information:** Is there any misleading information? Y/N. We asked the judges to consider only information that might misguide the customer, and ignore information that is obviously and inconsequentially wrong, and which a customer would thus ignore, knowing that the response is automated (for example, receiving an answer for a printer, when the request was for a laptop).
4. **Compared to model response:** How does the generated response compare with the model response? Worse/Same/Better.

3.1 Experimental setup

We had two specific goals for this evaluation. First, we wanted to compare document-level versus sentence-level methods. Second, we wanted to evaluate cases where only the sentence-level methods can produce a response, and therefore establish whether such responses, which are often partial, provide a good alternative to a non-response. We therefore presented two evaluation sets to each judge.

1. The first set contained responses generated by *Doc-Pred* and *Sent-Hybrid*. These two methods obtained similar precision values in the automatic evaluation (Table 1), so we wanted to compare how they would fare with our judges.
2. The second set contained responses generated by *Sent-Pred* and *Sent-Hybrid*, for which *Doc-Pred* could not produce a response. The added benefit of this evaluation set is that it enables us to examine the individual contribution of the sentence retrieval component.

Each evaluation set contained 20 cases, randomly selected from the corpus. For each case we presented the request email, the model response email, and the two generated responses, and asked the judges to assess the generated responses on the four criteria listed above. Our four judges, who were from the Faculty of IT at Monash University, had reasonable technical experience on the kinds of issues raised in the help-desk dialogues. We asked the judges to leave a question unanswered if they felt they did not have the technical knowledge to make a judgement, but this did not actually occur.

We have chosen to maximize the coverage of this study by allocating different cases to each judge, and thus avoid a situation where a particularly good or bad set of cases is evaluated by all judges. Because the judges do not evaluate the same cases, we cannot employ standard inter-tagger agreement measures [Carletta, 1996]. However, it is nevertheless necessary to have some measure of agreement, and control for bias from specific judges or specific cases. We do this by performing pairwise significance testing, treating the data from two judges as independent samples.[2] We do this separately for each method and each of the four criteria, and then eliminate the data from a particular judge if he or she has significant disagreement with other judges. This happened with one of the judges, who was significantly more lenient than the others on the *Sent-Pred* method for the first, second and fourth criteria, and with another judge, who was significantly more stringent on the *Sent-Hybrid* method for the third criterion. Thus, each evaluation set contains a maximum of 80 cases.

[2]The statistical test employed here is the Wilcoxon rank sum test for equal medians.

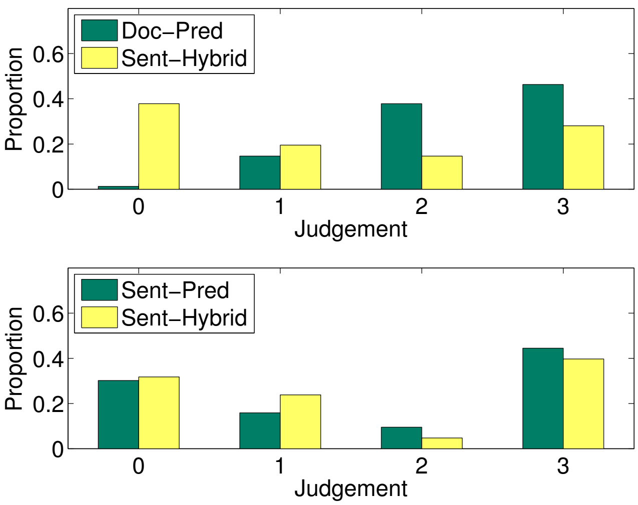


Figure 3: Evaluating the "informativeness" of generated responses.

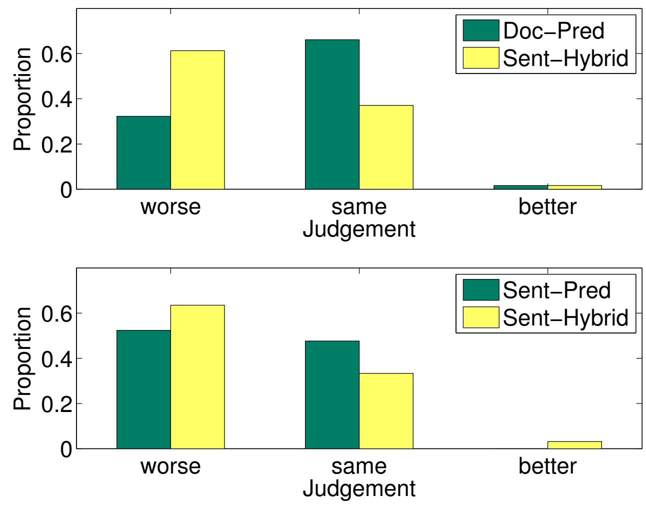


Figure 4: Evaluating generated responses compared to model responses.

3.2 Results

Figure 3 shows the results for the "informativeness" criterion. The top part of the figure is for the first evaluation set, and it shows that when both *Doc-Pred* and *Sent-Hybrid* are applicable, the former receives an overall preference, rarely receiving a zero informativeness judgement. Since the two methods are evaluated together for the same set of cases, we can perform a paired significance test for differences between them. Using a Wilcoxon signed rank test for a zero median difference, we obtain a p-value $\ll 0.01$, indicating that the differences in judgements between the two methods are statistically significant. The bottom part of Figure 3 is for the second evaluation set, comparing the two sentence-based methods. Here there do not appear to be significant differences, and this is confirmed by the paired significance test which produces a p-value of 0.13.

Similar observations are made for the "missing information" criterion. In the first evaluation set, the *Doc-Pred* method is judged to have missing information in 23% of the cases, compared to 57% for the *Sent-Hybrid* method, and the paired significance test produces a p-value $\ll 0.01$. The second evaluation set produces a p-value of 0.11, indicating an insignificant difference between the proportions of cases judged to have missing information, which are approximately 55% for the sentence-level methods. These high proportions are in line with the low F-Scores in the automatic evaluation (Table 1): missing information results in a low recall and hence a low F-Score.

The results for the "misleading information" criterion are as follows. In the first evaluation set, 6% of the responses produced by the *Doc-Pred* method are judged to have misleading information, compared to 15% of the responses generated by the *Sent-Hybrid* method. Although the proportion of misleading information is higher for the latter, the paired differences between the two methods are not statistically significant, with a p-value 0.125. For the second evaluation set, the proportions are 11% and 10% for *Sent-Pred* and *Sent-Hybrid* respectively, and their paired differences are also insignificant with a p-value of 1.0. These low proportions of misleading information are in line with the high precision values observed from the automatic evaluation (Table 1): while responses with a high precision may be incomplete, they generally contain correct information.

Lastly, the results for the "compared to model response" criterion are shown in Figure 4. The top part of the figure, corresponding to the first evaluation set, shows that *Doc-Pred* receives more "same" than "worse" judgements, compared to *Sent-Hybrid*, and they both receive a small proportion of "better" judgements. The paired significance test produces a p-value $\ll 0.01$, confirming that these differences are significant. The bottom part of the figure, corresponding to the second evaluation set, shows smaller differences between *Sent-Pred* and *Sent-Hybrid*, and indeed the p-value for the paired differences is 0.27. Notice that *Sent-Pred* does not receive any "better" judgements, while *Sent-Hybrid* does.

4 Discussion

The results from our previous work (Table 1) showed that the different response-generation strategies are all able to address a significant proportion of the requests, with varying degrees of success. These results were obtained through an automatic evaluation that performed a textual comparison between a generated response and the actual response for a given request. However, these results only provided a limited insight into whether the different strategies achieved their aims. The user study presented in this paper enabled us to evaluate specific characteristics that could only be judged subjectively.

- ***Doc-Pred.*** This document-level strategy attempts to reuse a complete response in the corpus for a new request. The results show that when such a strategy is possible it is better than a sentence-level strategy: the generated response is more informative and complete, and compares more favourably with the model response.
- ***Sent-Pred.*** This sentence-level strategy attempts to produce a response with as much content as is warranted by

evidence in the corpus. Hence, this strategy can yield a partial response. Our results show that indeed this strategy can miss crucial information, but what it does include in a response can be informative and is rarely misleading. The responses are sometimes as good as the model responses.

- ***Sent-Hybrid.*** This hybrid strategy is based on the *Sent-Pred* strategy, but it attempts to tailor a response to specific terms in the request. The main difference between this strategy and *Sent-Pred* is that rather than selecting a representative sentence from a sentence cluster, it selects a sentence that best matches the request. Hence, the generated responses are less general than those produced by *Sent-Pred* (and sometimes less general than the model responses). However, there were no statistically significant differences between the two strategies on any of the criteria we measured in the user study.

It is encouraging that the performance of *Sent-Hybrid* is at least as good as that of *Sent-Pred*, because we saw in the automatic evaluation that *Sent-Hybrid* has a higher coverage (Table 1). However, it is somewhat surprising that *Sent-Hybrid* did not outperform *Sent-Pred* overall. It is worth noting that in a few of the cases, *Sent-Hybrid* produced a better response than the model response. That is, the judges thought that the generated response contained additional useful information not appearing in the model response. However, this did not occur sufficiently to show up significantly in the results.

The similar performance of the two sentence-level methods may be due to a genuine insignificant effect from the retrieval component of the *Sent-Hybrid* method, or due to the fact that an effect could not be observed in the sample that was selected for the user study. Therefore, although the user study was valuable in showing that sentence-level strategies provide useful alternatives when document-level ones cannot be used, it was limited in that it left an aspect of our research inconclusive.

The data in the user study account for 2.4% of the corpus used in the automatic evaluation. Our corpus is divided into topic-based datasets. The data for the user study were selected from these different datasets in proportion to the number of dialogues in each topic. Although this data-selection policy makes the test set fair, it increases the difficulty of drawing specific conclusions. For example, it would be difficult to determine whether a particular response-generation strategy is more suitable for specific topics or for particular kinds of requests. In order to test such possibilities we would need to increase the sample size substantially. Alternatively, we could conduct preliminary automated evaluations for specific conditions, and then target these conditions in user-based evaluations.

These observations point to the need to balance the requirements derived from large corpora with the affordances provided by human subjects. That is, as the sizes of dialogue corpora increase, and several operating parameters of a system need to be considered, the number of requisite trials increases as well. At the same time, the amount of data that subjects can evaluate is limited, more so when they are required to read and judge long texts. These issues must be considered in tandem to devise appropriate sampling protocols for user studies.

Although large-scale, corpus-based systems are being routinely evaluated automatically in NL systems, scant attention has been given to the determination of a suitable sample size for trials with people (Section 5). In contrast, human experiments conducted in the social and medical sciences are concerned with sample sizes. Sampling methodologies, such as power analysis, were developed to help experimenters plan sample sizes [Lewicki and Hill, 2006]. These methodologies take into account factors such as measurement error, type of statistical test used, and desired level of significance. Although some of these factors, such as measurement error, are not always relevant for NL and dialogue systems, we can nevertheless use these methodologies in our studies.

5 Evaluation in related research

A comprehensive review of existing evaluation methods for practical dialogue systems is outside the scope of this paper. Instead, we mention a few systems we have encountered recently, with a particular emphasis on dialogue systems, whose response strategies rely on a corpus, and hence the usefulness of the evaluation depends on the size of the corpus.

There are two systems where the corpus is similar in size to our corpus [Berger and Mittal, 2000; Carmel *et al.*, 2000]. The corpus of the system described in [Berger and Mittal, 2000] consisted of 10,395 call-center request-response pairs, of which they used 70% for training, and 30% for testing. The evaluation on the test set, which is automatic, examined the rank of the real response in the retrieved list. Like our automatic evaluation, this approach assumed that the real response is the best one, but unlike our automatic evaluation, there was no consideration of other responses that might be similar to the real response. That is, the responses that appear near the real response in the retrieved list were not evaluated. The eResponder system [Carmel *et al.*, 2000] retrieved a list of request-response pairs from a corpus and presented a ranked list of responses for a given query. The corpus was an NSF archive called "Ask a Scientist or Engineer", whose size is not mentioned in the paper, but an internet report states that it has "thousands of questions" (`http://content.nsdl.org/wbr/Issue.php?issue=44`). The system was evaluated on 30 queries by a user-based evaluation of the relevance of the top 3, 5 and 10 retrieved responses.

Both of these systems returned a list of responses to the user — they did not attempt to produce a single response. This means that they are concerned with different evaluation issues than those considered here.

Four examples of systems which use smaller corpora are reported in [Lapalme and Kosseim, 2003; Roy and Subramaniam, 2006; Feng *et al.*, 2006; Leuski *et al.*, 2006]. Lapalme and Kosseim's system involved a corpus of 1,568 email dialogues, and evaluations of two tasks: a classification task tested on 144 annotated cases, and a retrieval task tested on 102 cases. Roy and Subramaniam's system involved a corpus of 2,000 transcribed call-center calls, and an evaluation of a clustering task tested on 125 annotated cases. In both of these systems there is also a response-generation task, which was not evaluated. In contrast, Feng *et al.* evaluated their response generation module. Their corpus consisted of 1,236 discussion threads, and they performed a manual evaluation where judges used a criterion similar to our "informativeness" criterion to assess the quality of responses as "exact answer", "good answer", "related answer" or "unrelated answer". The test set contained 66 cases, which is 5.4% of the corpus. This is almost double the proportion of our user study, but the size of the corpus is less than a fifth of ours. Finally, Leuski *et al.*'s system was based on a corpus of 1,261 questions. However, its animated character can utter only 83 distinct responses to these questions. Their test set consisted of 20 subjects, where each asked the system 20 questions. A manual evaluation was then carried out by three judges, who rated the system's responses on a 6-scale criterion that takes into account both utility and discourse quality.

To summarize, the above systems illustrate different types of response-generation tasks, which are evaluated using a variety of criteria, both in automatic and user-based evaluations. However, in the latter, the representativeness of the evaluation sets was not considered. With the increased availability of electronic resources, the consideration of these issues is timely for the dialogue and NL communities.

6 Conclusion

In this paper, we have discussed the evaluation of our corpus-based, response-generation strategies for an email-based, help-desk dialogue system. The various strategies take advantage of the strong regularities that exist in help-desk responses, by abstracting them either at the document level or at the sentence level. They then find correlations between requests and responses to build predictive models for addressing new requests. The hybrid method we presented was designed to overcome the loss of information resulting from abstracting response sentences. The deployment of sentence retrieval in combination with prediction was shown to be useful for better tailoring a response to a request. Our results show that each of the strategies can address a significant portion of the requests, and that when the re-use of a complete response is not possible, the collation of sentences into a partial response can be useful.

We identified limitations of our automatic evaluation, and presented a user study where human judgements provide a more subjective indication of the quality of the generated responses. Although this study addressed some of the limitations of the automatic evaluation, it also posed questions regarding the sampling of data for user-based studies of dialogue systems driven by a large corpus. We have not seen many dialogue systems of this kind in the literature. However, with the constant increase of digital information and archives, more of these systems will be developed, necessitating answers to the questions we have raised.

Acknowledgments

This research was supported in part by grant LP0347470 from the Australian Research Council and by an endowment from Hewlett-Packard. The authors also thank Hewlett-Packard for the extensive anonymized help-desk data.

References

[Berger and Mittal, 2000] A. Berger and V.O. Mittal. Query-relevant summarization using FAQs. In *ACL2000 – Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 294–301, Hong Kong, 2000.

[Carletta, 1996] J. Carletta. Assessing agreement on classification tasks: The Kappa statistic. *Computational Linguistics*, 22(2):249–254, 1996.

[Carmel *et al.*, 2000] D. Carmel, M. Shtalhaim, and A. Soffer. eResponder: Electronic question responder. In *CoopIS '02: Proceedings of the 7th International Conference on Cooperative Information Systems*, pages 150–161, Eilat, Israel, 2000.

[Feng *et al.*, 2006] D. Feng, E. Shaw, J. Kim, and E. Hovy. An intelligent discussion-bot for answering student queries in threaded discussions. In *IUI '06: Proceedings of the 11th international conference on Intelligent user interfaces*, pages 171–177, Sydney, Australia, 2006.

[Lapalme and Kosseim, 2003] G. Lapalme and L. Kosseim. Mercure: Towards an automatic e-mail follow-up system. *IEEE Computational Intelligence Bulletin*, 2(1):14–18, 2003.

[Leuski *et al.*, 2006] A. Leuski, R. Patel, D. Traum, and B. Kennedy. Building effective question answering characters. In *Proceedings of the 7th SIGdial Workshop on Discourse and Dialogue*, pages 18–27, Sydney, Australia, 2006.

[Lewicki and Hill, 2006] P. Lewicki and T. Hill. *Statistics: Methods and Applications*. StatSoft Inc, Tulsa, Oklahoma, 2006.

[Marom and Zukerman, 2007] Y. Marom and I. Zukerman. A predictive approach to help-desk response generation. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI'07)*, Hyderabad, India, 2007.

[Roy and Subramaniam, 2006] S. Roy and L.V. Subramaniam. Automatic generation of domain models for call-centers from noisy transcriptions. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 737–744, Sydney, Australia, 2006.

[Salton and McGill, 1983] G. Salton and M.J. McGill. *An Introduction to Modern Information Retrieval*. McGraw Hill, 1983.

[Zukerman and Marom, 2006] I. Zukerman and Y. Marom. A comparative study of information-gathering approaches for answering help-desk email inquiries. In *Proceedings of the 19th ACS Australian Joint Conference on Artificial Intelligence*, Hobart, Australia, 2006.

A tractable DDN-POMDP approach to affective dialogue modeling for general probabilistic frame-based dialogue systems

Trung H. Bui, Mannes Poel, Anton Nijholt, and Job Zwiers
University of Twente
{buith, mpoel, anijholt, zwiers}@cs.utwente.nl

Abstract

We propose a new approach to developing a tractable affective dialogue model for general probabilistic frame-based dialogue systems. The dialogue model, based on the Partially Observable Markov Decision Process (POMDP) and the Dynamic Decision Network (DDN) techniques, is composed of two main parts, the slot level dialogue manager and the global dialogue manager. Our implemented dialogue manager prototype can handle hundreds of slots; each slot might have many values. A first evaluation of the slot level dialogue manager (1-slot case) showed that with a 95% confidence level the DDN-POMDP dialogue strategy outperforms three simple handcrafted dialogue strategies when the user's action error is induced by stress.

1 Introduction

We aim to develop dialogue management models which are able to act appropriately by taking into account some aspects of the user's affective state. These models are called *affective dialogue models*. Concretely, our affective dialogue manager processes two main inputs, namely the user's action (e.g., dialogue act) and the user's affective state, and selects the most appropriate system action based on these inputs and the context. In human-computer dialogue, this work is difficult because the recognition results of the user's action and affective state are ambiguous and uncertain. Furthermore, the user's affective state cannot directly observe and usually changes over time. Therefore, an affective dialogue model should take into account both the basic dialogue principles (such as turn-taking and grounding) and the dynamic aspects of the user's affect. We found that Partially Observable Markov Decision Processes (POMDPs) are suitable for use in designing these affective dialogue models [Bui *et al.*, 2006].

However, solving the POMDP problem (i.e. finding a near-optimal policy) is computationally expensive. Therefore, almost all developed POMDP dialogue management approaches (mainly for spoken dialogue systems, see [Williams *et al.*, 2005] and the earlier work cited in this paper) are limited to toy frame-based dialogue problems with the size of several slots. Recently, William and Young [2006] proposed a scaling-up POMDP method called CSPBVI to deal with the multi-slot problem. The dialogue manager is decomposed into two POMDP levels, a master POMDP and a set of summary POMDPs. However, they have achieved this goal by oversimplifying the user behavior (assuming when the users are asked about a certain slot, they only provide a value for that slot) and reducing the size of the POMDP structure (e.g. approximating the number of values of each slots by only two values $best$ and $rest$). Furthermore, trials with real users, which allow to validate the system response time, were not conducted.

In our research, we opted for another approach which focuses on real-time online belief update for a general probabilistic frame-based (or slot-filling) dialogue system. Each slot is first formulated as a POMDP and then approximated by a set of Dynamic Decision Networks (DDNs). The approach is, therefore, called the DDN-POMDP approach. It has two new features, (1) being able to deal with a large number of slots and (2) being able to take into account the role of the user's affective state in deriving the adaptive dialogue strategies.

In this paper, we first describe our general affective dialogue model using the DDN-POMDP approach. Then, we present a simulated route navigation example and a first evaluation of our method. Finally, we present conclusions and discuss future work.

2 The DDN-POMDP approach for the frame-based affective dialogue problem

Our Affective Dialogue Model (ADM) is composed of two main parts: (1) the slot level dialogue manager and (2) the global dialogue manager. The first part is composed of a set of n slots $f_1, f_2, ..., f_n$ where each slot f_i is formulated as a POMDP (called the slot-POMDP and denoted by SP_i). The second part, the global dialogue manager, is handcrafted. It aims to keep track of the current dialogue information state and to aggregate the system slot actions nominated by the slot-POMDPs. These two parts and the ADM activity process are explained in detail in the next sections.

2.1 Slot Level Dialogue Manager

The state set of each slot-POMDP SP_i is composed of the user's goals for the slot i (Gu_i), the user's affective states (Eu), the user's actions for the slot i (Au_i), and the user's

Proceedings of the 5th Workshop on Knowledge and Reasoning in Practical Dialogue Systems, pages 34–37,
Hyderabad, January 8th, 2007.

grounding states for the slot i (Du_i). The observation set is composed of the observed user's actions for the slot i (OAu_i) and the observed user's affective states (OEu). Eu and OEu are identical for all slots. The action set is the system actions for the slot i.

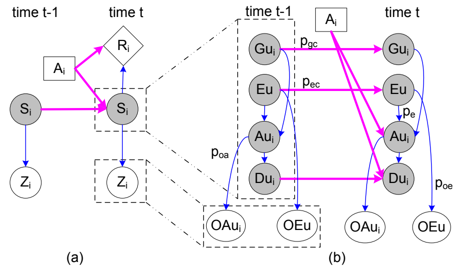


Figure 1: (a) Standard POMDP, (b) Two time-slice of the factored POMDP for slot i

Figure 1b shows a structure of the factored POMDP for slot i of our route navigation example (see Section 3). The features of S_i, A_i, Z_i (Figure 1a), and their correlation form a two time-slice Dynamic Bayesian Network (2TBN). Parameters $p_{gc}, p_{ec}, p_e, p_{oa}$, and p_{oe} are used to produce the transition and observation models in case no real data is available, where p_{gc} and p_{ec} are the probabilities that the user's goal and emotion change; p_e is the probability of the user's action error being induced by emotion; p_{oa} and p_{oe} are the probabilities of the observed action and observed emotional state errors. The reward model depends on each specific application. Therefore it is not specified in our general slot level dialogue manager.

For example, the full-flat slot-POMDP model SP_i of a simplified version of the route navigation example (see Section 3.1) is composed of 61 states (including an absorbing end state), eight actions, and ten observations.

We are interested in finding a solution to directly implement this POMDP model for practical dialogue systems. One intuitive approach is to compute the optimal dialogue strategy using a good approximate POMDP algorithm and use the result for selecting the appropriate system action. We used this approach to find the optimal dialogue policy for the above SP_i [Bui *et al.*, 2006] using Perseus [Spaan and Vlassis, 2005]. However, this approach does not work when the number of slot values and the user's affective states increases (for example, when $|Eu| = 5, m_i = 10$, the full-flat model of SP_i increases up to 1201 states, 22 actions, and 60 observations).

Therefore, to maintain the tractability and allow real-time online belief state update, we approximate each slot-POMDP by a set of $|A_i|$ k-step look-ahead DDNs (kDDNAs) ($k \geq 0$). A kDDNA has ($k + 2$) slices. The first two slices are similar to the 2TBN showed in Figure 1b, the next k slices are used to predict the user behavior in order to allow the dialogue manager to select the appropriate system action. Figure 2 shows a structure of the kDDNA ($k = 1$) used for SP_i of our route navigation example. The connection from the action nodes to immediate reward nodes in the next slices indicates that when a system slot action is selected that lead to the absorbing `end` state (such as ok or $fail$), the reward in all next slices are equal to `0`.

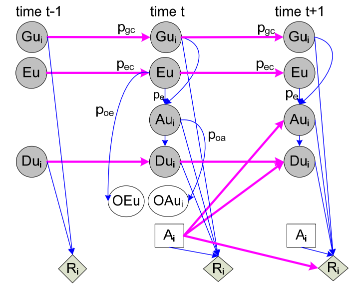


Figure 2: kDDNA with one-step look-ahead ($k = 1$)

2.2 Global Dialogue Manager

The global dialogue manager is composed of two components, the dialogue information state (DIS) and the action selector (AS).

The DIS is considered as the active memory of the dialogue manager, it automatically updates and memorizes the current probability distributions of the user's goal, affective state, action, grounding state of all slots and the recently observed user's action and affective state. The DIS is formally defined by the tuple $\langle P(Gu), p(Eu), P(A_u), P(D_u), oau, oeu \rangle$, where $P(Gu), P(A_u), P(D_u)$ are n dimensional vectors containing the probability distributions of the user's goal, action, grounding state aggregated from Gu_i, Au_i, and Du_i ($i \in [1, n]$), respectively; $p(Eu)$ is the probability distribution of the user's affective state; $oau \in OAu$ and $oeu \in OEu$ are the recent observed user's action and affective state, where OAu is constructed by the user's dialogue act types, slots and slot values [Bui, 2006], Eu and OEu are defined in Section 2.1.

The AS component is responsible for aggregating the system's slot actions nominated by slot-POMDPs. The system action set used by the AS component is constructed by the system's dialogue act types, slots, and slot values [Bui, 2006] and two special actions $giveSolution$ and $stop$. The AS is heuristic and application-dependent. An example of a set of rules to select global system action is described in Section 3.1.

2.3 Affective Dialogue Manager Activity Process

When the dialogue manager is initialized, it loads n slot-POMDP parameter files and creates a set of kDDNAs (m_i kDDNAs are created from the slot-POMDP parameter file i). Depending on each specific application, some slots the values of which can change in time (these slots are called list processing slots, for example the types of food in a selected restaurant [Bui *et al.*, 2004]) can use the same set of kDDNAs. The dialogue manager and the user then only work with a small number of list processing values (i.e. the ordinal numbers), a mapping between these ordinal numbers and the real values is done automatically by the dialogue manager.

The entire process of the affective dialogue manager is explained in this section by a cycle of four steps.

- **Step 1:** When the dialogue manager starts, the kDDNAs nominate greedy actions to the GDM based on the set of prior probability distribution specified in the slot-POMDP parameter files. These actions are combined by the action selector. The output is sent to the user (through the output generation module).

- **Step 2:** The dialogue manager then receives the observed user's action and user's affective state ($oau \in OAu$ and $oeu \in OEu$). The kDDNAs relevant to oau are activated to compute the next slot action. The DIS is also updated.
- **Step 3:** All new actions computed by the selected kDDNAs are sent to the action selector to produce the new system action.
- **Step 4:** The process repeats from step 2 until the GDM selects either $giveSolution$ or $stop$ action.

3 Implementation & Evaluation

The test example is a simulated route navigation in an unsafe tunnel. A serious accident has happened in a huge tunnel. A rescue team (n persons) is sent to the unsafe part of the tunnel to evacuate a large number of injured victims. The rescue members are currently at different locations in the tunnel. The team leader (denoted by "the user") interacts with the dialogue system (located at the operation center) to get the route description for the evacuating task. The system is able to produce the route description when knowing the locations of the rescue members. Furthermore, the system can infer the user's stressful state and use this information to act appropriately.

3.1 Implementation

The above example is formulated as n slots $(f_1, f_2, ..., f_n)$ and all slots have the same set of m values which are the locations in the tunnel $(v_1, v_2, ..., v_m)$. The user's affective states are five levels of the user's stress: no stress (no), low stress (low), moderate stress ($moderate$), high stress ($high$), and extreme stress ($extreme$). The user's grounding state is composed of two values $notstated$, $stated$. The user's dialogue act type set is $answer$, yes, and no. The system's dialogue act type set is ask, $confirm$, ok, $fail$, $giveSolution$, $stop$ (the two last dialogue act types are only used at the global dialogue manager level as being defined in Section 2.2). The user's goal is to find out the route description for n locations (known by the user). The system aims at showing the user the correct route navigation as soon as possible.

Slot level dialogue manager representation

Slot f_i is represented by $Gu_i = \{v_j | j \in [1, m]\}$, $Eu = \{no, low, moderate, high, extreme\}$, $Au_i = \{answer(v_j), yes, no | j \in [1, m]\}$, $Du_i = \{notstated, stated\}$, $OAu_i = Au_i$, $OEu = Eu$, $A_i = \{ask, confirm(v_j), ok(v_j), fail | j \in [1, m]\}$.

We use two criteria to specify the reward model for each slot, helping the user obtain the correct route description as soon as possible and maintaining the dialogue appropriateness [Williams *et al.*, 2005]. Concretely, if the system `confirms` when the user's grounding state is `notstated`, the reward is `-2`, the reward is `-3` for action `fail`, the reward is `10` for action `ok(x)` where $gu_i =$ `x` (`x` $\in \{v_j | j \in [1, m]\}$), otherwise the reward is `-50`. The reward for any action taken in the absorbing `end` state is `0`. The reward for any other action is `-1`. The high negative reward for selecting the incorrect slot value (`-50`) is used to force the dialogue manager agent to confirm the information provided by the user when the user's stress level is high.

The probability distributions for each kDDNA are generated using the parameters $p_{gc}, p_{ec}, p_e, p_{oa}, p_{oe}$ defined in Section 2.1 (p_{oa}, p_{oe} can be viewed as the speech recognition error and the stress recognition error) and two new parameters K_{ask} and $K_{confirm}$, where K_{ask} and $K_{confirm}$ are the coefficients associated with the ask and $confirm$ actions (i.e. $p_e(ask) = p_e/K_{ask}; p_e(confirm) = p_e/K_{confirm}$). We assume that when the users are stressful, they make more errors in response to the system ask action than the system $confirm$ action because the number of possible user's action in response to ask is greater than to $confirm$ when the user is not stress.

Global dialogue manager representation

The sets of observed user's actions and system actions are now represented by $OAu = \{answer(I), yes(I), no(I) | I \subseteq \{(f_i = v_i^*) | i \in [1, n]\}, v_i^* \in \{v_i | i \in [1, m]\}\}$, $A = \{ask(I), confirm(J), giveSolution(L), stop | I \subseteq \{f_i | i \in [1, n]\}, J \subseteq \{(f_i = v_i^*) | i \in [1, n]\}, L = \{(f_i = v_i^*) | i \in [1, n]\}, v_i^* \in \{v_i | i \in [1, m]\}\}$

The action selector generates the global system action based on the following rules (applying the first rule that satisfies the set of nominated actions):

1. If all slots nominate ask action then the global action is $ask(f_1, f_2, ..., f_n)$ or $ask(open)$,
2. If all slots nominate $confirm$ action then the global action is $confirm((f_1 = v_1^*), (f_2 = v_2^*), ..., (f_n = v_n^*))$ or $confirm(all)$,
3. If all slots nominate ok action then the global, action is $giveSolution((f_1 = v_1^*), (f_2 = v_2^*), ..., (f_n = v_n^*))$,
4. If some slots $(f_1^*, f_2^*, ..., f_i^*)$ nominate $confirm$ action with the values $(v_1^*, v_2^*, ..., v_i^*)$ then the global action is $confirm((f_1^* = v_1^*), (f_2^* = v_2^*), ..., (f_i^* = v_i^*))$,
5. If some slots $(f_1^*, f_2^*, ..., f_i^*)$ nominate ask action then the global action is $ask(f_1^*)$,
6. Otherwise, the global action is $stop$.

The current version of our implemented dialogue manager prototype is able to handle hundreds of slots, each slot can have many values. When a slot has hundreds or thousands of values (called many-value slot), directly embedding these values into the kDDNAs will lead to a significant delay in the belief update time. One of our solutions in this case is to formulate the many-value slot as a list processing slot as mentioned in Section 2.3. A dialogue example of the 10-slot case ($n = 10, m = 10, p_{gc} = 0, p_{ec} = p_e = p_{oa} = p_{oe} = 0.1, K_{ask} = 1, K_{confirm} = 10, k = 1$) is described in [Bui, 2006].

3.2 Evaluation

The performance of the DDN-POMDP dialogue strategy depends on both the global dialogue manager and the slot level dialogue manager (see Section 2).

Currently a simulated user model for the general n-slot case which is appropriate for a quantitative evaluation of the DDN-POMDP approach has not been available yet, therefore in this section we first evaluate the slot level dialogue manager by comparing the DDN-POMDP dialogue strategy with a random dialogue strategy and three simple handcrafted dialogue strategies for 1-slot case: (a) SDS-HC1 (first ask and then select ok action if $oau = answer$), (b) SDS-HC2 (first ask, then $confirm$ if $oau = answer$ and then select ok action if $oau = yes$), (c) ADS-HC (first ask, then $confirm$ if $oau = answer$ & $oeu = stress$ and select ok action if $oau = yes$).

The evaluation is conducted by letting each dialogue strategy interact with the same simulated user (the simulated user model is constructed using the 2TBN described in Figure 1b).

Figure 3 shows the average return of 10000 dialogue episodes of five dialogue strategies when the probability of the user's action error being induced by stress p_e changes from 0 (stress has no influence to the user action selection) to 0.8 (stress has high influence to the user action selection). The results of the average return (Figure 3) show that with a 95% confidence level the DDN-POMDP dialogue strategy outperforms all other remaining dialogue strategies when $p_e \geq 0.1$. The DDN-POMDP copes well when the user's action error being induced by stress increases. An example of the interaction between the DDN-POMDP dialogue manager and the simulated user (10 dialogue episodes) is shown in [Bui, 2006].

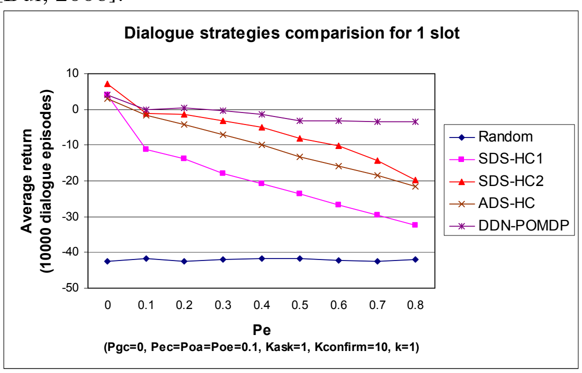


Figure 3: Average return of five dialogue strategies ($p_e \in [0, 0.8]$)

Figure 4 shows that the DDN-POMDP dialogue strategy also copes well with the observed user's action error p_{oa} (for example, the ASR error). The performance of all strategies in Figure 3 and 4 is low when p_e, p_{oa} increases because we set a strong negative reward when the system chooses incorrect solution and when the user's stress is $extreme$, the user acts randomly. When the observed user's action error is too high ($p_{oa} \geq 0.6$), the DDN-POMDP dialogue manager always selects $fail$ action therefore the average return is a constant (equal to -4). One interesting point is that the dialogue strategy SDS-HC2 copes well with the change of p_e (Figure 3) but its performance decreases rapidly when p_{oa} increases (Figure 4).

4 Conclusions and future work

The presented DDN-POMDP approach is shown to be able to handle a large number of slots and keep track of the user's affective state. A first evaluation has been conducted to compare the DDN-POMDP performance with three simple handcrafted dialogue strategies when the user's action error is induced by stress. We plan to evaluate the model with an n slots case by comparing the DDN-POMDP dialogue strategy

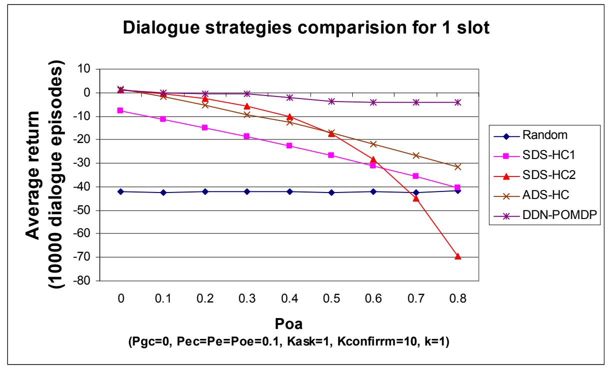


Figure 4: Average return of five dialogue strategies ($p_{oa} \in [0, 0.8]$)

with other well-developed handcrafted dialogue strategies for frame-based dialogue systems such as [Bui *et al.*, 2004]. Another issue is to study the real user behavior in crisis such as in the air traffic control domain in order to improve the user simulator.

Although it is hard to handle really complex dialogue systems using only POMDPs, this approach sheds light to a hybrid solution by combining traditional rule-based and POMDP approaches which hopefully can solve a part of many challenges in developing affective dialogue systems.

Acknowledgments

This work is part of the ICIS program (http://www.icis.decis.nl). ICIS is sponsored by the Dutch government under contract BSIK 03024.

References

[Bui *et al.*, 2004] T.H. Bui, M. Rajman, and M. Melichar. Rapid dialogue prototyping methodology. In *Proceedings of the 7th International Conference on Text, Speech & Dialogue (TSD)*, pages 579–586, Brno, Czech Republic, September 8-11 2004.

[Bui *et al.*, 2006] T.H. Bui, J. Zwiers, M. Poel, and A. Nijholt. Toward affective dialogue modeling using partially observable markov decision processes. In *Proceedings of Workshop Emotion and Computing, 29th Annual German Conference on Artificial Intelligence*, 2006.

[Bui, 2006] T.H. Bui. A tractable ddn-pomdp approach to affective dialogue modeling for general probabilistic frame-based dialogue systems. Technical report, University of Twente, 2006. (to appear).

[Spaan and Vlassis, 2005] M.T.J. Spaan and N. Vlassis. Perseus: Randomized point-based value iteration for pomdps. *Journal of Artificial Intelligence Research*, 24:195–220, 2005.

[Williams and Young, 2006] J.D. Williams and S. Young. Scaling pomdps for dialog management with composite summary point-based value iteration (cspbvi). In *AAAI Workshop on Statistical and Empirical Approaches for Spoken Dialogue Systems*, 2006.

[Williams *et al.*, 2005] J.D. Williams, P. Poupart, and S. Young. Factored partially observable markov decision processes for dialogue management. In *4th Workshop on Knowledge and Reasoning in Practical Dialog Systems*, 2005.

Case Based Utterance Generating for An Argument Agent

Takahiro Tanaka **Norio Maeda** **Daisuke Katagami** **Katsumi Nitta**
Tokyo Institute of Technology
{takat, maeda, katagami, nitta}@ntt.dis.titech.ac.jp

Abstract

To settle disputation, Alternative Disputation Resolution (ADR) has become popular as a replacement for trials. However, to lean mediation skills, a great deal of training is needed. So we have developed the online mediation support system for online mediator education. In this paper, we present an overview of the system and the argument agent which participates with the moot mediation as an opponent for self training. Moreover we explain the method of utterance generating for agent's utterance by retrieving similar situations from the case base.

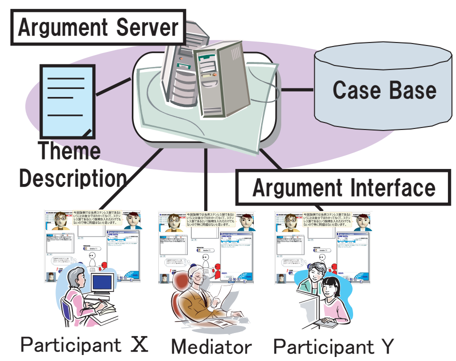


Figure 1: Overview of the System

1 Introduction

In recent years, the number of problems with online shopping and online auctions are on the increase. Instead of the settlement by trial, prompt and low-cost mediation and arbitration called Alternative Disputation Resolution (ADR) is in the spotlight.

Non-legal professionals can carry out mediation; however, as the demand increases there are not enough mediators. Trainee mediators not only need education on the procedures and skills for actual mediation, they also need to practice with sample cases, moot mediation [1]. However, when mediators practice, they always need to be accompanied by an instructor. They have to be together in the same place and this is very difficult. To deal with, we have done research into developing a system for training mediators [2]. The system consists of an online mediation environment and an advice agent instead of human teacher [3]. The online mediation environment not only provides an online argument environment, but also assists participants by storing cases and showing examples of similar cases. However, in the present problem of the system, the moot mediation needs three participants, mediator and both parties. Therefore, the learner can't study mediation skill by him/herself. To deal with, we are developing the argument agent which participates with the moot mediation as an opponent for self training.

In this paper, we exlpain the way of generating agent's utterance using similar situation from case base. In Section 2, we introduce the online mediation support system and present an overview of the argument agent and the case based utterance generating for the argument agent in Section 3. Finally, in Section 4, we explain the conclusion.

2 Online Mediation Support System

The system we are developing is intended to provide an online mediation environment and to assist mediators in developing their decision-making ability. This study brings together various mediation practices based around sample cases and suggests an educational model using simulation to assist trainers to acquire mediation skills by analyzing / comparing them.

2.1 Overview of the System

Figure 1 presents an overview of the "Online Mediation Support System". Online mediations are carried out by students connecting to the server through the Internet and using the Argument Interface. A trainee mediator and the two parties agents participate. The Theme Description includes information about the case to be mediated; it consists of issue points and relationship among issue points.

Each utterance is indexed by reference to the Theme Description, and this is stored in the case base. During the mediation, mediator can retrieve similar past cases and find suggestions to help in selection of the next move. Assistance is also provided by an advice agent suggesting the next possible move to the mediator.

Proceedings of the 5th Workshop on Knowledge and Reasoning in Practical Dialogue Systems, pages 38–41,
Hyderabad, January 8th, 2007.

2.2 Theme Description

The case data used in this research are natural-language texts representing conversations in mediations. To be able to analyze them statistically and to search for similar situations from the case base, we propose to index the remarks referring to the Theme Description. The description consists of three types of information: list of issue points, relationship among issue points and keywords of issue point. The system and the argument agent use this description as background knowledge of the mediation theme for searching similar situation, appreciate content of remarks and so on.

Issue Point

For mediators, it is important to understand what an issue of the current trouble is. Therefore, we analyze these mediation logs and extract features of troubles from the view of issue points [4], [5]. When instructors set cases to be used in mediation training, they should know what the points at issue are. Each issue point has an ID, parameters that show which parties are favorable and types of information (share or secret). For example, in mediations for auction problems, the following points at issue are to be included;

c1: The product auctioned has a defect. [X, Share]

- f11: It has a flaw. [X, Share]
- f12: It is broken. [X, Share]

c2: The product description was not complete. [X, Share]

- f21: The product picture showed the defective part. [Y, Share]
- f22: The description didn't identify the defect. [X, Share]

c3: A complaint was justified. [X, Share]

- f31: The complaint was made in good time. [X, Share]
- f32: It was a serious enough defect to cancel the contract. [X, Secret(Y)]

c4: The seller had announced "NO CLAIM NO RETURN" on the auction site. [Y, Share]

Relationship among Issue Points

These issue points are not independent. There is a relationship among them, attack and support. The relationship is important factor because it influences the progress of mediation. For example, f11 is a more concrete issue than c1. Point c3 and c4 contradict each other because f3 is advantageous for the buyer(X) and f4 for the seller(Y). In this example, we say "f11 supports (+) c1" and "c4 attacks (-) c3." These relationships are represented in the diagram (Figure 2).

Keyword of Issue Point

Recognition of Issue-points from sentences of natural language is generally hard problem because it needs high-level natural language analysis technologies. However, in our case, it is not so hard because the situation is limited by mediation theme. Therefore, instructors can expect and prepare keywords for each issue point (Table 1). These keywords are used for extracting issue points from remarks.

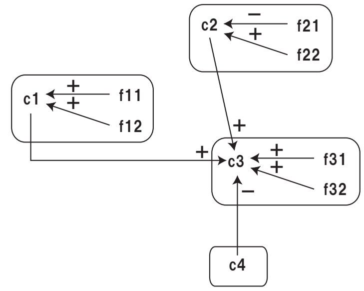


Figure 2: Relationship among issue points

Table 1: Example of Keyword

f11	spot, stain, dirt, blemish, blotch
f12	problem, failure, breakdown
f21	picture, vague, explanation

Issue point is recognized by observing whether the sentences include such keywords or not by performing a morphological analysis of the text. We are comparing the issue points of 6 mediation records (368 utterances) that were selected at random from 11 cases identified by the system with those that were identified by a human expert. The average of precision is 52.6 % and recall is 63.7 %. The cost for constructing the case base by the instructors can be reduced from "add all issue points manually" to "correct issue points if necessary".

2.3 Argument Interface

Trainee mediator carry out the online mediation with the Argument Interface as shown in Figure 3. The interface is implemented with Flash for running on web browser. Mediator inputs "remark texts (utterances)", "link data" and "avatar expressions". "Link data" is the data that represent how the present remark relates to which past remarks. Specifically, "link data" consist of "target past remark ID" and the "type of relation to the remark (new proposition, concession with and denial of past remarks, question and answer etc.)". "Avatar expressions" are facial expressions and gestures of animated agents: Cool, Happy, Angry, Sad and Surprised. The animated agent exhibits selected expression and voices the text. Facial expression is important nonverbal information for mediation [6].

2.4 Construction of a Case Base

Consequently, one utterance has two pieces of data be indexed; (1) the point at issue in the utterance (2) the relation (link data) to the previous utterance. After end of mediation, the system outputs the mediation record as a XML.

3 Argument Agent

The argument agent argues in the online mediation as a participant or both parties. Figure 4 shows the architecture of the agent. The agent consists of 5 modules as following;

- Argument Extract Module

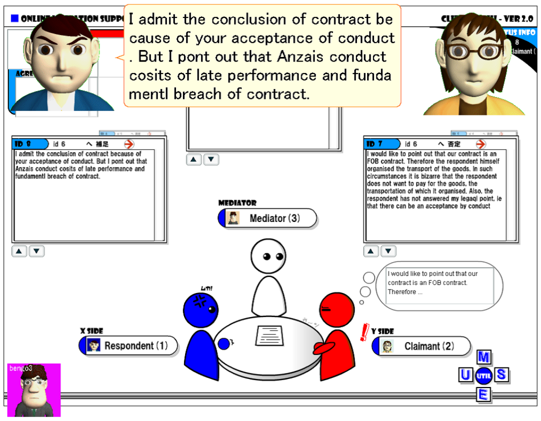


Figure 3: Argument Interface

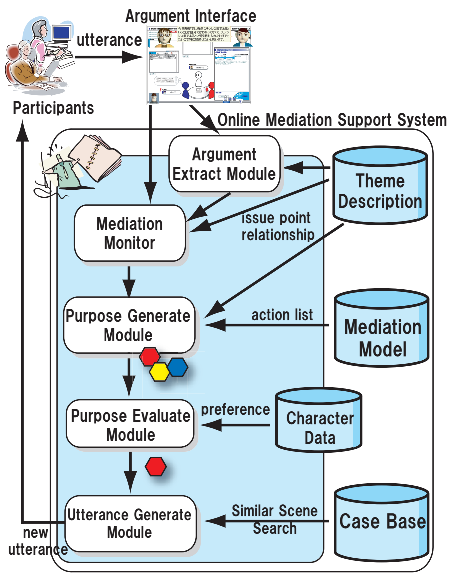


Figure 4: Architecture of Argument Agent

- Mediation Monitor
- Purpose Generate Module
- Purpose Evaluate Module
- Utterance Generate Module

At first, the agent receives text of utterance, link data, issue points and facial expression from participant's utterance. Referring to the Theme Description, the Argument Extract Module extracts argument formulas as assertion from the utterance. The Mediation Monitor [3] updates the situation of mediation which includes following data; which issue point was explained, assertion list of the mediation, participant's knowledge and close/open issue point etc. Then, using new situation, the Purpose Generate Module lists all possible purpose of agent's utterance based on the Mediation Model and the theme. The Purpose Evaluate Module evaluates these purposes and selects most suitable one from the list. The evaluation is based on the theme and the Character Data as following; whether the issue point is advantageous to the agent or not, how many other points exist which attack the issue point etc. Finally, the Utterance Generate Module retrieves reply candidate from similar scene in the case base by the similar scene search, then, generates agent's utterance based on the reply candidate.

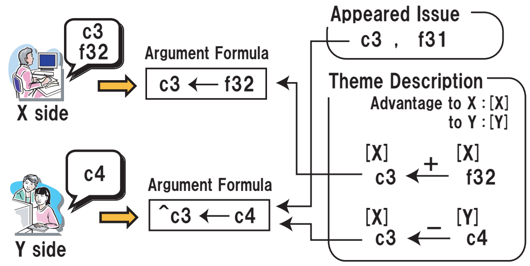


Figure 5: Extracting Argument Formula

3.1 Argument Extract Module

This module extracts the argument formula from text of utterance. The extraction is referring to the theme description as background knowledge. In this paper, the issue point ID means the proposition when that is described as the formula. Extracting argument formulas, the agent can estimate the relationship among issue points contained in the same utterance and whether the utterance is positive or negative assertion.

Extracting argument formula

Figure 5 shows a example of extracting argument formula by this module. The appeared issue means the shared knowledge which is already mentioned in the mediation. When new utterance which has issue points is inputted, the module extracts argument formulas based on the appeared issue points and the theme. If the participant X explains about c3 and f32, the module will extract the argument formula "$c3 \leftarrow f32$" from the support relationship between them and both issues are advantageous to X in the theme. Also, if the participant Y explains about c4, the module finds the attack relationship between c3 and c4 in the theme and c3 is available for the extraction because it was appeared in the mediation. Finally, the module will extract the argument formula "$\wedge c3 \leftarrow c4$" referring to the advantage of each issue points.

We evaluated the accuracy of the module comparing formulas extracted by the module with those that were extracted by the human from 6 mediations based on different 3 themes (2 mediations selected at random for each theme). To remove error of issue point recognition, all mediation record was corrected by the man. The average of precision is 88.2%, and the average of recall is 90.5%. Also, the precisions are almost same value in 3 themes. Therefore, the method can extract argument formulas of utterance independently from the theme description.

3.2 Purpose Generate Module

This module generates the purpose of agent's utterance based on the situation and theme description. The purpose consists of utterance type and content (argument formula).

Table 2: Part of the Mediation Model

Action of Party X	Action of Mediator	Action of Party Y	
		Disagree	Agree
Propose(P)	Right(Y) Ask for Explain(A)	Deny(P) Propose (^ P←Q)	Concede(P)
Deny(P)	Right(Y)	Explain(P,A)	Withdraw(P)
Concede(P) Concede(A)	Right(X) Right(Y)		

At first, this module lists possible types as agent's reply from link data referring to the Mediation Model. The model contains a list of actions which participants can take. We shows a part of the model in Table 2. We employed simpler version of an argumentation model based on the Pleadings Game [7].

For each listed types, the module decides suitable argument formula as agent's reply from the opponent utterance. For example, if the opponent utterance includes formula "$c3$", R will become to $c3$ for agree action. If the type is disagree action and there is issue point $c4$ in agent's knowledge which attacks $c3$, the module generates argument formula " ^ $c3 \leftarrow c4$" as R.

3.3 Utterance Generate Module

Referring to the purpose of utterance, this module generates agent's utterance using the Similar Scene Search. The search returns the list of reply candidates, so the agent compares these candidates with the purpose and select best one.

Similar Scene Search

Searching similar scenes is usually not easy because those similar utterances can always be slightly different from one another, as the expressions are different, and what we understand from them differs depending on previous utterances. To make searching for similar scenes easier, we propose a search method using an index of utterances [8]. This method goes back to two of the previous utterances from the present subject utterance (query). Then, the issue point, argument formulas and the relation (link data) are compared within these three utterances and similar scenes are assessed. Finally, the search shows reply candidates from similar scenes as the result.

We evaluated the accuracy of similar disputation scene search using argument formula using 11 mediation records about the same theme. To remove error of issue point recognition, all mediation record was corrected by the man. Also, we compared the search with the method using vector-space model. The precision is 63.0%, it is higher than the vector-space model (23.8%). The recall is 87.0% (vector-space model is 45.5%), therefore almost similar scenes were retrieved by this search.

Example

If any similar scenes are retrieved, the module uses the text for the reply text. We show an example of agent's reply on the online mediation as following; Human mediator, successful bidder and agent exhibitor participate with the mediation of online auction trouble. These texts are translated into English from Japanese.

- [ID=5] Speaker=Successful Bidder, Type=propose
  -Argument= ^ $f5 \leftarrow f10$
  "In the explanation of auction, he didn't explain about the material of the muffler. Also, in recent years, the company Z produces mufflers from stainless steel only. So, I thought the exhibited item made with stainless steel."
- [ID=6] Speaker=Mediator, Type=right
  "Please explain your opinion about the point."
- [ID=7] Speaker=Exhibitor(Agent), Type=denial
  -Argument= ^ $f10 \leftarrow f11$
  "The company produced the muffler from other materials several years ago. I inquired of the company after your claim."

Using similar scene, the agent explained the counter propose(^ $f10 \leftarrow f11$) to the opponent one(^ $f5 \leftarrow f10$).

4 Conclusion

We explained the argument agent for mediator training on the Online Mediation Support System. The agent monitors the process of mediation and participates with the moot mediation as an opponent for mediator's self training. In particular, we explained the method of utterance generating for agent's utterance by retrieving similar situations from the case base.

References

[1] John Barkai, Teaching Negotiation and ADR: The Savvy Samurai Meets the Devil, 75 Nebraska Law Review 704, 1996.

[2] Takahiro Tanaka, et. al, Online Support System for Mediator Education, Proceedings of IJCAI 2005, pp.1702-1703, Poster Session, 2005.

[3] Takahiro Tanaka, et. al, Advice Agent for Online Mediator Education, International Workshop of Agent-Based Systems for Human Learning, AAMAS06, pp.43-48, 2006.

[4] Ashley, K.D, Reasoning with cases and hypotheticals in HYPO, Int.J.ManMachine Studies 34, pp.753-796, 1991.

[5] Vincent Aleven, Teaching Case-Based Argumentation through a Model and Examples, University of Pittsburgh, 1997.

[6] Masahide Yuasa, Yoshiaki Yasumura and Katsumi Nitta, Negotiation Support Tool Using Emotional Factors, Proceeding of IFSA-NAFIP2001, pp2096-2911,2001.

[7] Thomas Gordon, The Pleadings Game, Ph.D.Thesis, 1993.

[8] Takahiro Tanaka, et. al, Case Based Online Training Support System for ADR Mediator, International Workshop of Artificial Intelligence and Legal Education, ICAIL 2005, pp22-27, 2005.

Multi-robot Dispatch

Andrew M. Olney
University of Memphis
Institute for Intelligent Systems
aolney@memphis.edu

Abstract

This paper reports the ongoing development of a natural language interface used to implement multi-robot dispatch. A human dispatcher uses the interface to direct multiple robotic tugs performing cargo loading and unloading tasks at an airport. The airport environment is highly dynamic and hazardous, having human ground crews, tugs, and planes as moving obstacles. In this paper we describe the task, current architecture, and development strategy.

Figure 1: An airport tug

1 Introduction

Two research streams have matured to the point that they may be combined for practical, real world applications. The first stream consists of task-oriented natural language interfaces. These systems are multimodal and frequently use an assistant agent that helps the user manage a complex situation ([Allen *et al.*, 1995], [Allen *et al.*, 2000], [Stent *et al.*, 1999]). The second research stream consists of autonomous robots moving at high speeds in hazardous conditions. The recent successes of from DARPA Grand Challenge suggest that these research robots are approaching viability in real world tasks [Montemerlo *et al.*, 2006].

In this paper, we describe a novel problem whose solution draws on both research streams. Our problem is to dispatch and monitor robotic tugs performing cargo loading and unloading tasks at an airport. While previous research indicates that a practical solution is reachable, the dispatch task is of such complexity that a solution is challenging. For example, [Lemon *et al.*, 2003] dispatches a single robot. However, in an airport environment, it is not cost-effective for one person to control one robot when that person could just as easily drive a non-robotic tug. Therefore multiple robots must be controlled by a single human dispatcher, which compounds the complexity of the task. Likewise, the DARPA grand challenge winner [Montemerlo *et al.*, 2006] cannot be seamlessly transported to our domain. Not only is the airport environment full of moving obstacles, but collisions with these obstacles has an unacceptable cost.

The rest of the paper is outlined as follows. Section 2 describes the airport tug task that is managed by the dispatch interface. Section 3 presents an overview of the system's architecture. Section 4 presents the development strategy being used to create the dispatch interface and associated autonomous robots. Section 5 concludes.

2 The Airport Tug Task

Airport tugs are similar in design to riding lawnmowers but are significantly more powerful. They are commonly seen delivering luggage to and from passenger aircraft. However, the airport tug task considered here is somewhat different, because the airport itself is used for commercial cargo rather than passenger transportation. In this situation, each tug pulls a train of trailers carrying containers that fit directly into the cargo hold of the aircraft. The tugs move between the planes and the sort facility where the containers are loaded, with a speed between ten and fifteen miles per hour. Each tug follows an unstructured path that may or may not be marked as a lane. Perhaps most importantly, the tugs are faced with many unpredictable obstructions, including containers, human ground crews, other tugs, and aircraft. The tugs operate at night and in a variety of adverse weather conditions, e.g. rain, fog, and snow. The cost of collision is high: any accident causing injury or damage to an aircraft are unacceptable.

Because the cost of collision is so high, it is desirable to have a human operator overseeing the work of robotic tugs. A human operator is desirable in order to:

- Disable an unsafe robot
- Coordinate and schedule the behavior of multiple robots

Proceedings of the 5th *Workshop on Knowledge and Reasoning in Practical Dialogue Systems*, pages 42–45,
Hyderabad, January 8th, 2007.

- Deal with situations beyond the robot's capabilities
- Act as an intermediary between other employees and the robots
- Create new employment opportunities

A useful existing analogy is currently deployed in some supermarkets. U-scan allows supermarket customers to scan and weigh their own items. A single human supervisor oversees 4-6 stations and deals with customer errors and tasks that are currently hard for computers, e.g. visual identification of produce. Likewise, a human dispatcher for autonomous tugs can assist when needed while delegating simple tasks to the tugs.

3 Architecture

The dispatch natural language interface consists of four components. Speech is transformed to text using Sphinx-4 [Walker *et al.*, 2004]. This text is sent to the Jess rule engine [Hill, 2003], where it is transformed into commands. Each robot is represented as a shadow fact in Jess, which allows Jess rules to be used to monitor the robots. Updates from the system are announced using text to speech from ScanSoft and a 3D interface based on the game Unreal Tournament 2004.

3.1 Speech Recognition

Sphinx-4 is an open-source, pluggable Java implementation of earlier Sphinx speech recognition systems [Walker *et al.*, 2004]. It's high modularity and zero-cost make it highly suitable for human-robot interaction research. In our dispatch natural language interface, Sphinx-4 is used in "push to talk" mode for highest fidelity. We are currently working with the airport personnel to determine the actual noise constraints, but for now we are assuming that a headset microphone with "push to talk" interaction will be sufficient.

The two most significant customizable pieces of a Sphinx-4 application are the acoustic model and the language model. Several existing models of each type are freely available for generic speakers on generic tasks. For high performance in customized tasks, it is desirable to create custom models. For acoustic modeling, Sphinx-4 is somewhat disadvantaged by not being trainable online, i.e. with a speaker enrollment period. For this reason, we are hoping that an speaker-independent acoustic model, when used with a customized language model, will yield acceptable results. Sphinx-4 offers several different language model paradigms, including n-grams, JSGF, and FST. Sphinx-4 also allows online swap out of language models. Currently the dispatch interface uses n-grams in a dictation-style interface while we work with airport personnel to establish the space of tasks and entities involved in tug operations (see Section 4). That information will make it possible to populate the system with JSGF representations.

3.2 Jess

Jess, a Java expert system shell, is a rule engine based on CLIPS [Hill, 2003]. Jess can be used for forward and backward chaining and can easily be connected to ontologies. In the dispatch interface, however, Jess is primarily used for scripting. Scripting is used for two different tasks, command interpretation and robot monitoring.

As stated before, spoken commands are rendered as text and input to Jess. Jess is then used to script commands to the robots. The command structure is imperative [Rayner *et al.*, 2000], but may be modified by logical operations on the Jess facts. This allows a parsimonious blending of logic for reasoning about the world with imperative commands for telling the robots what to do next, rather than forcing a single logical or imperative representation. Although Jess provides a number of built-in operators for scripting, it may be significantly enhanced by customizable user functions that may be defined online. These functions may make atomic a sequence of commands or otherwise provide new building blocks for task performance. Thus the framework allows for both high level control, e.g. "Move cargo from loading bay 1 to aircraft bay 3," and low level control, e.g. "Turn right 90 degrees".

Robot monitoring describes the current status of an "assistant agent" that helps the user manage robot performance. In essence, the robot monitor consists of a number of Jess rules that operate over shadow facts representing the robots. In Jess, shadow facts are a useful feature that allow dynamic real world objects to be represented within the Jess fact space. Shadow facts are attribute-value representations that are updated in real time as the corresponding real world object changes. Using shadow fact representations of robots, rules may be constructed that monitor the state of the robot. For example, if a robot is idle for a long period of time, a rule may be triggered that notifies the user. In the future, notifications will be priority managed so that dialogue interruptions occur only when essential, allowing non-essential notifications to be delayed until the user has completed the current task.

3.3 User Notification

Updates from the system are announced using text to speech from ScanSoft and a 3D interface based on the game Unreal Tournament 2004. Text to speech is handled using Microsoft SAPI [Rozak, 1996], which is supported by all commercial speech engines that run on Windows. This is advantageous because it provides plug and play capability for improved text to speech engines that are being released with increasing frequency. Plug and play also allows the user freedom to exercise personal preference a different speech engine, which should increase user satisfaction.

User notification also takes place through a 3D interface based on Unreal Tournament 2004. Unreal Tournament 2004 is a low-cost, high quality commercial game engine that comes with editing and modification tools and documentation. The Gamebots modification allows agents within the game to be controlled by an external program using network protocols [Kaminka *et al.*, 2002]. Gamebots has been further modified to create a realistic robot simulator (USARsim) for human robot interaction research, specifically unmanned search and rescue (USAR) [Wang *et al.*, 2003]. USARsim has advantages with respect to development (see Section 4) and user notification. For notification, USARsim can display multiple points of view (POV) for a robot, including the robot's camera POV a birds-eye POV centered on the robot. Moreover, USARsim provides a drivable camera that may be

Figure 2: USARsim airport/robot simulation

positioned anywhere in the simulation. As a result, the user can develop a true 3D understanding of the robot's situation.

Once robots are deployed in the real world, the USARsim interface will still be used. However in this case, what is shown to the user is what the robot perceives of its environment, i.e. the robot perceives an obstacle at coordinate (x,y,z), so the USARsim 3D world is updated with an obstacle at that coordinate. In this way, USARsim can provide a 3D representation of the robots' beliefs. Because the robots can have inaccurate beliefs, we propose an interface that, for each robot, displays its front camera video together with its USARsim front camera video, such that the user can be made aware of any major discrepancies between the two. These viewpoints, together with the drivable camera mentioned above, will provide valuable spatial information to the human dispatcher.

4 Development Strategy

A defining characteristic for this project is that the multi-robot dispatch interface is being developed in parallel with the robotic tugs. As a result, development is a moving target of requirements. This naturally suggests an iterative development strategy. Therefore the following steps should not be considered as a pipeline, but rather as landmarks in the development landscape that may be revisited:

- Solve task in simulation
- Move to small scale robots and mock-ups
- Move to robotized tugs and mock-up
- Small scale real trials (one dispatcher)
- Large scale real trials (multiple dispatchers)
- Full deployment

Different teams are currently working on the first two items in parallel. The five other teams are focusing on the autonomous tugs themselves, i.e. vision, localization, navigation, obstacle handling, and cognitive robotics. As these teams progress, new capabilities are added to the robots and others change. In order to limit the disruptiveness of such changes to the dispatch interface, we are working with airport personnel to develop a model of idealized tug behavior. This has been quite difficult to accomplish, because there do not appear to be any written training documents for tug operation.

To acquire the necessary information, we are using current tug operations as a model. We are in the early stages of equipping tugs with handheld video cameras, microphones, and GPS devices. These tugs will be driven as part of normal operations by senior drivers. The data thus recorded, together with satellite imagery we have obtained from Google Earth, will provide a wealth of information concerning actual routes, tug-tug interactions, and driver strategies for dealing with moving objects. This data will also make more apparent the degree of autonomy the tugs can be expected to have, and the corresponding situations where human intervention is likely to be required.

This data will be collected, analyzed, and discussed with the other five teams to obtain a consensus view of the robots' eventual capabilities. The dispatch natural language interface will then be tuned to this idealized model so that user testing (in simulation) can begin. User testing on the idealized model serves not only to validate the natural language interface but also serves to inform the basic robot requirements. As an end-to-end test of the system, the interface and idealized simulation may indicate that low level functionality of the robots needs to be modified. For example, users may expect robot functionality not currently implemented. Likewise, the end-to-end test may reveal that some sophisticated robot functionality may not be needed, since the human operator is easily able to intervene. Such findings would re-prioritize robot requirements and consequently affect the interface. Thus an iterative development cycle would begin anew.

5 Conclusion

This paper reports the ongoing development of a natural language interface used to implement multi-robot dispatch. A human dispatcher uses the interface to direct multiple robotic tugs performing cargo loading and unloading tasks at an airport. The airport environment is highly dynamic and hazardous, having human ground crews, tugs, and planes as moving obstacles. Thus human oversight of robotic operations is essential.

The airport tug task lies at the intersection of research streams in task-oriented natural language interfaces and autonomous mobile robots. However, the airport tug task is of sufficient complexity as to provide practical and research challenges for implementation.

This paper has described the task, current architecture, and development strategy. Immediate future efforts will include data analysis of field work described in Section 4 and user testing of the natural language interface using simulated and idealized robots.

Acknowledgments

The author would like to thank FedEx for its support.

References

[Allen *et al.*, 1995] James F. Allen, Lenhart K. Schubert, George Ferguson, Peter Heeman, Chung Hee Hwang, Tsuneaki Kato, Marc Light, Nathaniel G. Martin, Bradford W. Miller, Massimo Poesio, and David R. Traum. The TRAINS project: A case study in building a conversational planning agent. *Journal of Experimental and Theoretical AI*, 7:7–48, 1995.

[Allen *et al.*, 2000] James Allen, Donna Byron, Myroslava Dzikovska, George Ferguson, Lucian Galescu, and Amanda Stent. An architecture for a generic dialogue shell. *Natural Language Engineering*, 6, 2000.

[Hill, 2003] Ernest Friedman Hill. *Jess in Action: Java Rule-Based Systems*. Manning Publications Co., Greenwich, CT, USA, 2003.

[Kaminka *et al.*, 2002] Gal A. Kaminka, Manuela M. Veloso, Steve Schaffer, Chris Sollitto, Rogelio Adobbati, Andrew N. Marshall, Andrew Scholer, and Sheila Tejada. Gamebots: a flexible test bed for multiagent team research. *Commun. ACM*, 45(1):43–45, 2002.

[Lemon *et al.*, 2003] Oliver Lemon, Anne Bracy, Alexander Gruenstein, and Stanley Peters. An information state approach in a multi-modal dialogue system for human-robot conversation. In Hans Reiser, Peter Kunlein, and Henk Zeevat, editors, *Perspectives on Dialogue in the new Millennium*, volume 114 of *Pragmatics and Beyond*, pages 229–242. John Benjamins, 2003.

[Montemerlo *et al.*, 2006] Michael Montemerlo, Sebastian Thrun, Hendrik Dahlkamp, David Stavens, and Sven Strohband. Winning the darpa grand challenge with an ai robot. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, Boston, 2006. AAAI.

[Rayner *et al.*, 2000] Manny Rayner, Beth Ann Hockey, and Frankie James. A compact architecture for dialogue management based on scripts and meta-outputs. In *Proceedings of the sixth conference on Applied natural language processing*, pages 112–118, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

[Rozak, 1996] Mike Rozak. Talk to your computer and have it answer back with the microsoft speech api. Microsoft systems journal, Microsoft, January 1996.

[Stent *et al.*, 1999] Amanda Stent, John Dowding, Jean Mark Gawron, Elizabeth Owen Bratt, and Robert Moore. The commandtalk spoken dialogue system. In *Proceedings of the Thirty-Seventh Annual Meeting of the ACL*, pages 183–190, University of Maryland, College Park, MD, 1999. Association for Computational Linguistics.

[Walker *et al.*, 2004] Willie Walker, Paul Lamere, Philip Kwok, Bhiksha Raj, Rita Singh, Evandro Gouvea, Peter Wolf, and Joe Woelfel. Sphinx-4: A flexible open source framework for speech recognition. Technical Report TR-2004-139, Sun Microsystems, Inc., November 2004.

[Wang *et al.*, 2003] Jijun Wang, Michael Lewis, and Jeffery Gennari. Usar: A game-based simulation for teleoperation. In *Proceedings of the 47th Annual Meeting of the Human Factors and Ergonomics Society*, pages 493–497, Denver, CO, Oct. 13-17 2003.

PCQL: A Formalism for Human-Like Preference Dialogues*

Pontus Wärnestål, Lars Degerstedt and Arne Jönsson
Department of Computer and Information Science
Linköping University, Sweden
{ponjo, larde, arnjo}@ida.liu.se

Abstract

Preference dialogues display several interesting characteristics that have implications on how to design human-like dialogue strategies in conversational recommender systems. Using human-human preference dialogues as an empirical base, this paper introduces a novel data manipulation language called PCQL that comprises explicit descriptive, comparative and superlative preference management as well as implicit preference statements such as factual information queries. The usage of the PCQL language is demonstrated by an implementation of a music conversational recommender system.

1 Introduction

Adaptive dialogue systems using talking heads and other human attributes are moving from being tool-like towards being regarded as human-like [Qvarfordt *et al.*, 2003]. Such dialogue systems become conversational partners and users require more elaborate interaction capabilities, e.g. expressing vagueness and preferences.

This paper presents a *data manipulation language* (or *query language*) called PCQL[1] for representing both preferential and factual statements and queries. It is designed for modeling human-like dialogue in preference-aware systems, such as conversational recommender systems. It is intended to be used as the message-passing format in and out of an agent's dialogue manager. The notation of PCQL has been tailored for the specific needs of a dialogue agent expressing factual and preferential state changes in an effective way (cf. [Bentley, 1986]).

[Carberry *et al.*, 1999] provide a basic framework for capturing user preferences of varying strength in natural language dialogue. Carberry *et al.*'s model focuses on descriptive preferences (e.g. U2 in Figure 1). Our approach to human-like preference dialogue is based on Carberry *et al.*'s work, but extended so that our model accommodates (a) preferential statements and queries (including descriptives, comparatives and superlatives), and (b) factual statements and queries. This paper discusses a new formalism that can be used to capture these aspects as part of a coherent dialogue model and strategy.

The presented notation is novel in that it treats preference using logic-style operators on top of the factual level, similar to how modal and intentional logic separates fact from belief. In this way, PCQL allows for a flexible mixing of the factual and preference level that can be handled in a systematic way by the dialogue manager.

PCQL is influenced by how formulas of modal and intentional logic keep a clean separation between the factual level and the belief of the agents, but at the same time allows for mixing levels freely in compound formulas (cf. [Hintikka, 1962; Kripke, 1963]).

The feasability of the expressiveness of the PCQL formalism has been validated in a functional implementation of a recommender system and an implementation based on an extension of BNF of the PCQL notation exists[2].

The paper discusses the (abstract) syntax and expressiveness of PCQL and is structured as follows: First, some important features of human-like preference dialogues are identified, in the context of recommendations. Second, we present the PCQL notation and detail the constituents of the formalism and describe how preference dialogues are encoded. Then, we give an overview of how PCQL can be used in a system design. As an example we present a conversational music recommender system called ACORNSONG which has been used to test the expressive feasability of PCQL.

2 Human-Like Preference Dialogue

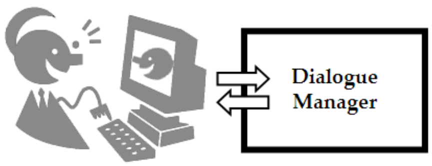
A system of categories describing preference dialogues needs to consider basic dialogue strategies, distribution of dialogue acts, and the way domain entities are referred to. As an empirical base for characterising preference dialogue we have previously collected a human-human dialogue corpus in the movie recommendation domain [Johansson, 2003]. The study's aim is to shed light over preference dialogues as they

*The support of the Swedish National Graduate School of Language Technology (GSLT) and Santa Anna IT Research Institute AB is acknowledged.

[1] Preference Conversational Query Language.

[2] PCQL exists as an open source project, available at http://herd.ida.liu.se/nlpfarm/.

Proceedings of the 5th *Workshop on Knowledge and Reasoning in Practical Dialogue Systems*, pages 46–54,
Hyderabad, January 8th, 2007.



S1a	Welcome.
S1b	What situation should we discuss?
U1	Work
S2a	Ok.
S2b	Are there any genres, artists or albums you prefer at work?
U2	I like *rock*
S3a	These artists and albums belong to the genre *rock*: `[list]`
S3b	Any of these you like in particular?
U3	What *alternative rock* artists are there?
S4a	These artists are in the genre *alternative rock*: `[list]`
S4b	Any of these you like in particular?
U4	*Creed* is the best
S5a	These genres and albums belong to the artist *Creed*. `[list]`
S5b	Any of these you like in particular?
U5	I might like *Human Clay* better than *Weathered*
S6a	Ok.
S6b	Are there any other genres, artists or albums you prefer at work?
U6	What *electronic* albums are there?
S7a	These albums are in the genre *electronic*. `[list]`
S7b	Any of these you like in particular?
U7	I don't like *Moby's* albums released before *Play*
S8a	Ok.
S8b	Are there any other genres, artists or albums you prefer at work?
U8	Artist is more important than album.
S9a	I think you will like the song *Just One* by *Hoobastank*.
S9b	Have you heard it?

Figure 1: Sample dialogue excerpt from the ACORNSONG system with factual statements and queries; and descriptive, comparative and superlative preferences. S = system, U = user.

occur in recommendation situations—where it is natural to utilise preferences.

Recommender systems aim at assisting users in searching, sorting, and selecting items from large information repositories. They differ from traditional information search engines in that they provide *personalised* service. The key issue for personalisation is to correctly—and efficiently—construct, update and utilise individual records of user preferences. The problem of how to achieve this is an active research area in the user modeling and recommender system communities. Most traditional recommender systems rely on scalar ratings of domain items (such as movies, books, web pages etc.) in the range of e.g. 1–5 and require users to explicitly rate each item by point-and-click interaction. An alternative and promising way to efficiently and effectively capture user preferences is to use natural language collaborative dialogue [Carenini *et al.*, 2003]; that is, much like how two friends would approach the task of recommending items to each other. This is the approach adopted in this work.

We start by defining a *preference dialogue* as an exchange of dialogue acts between two participants; one acting in a *recommender* role, and the other in a *customer* role (i.e. receiver of recommendations). The recommender is assumed to have extensive domain knowledge (such as access to a database of domain items), as well as a strategy for getting to know the customer's preferences, and a way of using this information in order to recommend relevant items. In a human-machine situation this translates naturally to the *system* having the recommender role, and the *user* having the customer role.

Looking at the overall dialogue flow in a typical preference dialogue, we can distinguish three phases:

1. Establishing initial descriptive preferences
2. Free exploration by query, and additional preference acquisition
3. Refinement of preferences using comparatives and superlatives

In phase 1, the recommender (or system) aims at establishing some basic preferences, preferably distributed over the majority of the domain's entity types (e.g. some preferred artists, some genres, and some album preferences in the music domain). Here, the initiative is mostly the recommender's who is guiding the user to efficiently acquire preferences through direct questions.

The customer (or user) may then, in phase 2, take initiative and explore the domain by posing factual questions about the domain. In the dialogue corpus it is common that preference statements occur as a result of being exposed to query results. This is consistent with the observations of e.g. [Carberry *et al.*, 1999, p. 187] who claim: "...users are often unaware of their preferences at the outset of planning and only bring these preferences into play as they must evaluate alternative actions and choose among them."

When an initial set of preferences have been accumulated, preferences may be refined by introducing comparative statements in phase 3 (e.g. utterance U5 as response to S5a/S5b in Figure 1). Initiative in the third phase is not as clear-cut as in the previous two. The corpus indicates that about half of the recommenders re-gained more control over initiatives in phase 3 and asked customers comparative questions. The other half simply acknowledged comparative preferences as they were stated by customers. For dialogue system strategy design, this behaviour is thus an open choice. Both approaches are "human-like" using the human-human dialogue corpus as guideline.

The phases are not one-directional since they may overlap each other to a certain extent in the dialogue. Each phase may also occur several times in a longer dialogue. Furthermore, all phases are not mandatory in all preference dialogues (e.g. there may be dialogues without much exploration by query). The three phases serve as useful guidelines when designing a dialogue strategy that describe human-like preference dialogue behaviour.

One observation on preference dialogues is that humans prefer to start out simple and then gradually refine factual queries/statements and preference statements in the on-going dialogue as opposed to construct complex utterances in one go. This should thus be supported in the dialogue strategy design.

When examining the preference dialogue corpus at utterance level, it was found that 50.7% of the customer utterances in preference dialogues were descriptive, comparative or superlative *preference statements*. A smaller part, 28.6%, of the utterances were *factual questions* about the domain and its entities. Preference statements and factual queries and responses are considered the principal *task-related* utterances in preference dialogues. The remaining part consisted of communication management such as repeats, and sub-dialogue clarifications (14.5%), and irrelevant utterances (e.g. questions concerning the experiment situation) (6.2%). According to the model presented by [Carberry *et al.*, 1999], there are three utterance types in which preferences are conveyed: DIRECT (e.g. *I like Bruce Willis*), INDIRECT (e.g. as part of queries; *What thrillers are there?*), and HEDGING, which signals an uncertain preference (e.g. *I might like Pulp Fiction*). Direct statements and hedgings falls into the descriptive category, whereas indirect statements belongs to factual information queries. Carberry *et al.* focus on descriptive preferences and do not mention comparatives and superlatives in their model. However, we feel they should naturally be included in the direct preference statement class.

In addition, there are four *conversational circumstances* in which preference elicitation occurs according to Carberry *et al.*: REJECT-SOLUTION, VOLUNTEERED-BACKGROUND, VOLUNTEERED, and QUESTION-AND-ANSWER. By combining utterance type and conversational circumstance we arrive at a specific negative or positive preference strength. For example, a direct preference statement in a reject-solution situation is the strongest (negative) preference (-6); whereas a positive indirect preference in a question-and-answer situation is moderate (3). See [Carberry *et al.*, 1999] for a more detailed account.

Task-related utterances in this dialogue genre can be viewed in terms of traditional dialogue acts such as statements and info-requests [Bunt, 1994]. As hinted above, the division between factual and preferential acts is important and serves as a useful tool to categorise acts specific for the preference dialogue. In order to arrive at a design of a formalism specifically targeted for preference dialogue and in particular recommendation situations we identify the following acts:

Factual-Question Requests take two distinct shapes in preference dialogues. In the first sense, it is a question of factual nature (typically from the customer's part) about the domain. This is the info-request in the traditional information-providing dialogue system sense, where the system's task is to deliver a database result (as a FACTUAL-STATEMENT).

Preference-Question In the second sense, the request is a preferential question from the recommender to the customer, where the goal is to acquire preferences as an answer from the customer. These PREFERENCE-QUESTIONS are mostly descriptive, but occur as comparatives or superlatives in some dialogues.

Answer As in the case of QUESTIONS there are both factual and preferential ANSWERS. These are responses from the customer to PREFERENCE-QUESTIONS from the recommender. Answering is an abstract act that can take several shapes: FACTUAL-STATEMENT, PREFERENCE-STATEMENT, and the simple YES/NO answer. Factuals as answer is most common for the recommender and PREFERENCE-STATEMENT is mostly a customer act. YES/NO answers exist for both roles.

Factual-Statement The FACTUAL-STATEMENT is a fundamental characteristic of information-providing dialogue and is the standard response to a factual request. Providing an answer from a database or other domain description is often the task of the recommender system.

Preference-Statement Comparative PREFERENCE-STATEMENTS naturally refer to two entity types or entity values (arity 2), whereas descriptive and superlative statements refer to one entity type or value (arity 1). Naturally, this act is reserved for the customer in the studied recommendation situations. However, it does occur that human recommenders provide *their* preferences as statements, e.g. before providing a recommendation. This special case is not very common, and is probably unsuitable for human-computer dialogues. The reason PREFERENCE-STATEMENT is separate from the ANSWER category is that PREFERENCE-STATEMENTS also occur as volunteerings, i.e. without a preceding PREFERENCE-QUESTION.

Agreement Some schemes contain agreements (ACCEPT and REJECT) as backward-looking functions. These two are common in this domain as ANSWERS to RECOMMENDATIONS. The REJECT act is viewed as a NO combined with a PREFERENCE-STATEMENT (e.g. *"No. I don't like thrillers"*). The ACCEPT act is a YES or ACKNOWLEDGEMENT, optionally combined with a PREFERENCE-STATEMENT.

Recommendation The recommendation act is central to preference dialogues in recommendation situations, and is the goal of a recommender system. A RECOMMENDATION is invoked when the recommender has gathered "enough" preferences from the customer in order to present an entity that she believes the customer will like. However, RECOMMENDATION is an abstract act, since it can be realised as a QUESTION (*"Have you seen film x?"*), as a STATEMENT (*"You will probably like film x"*), or even as a combination of the two (*"Have you seen film x? I think you will like it"*).

The characteristics outlined in this section provides an empirical base for developing a formal system that describes preference dialogues.

3 PCQL

PCQL is a formalism that consists of **action statements** that represent dialogue act specifications of preference dialogues. PCQL action statements are used both for representation of user and system acts and treat questions and statements in a symmetric way, both in and out of the system. The formalism is intended to be used as a message passing format for the dialogue manager module in a dialogue agent (see Figure 2)[3].

Since PCQL is a conversational formalism, the PCQL action statements have a double function. On the one hand, each statement *describes* some aspects of the factual and preference state (the FP **state**) of the dialogue system. On the other hand, each PCQL action statement expresses an *action* performed by the dialogue participant, a dialogue act, where the acting agent is doing something that will result in a response from the dialogue partner. The description is subordinate to the dialogue action, but the latter requires the first to be fully understood. In that sense, the descriptive expression is a parameter to the stated action.

3.1 FP State Formulas

The expressions of PCQL that are used to describe (aspects of) the FP state are called FP **state formulas**. In this section, we define the syntax of this formalism[4].

The FP state formulas express relations and entities of the domain that are in focus in the dialogue. The basic constituents of this language are constraints over **entity types** and **entity values**. The entity types are predefined types of possible entity values, such as Genre, which can be enumerations of known entities or open domains such as "any string". The entity values are either atomic domain entities—such as Electronic— or sets/intervals of entity values—such as $\{$Rock, Electronic$\}$ and [1989..1999].

The constraints can be formed using the factual operators shown in Table 1. A special entity type is YN consisting of the values Yes and No. References to entities through other entities (relations or attributes) are handled with two constructs. The first is to use the π operator to mark entity types whose values are inferred from the other constraints in a formula. For example, *"Albums of The Beatles and Deep Purple"* can be described as π Album, Artist $\in$ $\{$The Beatles, Deep Purple$\}$. Informally, we may read this as follows: Artist $\in$ $\{$The Beatles, Deep Purple$\}$ specifies a set of entities (in this case two); π Album projects this set of entities on the albums (in this case all albums by either *The Beatles* or *Deep Purple*). The second construct is that entity values can indirectly be referred to as attributes of other entity values using dot notation on the entity type names, for example My Song.Album denotes the album of which the song My Song belongs.

Factual	Name	Arity	Meaning
$\top/\bot$	max/min	1	*newest/oldest*
π	projection	1	*entity reference*
$=/\neq$	(not) equals	2	*is/ is not*
$</>$	comparison	2	*newer/older*
$\in/\notin$	(not) member	2	*one of/ not one of*
Preference			
$\odot$	Indifferent	1/2	*doesn't matter*
$\oplus/\ominus$	Descriptive	1	*good/bad*
$\boxplus/\boxminus$	Superlative	1	*the best/the worst*
$\rhd/\lhd$	Comparative	2	*better/worse*

Table 1: Factual and preference operators of the FP state formulas. The factual operators are used to form unary and binary constraint expressions. The Preference operators are used on the factual constraints to formulate: Descriptive, comparative, and superlative ratings' polarities are either positive or negative. Please note that hedges ($\diamond$) can be combined with descriptive, superlative, and comparative preference operators.

We form constraints from atomic entity types and entities, and by augmenting atomic constraints with factual operators (see Table 2 for examples). From the factual constraints, we form conjunctive formulas, called **factual** FP state formulas, or simply **F state formulas**, where comma is used as conjunction sign. Intuitively, the meaning of the F state formulas can be read as specifications of sets of entities. The unary operators are really aggregate operators on such sets, where the aggregate is given implicitly by the remaining formula[5].

Given the set of F state formulas, we form atomic **preference formulas** by augmenting the preference operators shown in Table 1. It is not allowed to nest the preference operators in the same atomic preference formula (since this would increase the complexity of the language without being useful). From the F state formulas and the atomic preference formulas, we form conjunctive formulas using comma as the conjunction sign. Furthermore, each preference operator may be indexed with a *hedging* symbol ($\diamond$), that indicates uncertainty about the preference [Carberry *et al.*, 1999]. The intuitive reading of the preference formulas are as statements of

[3] In this article, we consider mainly utterances that include full descriptions of entities and preferences. However, it is straightforward to capture also more fragmentary user utterances—such as "better", "more"—by allowing fragments of PCQL action statements.

[4] We use an abstract syntax notation for the FP state formulas. A concrete syntax exists, but it is less readable in articles. For example: $\boxplus$ corresponds to $++$, and $\rhd$ corresponds to $>>$.

[5] The Max/Min operators have higher priority than projection, in formulas where both occur.

like and *dislike* of the sets of entities described by the factual part of the formula.

Finally, the factual and preference operator symbols form two **operator types**, denoted by $\circ$ and $\circledcirc$ respectively. The type symbols $\circ$ and $\circledcirc$ can be used in any formula in place of an operator to express uncertainty or requests concerning the operator's position. For example, the sentence *"Is Bob Dylan an artist or not?"* can be described using $\circ$, as the FP state formula $(\mathsf{Artist} \circ \mathsf{Bob\ Dylan})$. Similarly, the preference statement *"Is The Beatles better or worse than Deep Purple?"* can be described using $\circledcirc$, as the FP formula $(\mathsf{Artist} = \mathsf{The\ Beatles}) \circledcirc (\mathsf{Artist} = \mathsf{Deep\ Purple})$.

This forms the complete FP state formula language, for which various examples can be found in Tables 3 and 4. The format of the FP state formulas is influenced by how formulas of modal (and intentional) logic keep a clean separation between the factual level and the belief of the agents, but at the same time allows for mixing levels freely in compound formulas.

An FP state formula describes a conjunctive aspect of the total FP state that is relevant for a particular dialogue act. We say that each FP state formula expresses an FP **state mapping** from the dialogue act to some entities of the FP state that are in focus.

Factual State Mapping

The F state formulas (with only factual constraints) deal with information-providing aspects of the system state. We distinguish between F state formulas that concern explicitly stated entities and those that are indirectly referenced using the projection operation (π).

Table 2 shows the identified classes of factual descriptions in dialogue acts we have found from our examined material, as discussed in Section 2.

In the explicit factual FP state formulas, entities are referred to by their name (in the system). In explicit aggregates and relative statements, it is the aggregate or relative value that is explicit. For example, in *"most popular in the 70s"* the aggregate set *"the 70s"* is explicitly mentioned.

In the referential factual FP state formulas, entities are referred indirectly through properties or relations that specify them. This means that the formula must also specify of what type the referred entity is. Referential formulas are most obviously occuring in questions, but may also occur in informative statements. In particular, they may be part of the informative part of user preference utterances.

Preference State Mapping

Preferential user utterances are built "around" F state formulas, using the preference operators.

Descriptive and superlative statements are syntactically handled in the same way in FP state formula mapping schemes, as shown in Table 3. Both types of constructs amount to similar 1-arity formulas. However, observe that the meaning of superlatives is a form of aggregate functions operating on sets, which is more complex than the descriptive case. Since these aggregates are given implicitly by the context, this complexity is hidden from the formula. For example, the FP state mapping of the sentence *"The Beatles is the best artist in the Genre Pop"* can be described by the FP state formula $\boxplus(\mathsf{Artist} = \mathsf{The\ Beatles}), (\mathsf{Genre} = \mathsf{Pop})$[6]. Most factual constructs make sense as part of a preference statement. The constructs that make little sense are: explicit and referential negation, and Yes/No. In real dialogue, some of the listed utterances are less important than others. However, recall that we want to be able to use PCQL *after* contextual interpretation. In some cases this means that the FP state formula at hand actually contains the collected information of a whole sub-dialogue. In a collected formula, more complicated constructs may have been gathered over time. Thus, PCQL covers both the collected formulas and the simpler ones in a natural way.

Compound FP is a "new" type of formula that occur only on the preference level. This class contains utterances that separately combines one part that is expressing a preference with one part that is factual (see Table 3).

Comparative utterances are 2-arity constructs, and are handled differently than the 1-arity preference formulas. Table 4 shows how the factual classes are handled by FP state formulas in comparative preference contexts using infix notation.

3.2 PCQL Action Statements

When we use PCQL to model dialogue acts we attach **action tags** to FP state formulas. An action tag is a domain or applications-specific dialog action category that accepts specific FP state formulas as valid arguments. We have identified three basic generic types of action tags for PCQL:

- **inform type** (I-TAGS): Actions that inform the other party of something.
- **ask type** (A-TAGS): Actions that ask the other party something.
- **conventional type** (C-TAGS): Ritualized actions such as greeting, thanking, etc.

Each inform type action tag is used to assert facts, give answers, preferences and/or values. An I-TAG accepts one or two arguments, where the (optional) second argument is a collection of values (e.g. a database result set). The syntax of an **inform type action statement** is[7]:

$$\langle \text{I-TAG} \rangle \ [\![\ \langle fp \rangle \ ]\!] \ \{ \ \text{VALUES} \ [\![\ \langle vlist \rangle \ ]\!] \ \}^?$$

where $\langle$I-TAG$\rangle$ is an I-TAG, $\langle fp \rangle$ is an FP state formula and $\langle vlist \rangle$ an attribute-value map from entity types to entity values.

The ask type action tags are used to ask preferential and factual questions. The syntax of an **ask type action statement** is:

$$\langle \text{A-TAG} \rangle \ [\![\ \langle fp \rangle \ ]\!]$$

where $\langle$A-TAG$\rangle$ is an A-TAG and $\langle fp \rangle$ is a FP state formula. The formula $\langle fp \rangle$ is here interpreted as a question, or request, for information. The operator $\circledcirc$ can used to request type of preference. For factual requests, projection π and aggregates

[6]Note that the FP state formula does not say anything about what dialogue act is performed, which in this case is a PREFERENCE-STATEMENT

[7]In the syntax-definitions we use the meta-notation $\{\ldots\}^?$ with meaning "zero or one occurrence", and $\langle x \rangle$ for syntactic meta-variable x.

Factual: Explicit	**Utterance**	**FP State Formula**
Entity Type	What is genre?	Genre
	One of genre, artist and album	Genre, Artist, Album
Entity	Techno	$\text{Genre} = \text{Techno}$
Enumeration	Both Dylan and Waits	$\text{Artist} \in \{\text{Dylan, Waits}\}$
Yes/No	Yes	$\text{YN} = \text{Yes}$
Negation	Not Dylan	$\text{Artist} \neq \text{Dylan}$
Interval	Album three to five	$(\text{AlbumNo} \in [3..5])$
Relative	Newer than 1975	$(\text{Year} > 1975)$
Aggregate	The latest	$\top$ Year
Aggregate	Most sold album of the 70's	$\top$ SoldCopies, $(\text{Year} \in [1970..1979])$
Factual: Referential		
Entity	An album by Dylan	π Album, $\text{Artist} = \text{Dylan}$
Enumeration	Albums by either Dylan or Waits	π Album, $\text{Artist} \in \{\text{Dylan, Waits}\}$
Negation	All albums except The Beatles'	π Album, $\text{Artist} \neq \text{The Beatles}$
Interval	Songs from the 70s	π Song, $\text{Year} \in [1970..1979]$
Relative	Albums older than Weathered	π Album, $(\text{Year} < \text{Weathered.Year})$
Aggregate	The first of Dylans' albums	π Album, $\perp$ Year, $(\text{Artist} = \text{Dylan})$

Table 2: FP State Formula Mappings for factual utterance types. The table shows by prototypical examples how expressions of factual state in utterances correspond to FP state formulas.

are normally used. However, any formula can be seen as an implicit question, which may warrant the addition of projections to all kinds of formulas. For example, the FP state formula $\oplus\, (\pi\ \text{Album}, (\text{Artist} = \text{Dylan}))$ can be seen as the implicit yes-no question *"Do you like (all) albums of Bob Dylan?"* which can be made explicit by adding of π YN.

Similarly, conventional type tags express conventional actions. These statements accept one (possibly empty) argument. The syntax of an **conventional type action statement** is:

$$\langle\text{C-TAG}\rangle \llbracket \langle fp \rangle \rrbracket$$

where $\langle$C-TAG$\rangle$ is a C-TAG and $\langle fp \rangle$ is an FP state formula. For example, the C-TAG GREET could be implemented as an empty-argument action to represent the utterance *"Hello"*, but it could also accept an FP state argument such as: $\text{GREET}\llbracket\, (\text{Name} = \text{Tom})\, \rrbracket$ to represent *"Hello Tom"*.

Each dialogue act may correspond to a PCQL action tag. The complete PCQL action statement (action tag and FP state formula) expresses the PCQL **action mapping** that specifies the dialogue act performed by the agent. Table 5 shows some of the possible mappings for the identified dialogue act types discussed in Section 2. In these examples the focus is on the structure of the dialogue act and action tag. Therefore, only simple FP state descriptions are used, but any of the previously discussed mappings can be used here as well.

4 Using PCQL in Conversational Recommender Systems

This section describes the ACORNSONG design and implementation, which is a conversational music recommender system that uses PCQL. ACORNSONG's goal is to construct and maintain a list of songs ranked based on the user's preferences (as detected in dialogue).

4.1 Architecture

In general, a conversational recommender system implementing the recommender role in a human-like fashion (see Section 2) needs an **information repository** (typically a relational database describing the domain), a **dialogue strategy** for asking the user for her preferences in an efficient way as well as responding to user queries, a **preference model** for representing and storing user preferences, and a **recommendation algorithm** for predicting how well each domain item fits the user's preferences. A complete system also needs a parser that interprets spoken or written natural language and creates PCQL statements. A natural language generator for the surface realisation of outgoing PCQL is also needed. In this paper we focus on the internal workings of the dialogue manager, which takes PCQL as its input (generated by a natural language parser), and returns PCQL as output to a generation module. Figure 2 shows a schematic of the involved components in the dialogue manager.

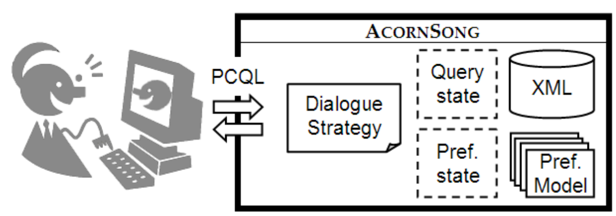


Figure 2: ACORNSONG's dialogue manager sub-modules. The preference state is modelled in preflets. Preflets are data structures used in ACORNSONG that cater for domain preference variation based on FP state formula mappings.

The information repository in ACORNSONG is an XML file that describes the available song files with genre, artist, and album information. The current version of ACORNSONG contains 7800 songs, distributed over 45 genres, 1809 artists, and 619 albums.

There are several potential influences on a system's dialogue strategy, such as speech recognition accuracy in a spoken dialogue system, as noted by [Singh *et al.*, 2002], conver-

1-Arity Preference	Utterance	FP State Formula
Explicit Entity Type	The artist does not matter	$\odot$ Artist
	Genre and artist are important, but not album	$\oplus$ (Artist, Genre), $\ominus$ Album
	Artist is most important	$\boxplus$ Artist
Explicit Entity	I like The Beatles	$\oplus$ (Artist = The Beatles)
	Techno is not good	$\ominus$ (Genre = Techno)
	Dylan is the best artist	$\boxplus$ (Artist = Dylan)
Explicit Enumeration	I like Dylan, The Beatles and Deep Purple	$\oplus$ (Artist $\in$ {Dylan, The Beatles, Deep Purple})
	I like Dylan and Waits best	$\boxplus$(Artist $\in$ {Dylan, Waits})
Explicit Interval	I like Album three to five	$\oplus$ AlbumNo $\in$ [3..5]
	I like Album three to five best	$\boxplus$ AlbumNo $\in$ [3..5]
Explicit Relative	I like everyting older than 1975	$\oplus$ (Year $<$ 1975)
	I like everything older than 1975 best	$\boxplus$ (Year $<$ 1975)
Explicit Aggregate	I like the most sold albums from the 70's	$\oplus$ ($\top$ SoldCopies, (Year $\in$ [1970..1979]))
Referential Entity	I like all of Dylans' album	$\oplus$ (π Album, (Artist = Dylan))
Referential Enumeration	I like songs of Creed and Bush	$\oplus$ (π Song, (Artist $\in$ {Creed, Bush}))
	I like songs of Creed and Bush best	$\boxplus$ (π Song, (Artist $\in$ {Creed, Bush}))
Referential Interval	I like songs from the 60's	$\oplus$ (π Song, (Year $\in$ [1960..1969]))
	I like songs from the 60's best	$\boxplus$ (π Song, (Year $\in$ [1960..1969]))
Referential Relative	I like all Moby's albums before Play	$\oplus$ (π Album, (Artist = Moby), (Year $<$ Play.Year))
	My favorites are all Moby's albums before Play	$\boxplus$ (π Album, (Artist = Moby), (Year $<$ Play.Year))
Referential Aggregate	I like Dylan's latest album	$\ominus$ (π Album, $\bot$ Year, (Artist = Dylan))
	Dylan's latest album is the worst	$\boxminus$ (π Album, $\bot$ Year, (Artist = Dylan))
Compound FP	I like Elvis when I am working	$\oplus$ (Artist = Elvis), (Situation = Work)
	Elvis is the best when I am working	$\boxplus$ (Artist = Elvis), (Situation = Work)

Table 3: FP state formula mappings for descriptive and superlative preference utterances.

sational history, back-end database status, etc. In this work, we focus on the user preference model's influence on the dialogue. To allow for basic mixed-initiative dialogue we have chosen a frame-based approach [Allen *et al.*, 2000] since its design is simple but allows for reasonably free user interaction. Following the guidelines derived from section 2 the system's strategy is designed to (a) interview the user about her preferences (phase 1), (b) respond to user queries (phase 2). Phase 3 is implicit in the current version of ACORNSONG; that is, the system records comparatives and superlatives but does not explicitly ask the user for such preference statements.

Users may choose to volunteer preferences (PREFERENCE-STATEMENTS realised as I-TAG actions called INFORM), such as utterance U2 and U5 in Figure 1. Users may also take initiative to query the system (FACTUAL-QUESTIONS realised as ASK actions) such as utterance U3 as a response to S3b in Figure 1). There are two conventional type action tags available for both user and system implemented in ACORNSONG: GREET and BYE (both with empty arguments). Table 5 lists examples of these action tags. In addition, there are system-specific I- and A-TAGs that are required in preference and recommendation dialogue as described in Section 2. Examples of these include the I-TAG MOTIVATE that informs the user *why* a certain recommendation was given (e.g. utterance S9a in Figure 1); the A-TAG NEWREC that asks if the user wants a new recommendation; and the A-TAG HEARDSONG that asks if the user has heard a particular song (e.g. utterance S9b).

The dialogue manager's strategy examines the *preference state* as well as the *query state* (see the dialogue manager submodules in Figure 2) to construct a turn. A system turn typically consists of one feedback act and one initiative act. In the case of *factual* user queries the feedback act consists of reporting the result of the database query (INFORM), and the initiative act consists of a preference question (ASK) that encourages the user to provide preferences about the discussed domain entities. In the case of *preference* statements, the system fetches relevant information about the topic of the statement (such as S3a in Figure 1), and then encourages the user to provide more preferences.

Consider the example dialogue in Figure 1: Utterance S2b is the initiative act asking the user for preferences in an open-ended fashion. The (compound) FP state formula for 2b is: $\oplus$ (Value $\in$ {Genre, Artist, Album}), (Situation = Work) and the action is ASK.

After each turn, the dialogue manager queries the preference state of its status and this influences the choice of feedback and initiative acts. If there is no preference data available the system starts phase 1 by producing an ASK action with a situation type as parameter in order to define a situation for the upcoming preference dialogue (S1b in Figure 1). If there are not enough preferences in the model to produce recommendations the system needs to encourage the user to provide more preferences. One basic strategy to do this is to find an entity type the user has not yet provided any preferences for and the user with an ASK action, such as *"Are there any albums that you like?"* if Album is an entity type with no attached preference values. The ASK action can be preceded by an I-TAG feedback action that explains that more preferences are needed. The dialogue in Figure 1 shows more examples of surface realisation of this strategy.

2-Arity Preference	Utterance	FP State Formula
Explicit Entity Type	Artist is more important than Album	Artist $\rhd$ Album
Explicit Entity	The Black album is better than the Red album	(Album = Black) $\rhd$ (Album = Red)
	I prefer techno to songs by Scooter	(Genre = Techno) $\rhd$ (Artist = Scooter)
Explicit Enumeration	I like Dylan and Wait better than The Beatles	(Artist $\in$ {Dylan, Waits}) $\rhd$ (Artist = The Beatles)
Explicit Interval	I like Album three to five better than the others	AlbumNo $\in$ [3..5] $\rhd$ AlbumNo $\notin$ [3..5]
Explicit Relative	I prefer newer than 1975 over older	(Year $>$ 1974) $\rhd$ (Year $<$ 1975)
Explicit Aggregate	I like the most sold from the 70's better than rock	($\top$ SoldCopies, (Year $\in$ [1970..1979])) $\rhd$ (Genre = Rock)
Referential Entity	I like Dylan's genre better than Scooter's	(π Genre, (Artist = Dylan)) $\rhd$ (π Genre, (Artist = Scooter))
Referential Interval	I like songs from the 90's better than classical	(π Song, (Year $\in$ [1990..1999])) $\rhd$ (π Song, (Genre = C))
Referential Enumeration	I like albums by D or W better than B	(π Album, (Artist $\in$ {D, W})) $\rhd$ (π Album, (Artist = B))
Referential Relative	I like all S's albums before T better than Dylan	(π Album, (Artist = S), (Year $<$ T.Year)) $\rhd$ (Artist = D)
Referential Aggregate	I like Dylan's latest album better than Creed	(π Album, $\perp$ Year, (Artist = Dylan)) $\rhd$ (Artist = Creed)
Compound FP	I like Bush better than Moby when I am working	(Artist = Bush) $\rhd$ (Artist = Moby), (Situation = Work)

Table 4: FP State Formula Mappings for 2-arity comparatives.

Act	Utterance	PCQL Action Statement
Factual Question	What electronic albums are there	ASK $[\![$ π Album, (Genre = Electronic) $]\!]$
Preference Question	Is Moby better or worse than Creed?	ASK $[\![$ (Artist = Moby) $\circledcirc$ (Artist = Creed) $]\!]$
	Which artists are better than Metallica?	ASK $[\![$ (π Artist) $\rhd$ (Artist = Metallica) $]\!]$
	What do you think about techno?	ASK $[\![$ $\circledcirc$ Genre = Techno $]\!]$
	Which song do you like best on album Weathered?	ASK $[\![$ $\oplus$ (π Song, (Album = Weathered)) $]\!]$
	Which genres or artists do you prefer?	ASK $[\![$ $\oplus$ (Value $\in$ {Genre, Artist}) $]\!]$
Answer	I like techno but I don't like Moby	INFORM $[\![$ $\oplus$ (Genre = Techno), $\ominus$ (Artist = Moby) $]\!]$
Factual Statement	These artists belong to the genre rock: `[X,Y,Z,...]`	INFORM $[\![$ π Artist, (Genre = Rock) $]\!]$ VALUES $[\![$ Artist : {X, Y, Z, . . .} $]\!]$
Preference Statement	I like Creed when I work	INFORM $[\![$ $\oplus$ (Artist = Creed), (Situation = Work) $]\!]$
Recommendation	Have you heard the song Just One?	INFORM $[\![$ π YN, (Song = Just One) $]\!]$
Agreement	No, I don't like Hoobastank	INFORM $[\![$ YN = No $]\!]$
		INFORM $[\![$ $\ominus$ (Artist = Hoobastank) $]\!]$
Greet	Hello.	GREET $[\![$ $]\!]$
Bye	Good bye.	BYE $[\![$ $]\!]$

Table 5: A sub-set of PCQL action mappings in ACORNSONG for dialogue acts in preference dialogue (see Section 2). Listed here is one I-TAG (INFORM), one A-TAG (ASK), and two C-TAGS (GREET and BYE, both with empty arguments).

4.2 Modeling Preferences using PCQL

Preference strengths are based on Carberry *et al.*'s model (see Section 2) and stored in a structure called a preference map. ACORNSONG's preference strength model differ from Carberry *et al.*'s in four ways: (a) by adding *entity type weights*, relative importance of each type is modeled; (b) by combining weights and strengths we arrive at a more fine-grained strength interval on a continuous scale (close in spirit with the ADAPTIVE PLACE ADVISOR [Thompson *et al.*, 2004]); (c) by connecting the derived strengths to different preference models for different situations we derive *situation-dependent* preference strengths for a given domain item (e.g. for Situation = Work in Figure 1); and (d) by modeling comparative and superlative preferences the interaction involves a more human-like quality. Carberry *et al.*'s model is implemented as follows: Formally, by a **weighted entity type name** $a_{(w)}$ we understand an entity name a with an associated weight w within the interval $[0, 1]$. Each time a type name is mentioned in the dialogue it typically gets a weight increase or decrease (simultaneously causing other type weights to decrease or increase). Users may also directly dictate importance, such as utterance U8 in Figure 1 which results in a weight adjustment for the types Artist and Album.

For example, consider the following preference map based on parts of the dialogue in Figure 1:

$$\begin{array}{ll} \text{Genre}_{(0.5)} & \rightarrow \text{Rock}_{(5)}, \text{Electronic}_{(2)} \\ \text{Artist}_{(0.4)} & \rightarrow \text{Creed}_{(6)} \\ \text{Album}_{(0.1)} & \rightarrow \text{Mind Storm}_{(-3)} \end{array}$$

This structure contains the entity type name Genre with strength-annotated values for Rock and Electronic. The weight for Genre is 0.5, the strength for Rock is 5, and the strength for Electronic is 2. Similarly, it contains the type Artist with weight 0.4 a value Creed with strength 6, and the type Album with a value. The Album entity value Mind Storm is one of the *referential* preferences derived from utterance U7 in Figure 1. This preference map is well-formed since the weights 0.5, 0.4, and 0.1 have the sum 1.0.

A total **preference score** for each song in the current preference model (i.e. Situation = Work) is calculated after each turn by a basic recommendation algorithm, which multiplies detected preference strengths with the entity type weights. Recommendations can then be made by selecting songs with the highest preference scores, and realised in the on-going dialogue (such as utterance S9a in Figure 1).

5 Summary

We have presented PCQL, a query language for preference dialogues. PCQL is a result of preference dialogue analyses, and is a complete formalism for designing conversational recommender systems that behave human-like in their dialogue strategy. Central to PCQL is the dual purpose of describing actions to be performed by an agent and expressing aspects of a factual and preference state, termed FP state. In the paper we describe how to map preference utterances using FP state formulas. We present FP state formulas for factual utterances and various types of preference utterances. We also show how to map FP state formulas to dialogue acts and how they are used in a dialogue system. The PCQL formalism is demonstrated in the application domain of recommender systems, and the ACORNSONG music recommender system is briefly presented.

Our work with ACORNSONG suggests that the same categories and dialogue descriptions are found in approximately the same proportions as in the movie domain, upon which PCQL was based. Future work include generalizing to yet other domains and to refine the dialogue strategies in further user evaluations.

References

[Allen *et al.*, 2000] J. Allen, M. Byron, M. Dzikovska, G. Ferguson, L. Galescu, and A. Stent. An Architecture for a Generic Dialogue Shell. *Natural Language Engineering*, pages 1–16, December 2000.

[Bentley, 1986] Jon Bentley. Programming Pearls: Little Languages. *Commun. ACM*, 29(8):711–721, 1986.

[Bunt, 1994] Harry Bunt. Context and Dialogue Control. *Think*, 3:19–31, 1994.

[Carberry *et al.*, 1999] Sandra Carberry, Jennifer Chu-Carroll, and Stephanie Elzer. Constructing and Utilizing a Model of User Preferences in Collaborative Consultation Dialogues. *Computational Intelligence*, 15(3):185–217, 1999.

[Carenini *et al.*, 2003] Giuseppe Carenini, Jocelyin Smith, and David Poole. Towards More Conversational and Collaborative Recommender Systems. In *Proceedings of the International Conference of Intelligent User Interfaces*, pages 12–18, Miami, Florida, USA, 2003.

[Hintikka, 1962] J. Hintikka. *Knowledge and Belief*. Cornell University Press, Ithaca, New York, 1962.

[Johansson, 2003] Pontus Johansson. Madfilm - a multimodal approach to handle search and organization in a movie recommendation system. In *Proceedings of the 1st Nordic Symposium on Multimodal Communication*, pages 53–65, Helsingör, Denmark, 2003.

[Kripke, 1963] S. A. Kripke. Semantical Considerations on Modal Logics. *Acta Philosophica Fennica*, 16:83–94, 1963.

[Qvarfordt *et al.*, 2003] Pernilla Qvarfordt, Arne Jönsson, and Nils Dahlbäck. The role of spoken feedback in experiencing multimodal interfaces as human-like. In *Proceedings of ICMI'03, Vancouver, Canada*, 2003.

[Singh *et al.*, 2002] Satinder Singh, Diane Litman, Michael Kearns, and Marylin Walker. Optimizing Dialogue Management with Reinforcement Learning: Experiments with the NJFun System. *Journal of Artificial Intelligence Research*, 16:105–133, 2002.

[Thompson *et al.*, 2004] Cynthia Thompson, Mehmet Göker, and Pat Langley. A personalized system for conversational recommendations. *Journal of Artificial Intelligence Research*, 21:393–428, 2004.

Using Collective Acceptance for Modelling the Conversational Common Ground: Consequences on Referent Representation and on Reference Treatment

Sylvie Saget
IRISA - Projet CORDIAL
6, rue de Kerampont - BP 80518
F-22305 Lannion
Sylvie.Saget@enssat.fr

Abstract

A spoken dialog system is commonly considered as being rational. The system's rationality is notably transcribed by its sincerity (following Grice's Maxim of Quality) and by the coherence of its mental state, [Lee, 1997]. On the whole, these fundamental hypothesis lead to high complexity, due to the representation of complex epistemic states and due to the complexity of the associated reasoning processes. In previous works, [Saget, 2006], we claim that an epistemic state is not required *at the dialog level*. The proper mental attitude is (collective) acceptance. The subject of this paper is to introduce the consequences of considering referring acts, as involving (collective) acceptance on referent representation and on reference treatment.

1 Introduction

A spoken dialog system is commonly considered as being rational. The system's rationality is notably transcribed by its sincerity (following Grice's Maxim of Quality) and by the coherence of its mental state, [Lee, 1997].

Moreover, utterance treatment (generation and interpretation) is notably based on the (Subjective) Common Ground, among dialog partners, [Stalnaker, 2002; Clark and Wilkes-Gibbs, 1986; Clark, 1996]. Accommodation, [Lewis, 1979], is then a way to ensure the coherence of their epistemic state while solving coordination problems.

Respecting this fundamental hypothesis constraints spoken dialog systems to support rich epistemic states (containing mutual beliefs and nested beliefs) and the associated reasoning process.

On the whole, concrete spoken dialog systems are not able to deal with these constraints. First at all, on account of the high cost of maintaining a rich user model, the generation of referring expression tailored to the addressee is not done in practice [Carletta and Mellish, 1996; Taylor *et al.*, 1996]. Furthermore, J.A. Taylor et al. [Taylor *et al.*, 1996] proved that these constraints are difficult to deal with in practice. Besides, they proved that, in most cases, nested beliefs are not necessary beyond the second level of nesting (ie. what an agent thinks another agent thinks a third agent (possibly the first one) thinks), as long as deception is not involved. As a result, most spoken dialog systems support nested belief not beyond the second level of nested. Such a complexity seems to be incoherent with human efficiency.

In previous works [Saget, 2006; Saget and Guyomard, 2006], we claim that this problem must be avoided, *at the dialog level*, by considering the mental attitude of *acceptance*. This paper explores the consequences of considering referring acts as involving (collective) acceptance on referent representation and on reference treatment.

2 The philosophical notion of acceptance

Based on the observation that sometimes one may encounter situations where one has to make judgements or has to produce utterances that are contrary to ones private beliefs, philosophers, such has [Cohen, 1992], have introduced the notion of (Collective) Acceptance, which is a (social) intentional mental attitude. (Collective) Acceptances have the following properties, in contrast with beliefs [Wray, 2001]:

- They are voluntary (or intentional);
- They hold on utility or success (thus we can accept something we believe false);
- They do not required justifications;
- All or nothing: we decide to accept or not to accept.

Thus, an individual acceptance $Acc_i(\phi)$ by i of ϕ results from an individual decision or choice in regards of a goal to achieve (which we consider as the current one for i). Collective Acceptance is the social counterpart of individual acceptance:

- $CollAcc_{ij}(\phi)$ stands for "ϕ is a collective acceptation between agents i and j, on i's point of view";
- Collective Acceptance is an intentional attitude, ie. it comes from individual acts of the involved agents: $MB_{ij}(Acc_i(\phi) \wedge Acc_j(\phi)) \Rightarrow CollAcc_{ij}(\phi)$ where:
  - $MB_{i,j}(\phi)$ stands for "ϕ is a shared belief between agents i and j, on i's point of view" and mutual beliefs are formalized as:
  $$MB_{i,j}(\phi) \equiv B_i(\phi \wedge MB_{j,i}(\phi))$$
  - $B_i(\phi)$ stands for "i (implicitly) believes (that) p";
- $CollAcc_{ij}(\phi)$ is symmetric, not reflexive and not transitive.

Proceedings of the 5th Workshop on Knowledge and Reasoning in Practical Dialogue Systems, pages 55–58,
Hyderabad, January 8th, 2007.

M. J. Baker, [Baker, 1994], considers that the result of a negotiation is a collective acceptance. Recently, B. Gaudou et al. [Gaudou *et al.*, 2005] use the notion of acceptance in order to model public beliefs (the use of acceptance allows the authors to avoid the constraining hypothesis on the sincerity and cooperativity of agents in multi-agent systems). These works are concerned by *the basic task level*[1] whereas this paper in concerned by *the spoken dialog level*.

The main difference leads in the justifications for using the mental attitude which is (collective) acceptance. Roughly speaking, the underlying motivation of B. Gaudou and M. Baker's approaches are the notion of *trust* and *compromise*. Furthermore, the key justification to the use of acceptance is the requirement of a compromise in coordination problems. That is the result of a coordination among a group of agents may not be coherent, in the general case, with the individual state of involved agents. In fact, a compromise on the part of at least one agent *in regards of what is the ideal solution on his point of view* is required in most cases in order to ensure coordination's success. A similar approach, in the case of competitive activities, is due to Dunne et al., [Dunne *et al.*, 2005].

In J. Cohen's famous book, "An essay on belief and acceptance" [Cohen, 1992], the author argues that the conversational implicature "a person's saying that p implies that this person believes p" is not the rule and that speech acts such as concessions, acknowledgements, agreements and admissions that p do not imply the existence of the corresponding belief. In such cases, "I thereby accept that p" means that "I take that proposition as a premise for any relevant decision or argument" [Cohen, 1992]. In previous work [Saget, 2006], we claim that an act of reference using a particular description $\imath x.descr(x)$ of an object o does not imply that the speaker believes that $descr(o)$ holds, but implies that the speaker believes that this description enables the addressee to pick out the correct intended referent.

In the particular case of dialog, acceptance is involved as resulting of the choice of a particular linguistic description in order to achieve the intended communicative intention because this choice is goal-directed, because of the mediated nature of dialog and this is increased in the particular case of task-oriented dialog ([Saget, 2006; Saget and Guyomard, 2006]).

3 Reference and Acceptance

In order to model dialog as a collaboration, reference treatment has to be considered at the speech act level [Cohen and Levesque, 1994], as it is done in A. Kronfeld's work [Kronfeld, 1990].[2]

Defining reference as involving acceptance consists notably in:

1. Defining precisely the goal of referential acts in regards of which they are chosen,
2. Defining the relationship between referential expression and their true value.

3.1 Kronfeld's reference theory

Individuating Sets and Individual Object Representation
In order to avoid the constraints related to the standard name assumption, the notion of *Individuating Set* (IS) has been introduced by D. Appelt and A. Kronfeld, [Appelt and Kronfeld, 1987; Kronfeld, 1990]. An individuated state [3] is a mental representation of an object, constituted of *intentional object representations* (IOR). They distinguish several kinds of IOR, such as perceptive IORs (IOR_P), which result from perceptive acts, and discursive IORs (IOR_D), which result from referential acts in a dialogue. These intentional object representations are formalized in terms of *beliefs*, explicitly or implicitly as in S. Jorgensen' implementation [Jorgensen, 2000].

Then, each IS is a particular mental representation, mr_k, of an object, $o_{k'}$. Each particular description, or IOR, is attached to a particular mr_k, using (cf. [Jorgensen, 2000]):

$designate(o_1, John)$: for proper name,

$indef(o_1, \lambda x.human(x))$: for indefinite,

$def(o_1, \lambda x.king(x))$: for definite.

Let's consider the following example: two agents i and j are looking at a table where there is a red book. i tells j: "The book is one of John's books". The following ISs are part of i's mental state:

$mr_1 = \{o_1\}$
(IOR_P) $B_i(def(o_1, \lambda x.book(x) \wedge red(x) \wedge on(x, o_2)))$
(IOR_D) $B_i(def(o_1, \lambda x.book(x)))$
(IOR_D) $B_i(def(o_1, \lambda x.owner(x, o_3)))$

$mr_2 = \{o_2\}$
(IOR_P) $B_i(def(o_2, \lambda x.table(x)))$

$mr_3 = \{o_3\}$
$B_i(designate(o_3, John))$

where $mr_k = o_{k'}$ stands for "the Intentional Set mr_k represents the object $o_{k'}$".

In order to manipulate Intentional Sets, two functions are used:

$IS : ior \rightarrow \{mr\}$
example:
$IS(designate(x, John)) = \{o_3\}$

$IOR : rm \rightarrow \{ior\}$
example:
$IS(o_3) = \{def(o_1, \lambda x.book(x) \wedge red(x) \wedge on(x, o_2)), def(o_1, \lambda x.book(x)), def(o_1, \lambda x.owner(x, o_3))\}$

[1] As spoken dialog systems are mainly concerned with goal-oriented dialog, the basic activity corresponds to the task which motivates the interaction, the dialog, with the system.

[2] A computational implementation is provided in [Jorgensen, 2000]. Moreover, we have proposed a rational model of dialog based on collective acceptance and the model of referential act [Saget and Guyomard, 2006] based on A. Kronfeld's work in the rational model used [Bretier *et al.*, 1995].

[3] More precisely, *"An agent's individuating set for an object is the maximal set of terms such that each term is believe by the agent to denote this object"*, [Lochbaum, 1995].

IORs are then considered as being a tool, a way for accessing to a particular IS.

Referring

Based on the notion of Individuating Set, A. Kronfeld considers that an agent refers when:

"the speaker has a mental representation denoting what he believes to be a particular object, and he intends to hearer to come to have the representation denoting the same object $(\cdots)$", [Kronfeld, 1990].

Following A. Maida, [Maida, 1991], P. Bretier et al. use the notion of representation rather than the notion of denotation which is too strong.

Goals of referring acts

A. Kronfeld defines the underlying goal of referring act, [Kronfeld, 1987]:

"When a speaker performs the speech act of referring, he has in mind an individuating set which he believes determines a particular object, and he uses a noun phrase with the following intentions:

1. *Literal goal: that as a result of the hearer's recognition of the noun phrase as a referring expression, the hearer will generate a local individuating set that will determine this very same object;*
2. *Discourse purpose: that the hearer will apply various operations to the newly created individuating set so that it will meet the appropriate identification constraints."*

Besides, according to H.H. Clark et al. in [Clark and Bangerter, 2004], the literal goal has to be refined in considering this two interrelated goals:

1.1. *Identification: Speaker is trying to get his addressee to identify a particular referent under a particular description.*

1.2. *Grounding: Speaker and his addressee are trying to establish that the addressee has identified the referent as well enough for current purpose.*

The discourse purpose is, for goal-oriented dialog, a bridge relating the dialog level to the basic activity level. Identification constraints comprise notably the intended object's properties the addressee have to know in order to realize the basic task. A well known example is the case of someone who wants to send a card to one of his friends. Another person says him "Send it to John's house". The identification constraint inheriting, from the basic activity of 'Sending somebody a letter', by reference treatment is to know the address of John's house.

Such identification constraints are transcribed, by K. Lochbaum [Lochbaum, 1995], in her extension of the knowledge preconditions of Shared Plans. She stipulates that each parameter p, designated by an IOR (an "access point"), of an action a must have a satisfying description, this is formalized by:

$has.sat.descr(G, p, \mathcal{F}(\bar{a}, p))$ [4]
where:

- G stands for the group of agents, $|G| \geq 1$, who must realize the action a,
- $\bar{a}$ stands for the type of action a belongs to,
- $\mathcal{F}(\bar{a}, p)$ stands for a function whose output is the set of identification constraints for the parameter p in the context of a's type $\bar{a}$.

Having a satisfying description of a particular object, which is designated by its mental representation's "access point" p ($IS(p)$), is defined by the knowledge of a property satisfying the identification constraints provided by $\mathcal{F}(\bar{a}, p)$. That is the existence of an IOR p' in $IS(p)$ satisfying $\mathcal{F}(\bar{a}, p)$. p' may be different from p or equal.

3.2 Considering acceptance

Reasons to choose a referential expression

In the general case, we consider that the linguistic description has not to be coherent with the dialog partners' belief states. The linguistic description must be appropriate in order to reach the goals of referential acts. But, in the particular case where the linguistic description corresponds (exactly or in part) to the properties concerned by the identification constraint, then the linguistic has to be true. Otherwise, this is a case of *real* deception. Concerning reference treatment, this is *a constraint of true value*.

Referent representation

Individuated sets are considered as being constituted by a set of beliefs (IORs) on the object representation. Then, considering (collective) acceptance leads to a clear separation linguistic descriptions, which are referential expressions, whose aim is to be an "access point" in dialog partners' mental states. These descriptions' content are "dialog dedicated". They are not crucial at the basic level task, such an importance manifest itself through the true value constraint.

How is the referential/attributive distinction transcribed? As previously said, the rational choice/acceptance of a noun phrase in the referential use is properly transcribed by the mental attitude which is (acceptance). In the case of an attributive use, the speaker's underlying goal is to provided informations on the intended to the addressee. Consequently, this is providing evidence of the true value of the noun phrase's content. The speaker's aim is to convince or to inform the addressee, the resulting mental attitude is then a belief. In the case of the example earlier done :

$mr_1 = \{o_1\}$

$(IOR_P)\ B_i(def(o_1, \lambda x.book(x) \wedge red(x) \wedge on(x, o_2)))$

$(IOR_D)\ CollAcc_{ij}(def(o_1, \lambda x.book(x)))$

$(IOR_D)\ MB_i(def(o_1, \lambda x.owner(x, o_3)))$

$mr_2 = \{o_2\}$

$(IOR_P)\ B_i(def(o_2, \lambda x.table(x)))$

$mr_3 = \{o_3\}$

$B_i(designate(o_3, John))$

[4] As time is not explicitly in the rational model defined by D. Sadek, the time parameter is not considered.

4 Conclusion

Considering a spoken dialog system as being sincere and modelling the Conversational Common Ground by an epistemic state leads to such complexity problems that most existing spoken dialog systems restrict the theoretical assumption followed.

In previous works, we shown that the appropriated alternative is to consider the pragmatic and goal-directed aspects of the choice of a linguistic description, in order to reach mutual understanding. This consideration leads to consider the Conversational Common Ground as modelled in term of collective acceptances, rather than in term of a belief state.

A referential expression is viewed as a tool enabling dialog partners to access to the mental representation of the correct intended object. In order to do so, the description has to distinguish the correct object from the other possible candidates, a goal which is already used by traditional methods ([Dale and Reiter, 1995]). Moreover,in the general case, referential expression have not to be true. A first case where the true value is necessary has been shown.

Future works will be concerned by the computational implementation of the model and principles by extending S. Jorgensen's work.

Acknowledgment

This work is partially financed by the grant A3CB22 / 2004 96 70 of the regional council of Brittany.

References

[Appelt and Kronfeld, 1987] D. Appelt and A. Kronfeld. A computational model of referring. *Proceedings of The 10th IJCAI*, pages 640–647, 1987.

[Baker, 1994] M. J. Baker. A model for negotiation in teaching-learning dialogues. *Journal of Artificial Intelligence in Education*, 5(2):199–254, 1994.

[Bretier *et al.*, 1995] P. Bretier, F. Panaget, and M.D. Sadek. Integrating linguistic capabilities into the formal model of rational agent: Application to cooperative spoken dialogue. In *AAAI Fall Symposium on Rational Agency: Concepts, Theories, Models, and Application*, 1995.

[Carletta and Mellish, 1996] J. Carletta and C. Mellish. Risk-taking and recovery in task-oriented dialogue. *Journal of Pragmatics*, 26:71–107, 1996.

[Clark and Bangerter, 2004] H. H. Clark and A. Bangerter. Changing conceptions of reference. In I. Noveck & D. Sperber, editor, *Experimental pragmatics*, pages 25–49. Palgrave Macmillan, Basingstoke, England, 2004.

[Clark and Wilkes-Gibbs, 1986] H.H. Clark and D. Wilkes-Gibbs. Referring as a collaborative process. In *Cognition*, volume 22, pages 1–39. The MIT Press, 1986.

[Clark, 1996] H.H. Clark. *Using language*. Cambridge University Press, Cambridge, UK, 1996.

[Cohen and Levesque, 1994] P.R. Cohen and H.J. Levesque. Preliminaries to a collaborative model of dialogue. *Speech Communication*, 15:265–274, 1994.

[Cohen, 1992] J. Cohen. *An Essay on Belief and Acceptance*. Oxford University Press, Oxford, 1992.

[Dale and Reiter, 1995] R. Dale and E. Reiter. Computational interpretations of the gricean maxims in the generation of referring expressions. *Cognitive Science*, 19(2):233–263, 1995.

[Dunne *et al.*, 2005] P. E. Dunne, M. Wooldridge, and M. Laurence. The complexity of contract negotiation. *Artificial Intelligence*, 164(1-2):23–46, 2005.

[Gaudou *et al.*, 2005] B. Gaudou, A. Herzig, and D. Longin. A logical framework for grounding-based dialogue analysis. 2005.

[Jorgensen, 2000] S.W. Jorgensen. *Computational reference - An investigation, Development and implementation of Kronfeld's Theory of reference*. PhD thesis, Copenhagen Business School, 2000.

[Kronfeld, 1987] A. Kronfeld. Goals of referring acts. *TINLAP-3*, pages 143–149, 1987.

[Kronfeld, 1990] A. Kronfeld. *Reference and Computation : An Essay in Applied Philosophy of Language*. Cambridge University Press, 1990.

[Lee, 1997] M. Lee. Rationality, cooperation and conversational implicature. In *Proceedings of the Ninth Irish Conference on Artificial Intelligence*, 1997.

[Lewis, 1979] D. Lewis. Scorekeeping in a language game. *Journal of Philosophical Logic*, 8:339–359, 1979.

[Lochbaum, 1995] K. Lochbaum. The use of knowledge preconditions in language processing. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, pages 1260–1266, San Francisco, 1995.

[Maida, 1991] A. Maida. Maintaining mentals model of agents who have existential misconceptions. *Artificial Intelligence*, 50(3):331–383, 1991.

[Saget and Guyomard, 2006] S. Saget and M. Guyomard. Goal-oriented dialog as a collaborative subordinated activity involving collaborative acceptance. In *Proceedings of the 10th Workshop on the Semantics and Pragmatics of Dialogue (Brandial 2006)*, pages 131–138, University of Potsdam, Germany, 2006.

[Saget, 2006] S. Saget. In favour of collective acceptance: Studies on goal-oriented dialogues. In *Proceedings of Collective Intentionality V*, Helsinki, Finland, 2006.

[Stalnaker, 2002] R. Stalnaker. Common ground. *Linguistics and Philosophy*, 25, 2002.

[Taylor *et al.*, 1996] J.A. Taylor, J. Carletta, and C. Mellish. Requirements for belief models in cooperative dialogue. *User Modeling and User-Adapted Interaction*, 6(1):23–68, 1996.

[Wray, 2001] K.B. Wray. Collective belief and acceptance. *Synthese*, 129:319–333, 2001.

Author Index