Natural Language Engineering **15** (1): 9–30. © 2008 Cambridge University Press
doi:10.1017/S1351324908004889 Printed in the United Kingdom

Interactive question answering and constraint relaxation in spoken dialogue systems

S. VARGES[1], F. WENG[2] and H. PON-BARRY[3]

[1]*Department of Information and Communication Technology, University of Trento, 38050 Povo di Trento, Italy*
[2]*Bosch Research and Technology Center, 4009 Miranda Ave., Palo Alto, CA 94304, USA*
[3]*School of Engineering and Applied Sciences, Harvard University, 33 Oxford St., Cambridge, MA 02138, USA*
e-mail: varges@dit.unitn.it; fuliang.weng@us.bosch.com; ponbarry@eecs.harvard.edu

(*Received 1 March 2007, revised 1 October 2007*)

Abstract

We explore the relationship between question answering and constraint relaxation in spoken dialogue systems and develop dialogue strategies for selecting and presenting information succinctly. In particular, we describe methods for dealing with the results of database queries in information-seeking dialogues. Our goal is to structure the dialogue in such a way that the user is neither overwhelmed with information nor left uncertain as to how to refine the query further. We present two sets of evaluation results for a restaurant selection task: one is a system performance evaluation experiment involving twenty subjects, the other is an experimental evaluation of the use of suggestions involving sixteen subjects.

1 Introduction

Information presentation is an important issue when designing a dialogue system. This is especially true when the dialogue system is used in a high-stress environment, such as driving a vehicle, where the user is already occupied with the driving task. In this paper, we explore efficient dialogue strategies to address these issues, and present implemented knowledge management, dialogue and generation components that allow cognitively overloaded users – see CHAT system (Weng *et al.* 2004b, 2007) for example – to obtain information from the dialogue system in a natural way. We describe a knowledge manager that provides factual and ontological information, a content optimizer that regulates the amount of information and a generator that realizes the selected content. The domain data is divided between domain-specific ontologies and a database backend.

We developed the system for restaurant selection, MP3 player and navigation tasks. The specific task we are focusing on in this paper is the selection of a *single* item out of a larger set of items. Thus, this work is relevant to those applications that require the identification of a single item for the task to proceed/action be carried out: selecting the song to play next, the restaurant to go to tonight or the route to take to a specific location. We can thus contrast the presented work with

https://doi.org/10.1017/S1351324908004889 Published online by Cambridge University Press

information browsing and similar activities that do not necessarily need to arrive at a single item. We also contrast our task with answering factoid/list questions, (see Kaplan 1983; Frank *et al.* 2005, for example), where the required information to answer the question is assumed to be contained in the user's first utterance. In contrast, users of a dialogue system for restaurant selection, for instance, may not provide all the required information in the first dialogue turn, for example, because they are not aware of the detailed options available or even because they have not made up their mind yet.

We address the task of interactive question answering within a *spoken* dialogue system (SDS). On the one hand, using a spoken modality allows the user to focus their visual attention on other tasks such as driving. On the other hand, it imposes additional constraints on the question answering system, for example by not being able to visually present larger lists of items and the requirement to keep spoken system output short. These are some of the issues we will address in this paper.

The presented work was built on top of a set of spoken language technologies previously developed for tasks such as controlling an autonomous mobile robot helicopter (Lemon *et al.* 2002). In the following section, we give an overview of the CHAT dialogue system covering all the important modules that are used to address the needs of interactive item selection. We then describe the knowledge management (Section 3), information presentation (Section 4) and generation components (Section 5). In Section 6 we present evaluation results obtained from a user study. This is followed by a general discussion (Section 7), related work (Section 8) and conclusions.

2 System architecture

The CHAT system has adopted many state-of-art technologies and has grown beyond its heritages over the years (Weng *et al.* 2001; Lemon *et al.* 2002; Bratt *et al.* 2005). The system consists of several core modules, including modules for spoken language understanding (SLU), dialogue management (DM), content optimization (CO), knowledge management (KM) and response generation (RG) (see Figure 1). Specifically, the KM, CO and RG modules are newly created to address the needs of interactive item selection.

The SLU module takes the output of a Nuance speech recognizer and robustly produces structured semantic representations for the dialogue manager, given noisy input from the distracted multi-tasking user in combination with speech recognition errors. To deal with spoken irregularity, it integrates multiple understanding components such as an edit region detection algorithm (Zhang and Weng 2005; Zhang, Weng and Feng 2006), a partial name identifier (Weng *et al.* 2004a), a domain-specific shallow semantic parser and a domain-general deep structural parser (Weng *et al.* 2001). The edit region detection algorithm identifies hesitation, repeat or repair regions in an utterance and removes them for easier late stage processing. The partial name identifier connects a partial name to its full name.

The DM originated from the CSLI dialogue manager (Lemon *et al.* 2002) and follows the information state update approach (Larsson and Traum 2000). It uses a dialogue move tree to keep track of multiple dialogue threads and multiple

https://doi.org/10.1017/S1351324908004889 Published online by Cambridge University Press

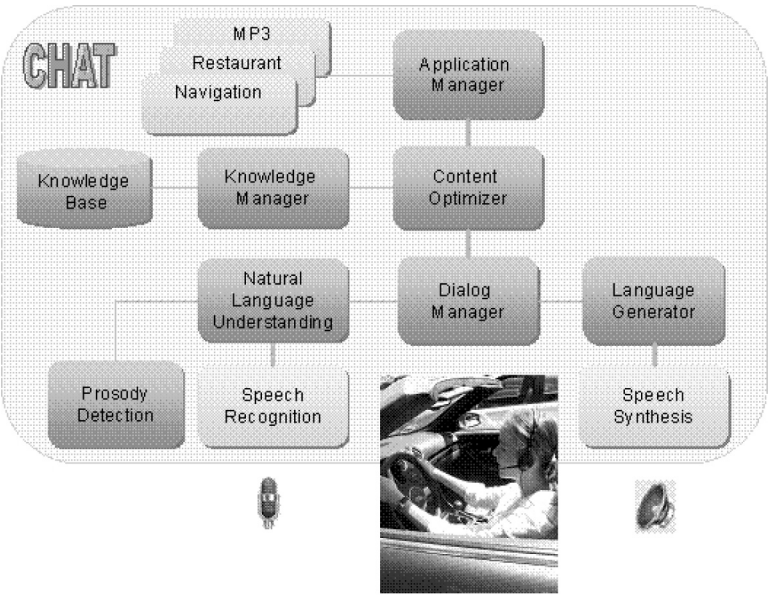


Fig. 1. Architecture of the CHAT dialogue system.

applications (Mirkovic and Cavedon 2005; Purver, Ratiu and Cavedon 2006). Another major functionality of the DM is to convert the semantic interpretation obtained from the SLU module into semantic frames, which are used to query knowledge bases through the CO and KM. Motivated by the need to quickly tailor the system to new domains, the conversion is realized through 'dialogue move scripts', that is, only the scripts need to be adapted, not the underlying Java code (Mirkovic and Cavedon 2005; Weng *et al.* 2005). The scripts define short sequences of dialogue moves. For example, a command move ('play song X') may be followed either by a disambiguation question or a confirmation that the command will be executed. When results from the CO are returned, they are passed to the language generator and presented to the user through Nuance Vocalizer, a text-to-speech synthesizer. The processing strategies for these queries and the information flow among the DM, CO and RG are the central theme of this paper.

To form a complete overview, a brief description of the KM, CO and their connection to the other modules is given here. The KM controls the access to knowledge base sources. The CO module acts as an intermediary between the DM module and the KM module, controls the amount of content and provides recommendations to users. It receives queries from the DM, resolves possible ambiguities and queries the KM. It performs an appropriate optimization strategy based on the returned results (Pon-Barry, Weng and Varges 2006).

Similar to the information channel to the user, the RG module uses a hybrid rule-based/machine learning-based approach. It takes query results from the KM via CO and generates natural language sentences as system responses to user utterances. The query results are converted into natural language sentences using a rule-based

https://doi.org/10.1017/S1351324908004889 Published online by Cambridge University Press

bottom-up production system. Finally, a scoring and ranking algorithm is used to select the best generated sentence (Varges 2005).

The dialogue system is fully implemented and supports mixed initiative dialogues for multiple domains: restaurant selection, MP3 player and navigation tasks (Weng *et al.* 2007). The examples in this paper are mostly taken from the restaurant selection task.

3 Knowledge and content management

A major purpose of the dialogue system is to allow the user access to backend databases. Questions issued by the user are mapped to constraints that are used in the database query. For example, in the Restaurant domain, the request in example (1a) results in the set of query constraints in (1b) (which we also call a 'semantic frame'; more details in Section 3.2):

```
(1a) 'I want to find an inexpensive Japanese restaurant that
      takes reservations'

(1b) system:Category = restaurant:Restaurant
     restaurant:PriceLevel = 0-10
     restaurant:Cuisine = restaurant:japanese
     restaurant:Reservations = yes
```

Depending on the database, queries may yield very different results: an empty result set requires different strategies from a large one both at the human–machine dialogue and the database query level. In this section, we first outline our general approach to handling query constraints and then describe some implementation details. In Section 4, we consider dialogue strategy and presentation issues.

3.1 Constraint handling in user queries to database

A query is composed of a (non-empty) set of constraints. This set needs to contain at least the category of the objects sought, for example restaurants (`hasCategory=restaurant`) or MP3 songs (`hasCategory=MP3`). Effectively, this allows us to identify the database table relevant to the query. The type is usually explicitly given by the user in the form of a base noun ('restaurant', 'song') but may also be apparent from the dialogue/application context. The 'lifecyle' of a constraint in the dialogue is determined by three phases:

1. Introduction
2. Generalization (relaxation) and specialization (constraining)
3. Removal

Generalization and specialization may be performed repeatedly, or not at all. Every constraint consists of a name and value, e.g., `hasDressCode=casual`. The nature of the value determines how a constraint is organized in a specific 'shape':

1. hierarchical: for example, `hasCuisineType=Italian` or `French` or `Thai` or `Korean`, ... , where `Thai` and `Korean` are subtypes of `Asian`, and `Italian` and

https://doi.org/10.1017/S1351324908004889 Published online by Cambridge University Press

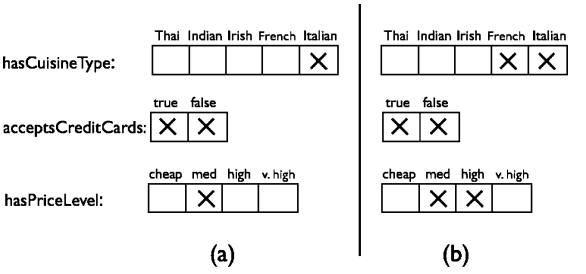


Fig. 2. Example of constraint relaxation in query in terms of underlying model.

`French` are subtypes of `European`. Similarly, the music domain distinguishes `Classic` and `Popular` at the top level; popular music specializes into `Rock`, `80sPop`, etc.

2. linear-ordinal: for example, `hasPriceLevel=cheap` (USD 0–9) or `moderate` (USD 10–19) or `expensive` (USD 20–29)
3. binary: for example, `acceptsCreditcards=true` or `false`

The shape of a constraint determines how constraint relaxation and specialization are performed; other shapes of constraint organization are possible in principle. Constraint operations can be described w.r.t. the underlying model (the database, in practice). This model contains ground and, in case of hierarchical constraints, most specific values for the various constraints of each data point.

Consider, for example, a query for a mid-range Italian restaurant. Figure 2(a) shows the required values that a suitable restaurant needs to 'match'. If the original query does not return any result, the query may be relaxed to any Mediterranean restaurant in the medium to moderately high price range (Figure 2b). The interpretation of 'matching' is as a disjunction since any given restaurant has only one specific value in each category. In the example at hand, 'any Mediterranean restaurant' can be interpreted as `hasCuisineType=Italian` $\vee$ `hasCuisineType=French`. Since nothing is said about acceptance of credit cards, both values are acceptable in both queries. Generally, if a constraint that yields an empty result set is relaxed so that any of the values is permitted, this has the same effect as removing the constraint altogether. For example, in the binary case this results in a disjunct such as `true` $\vee$ `false`.

Constraint specialization can similarly be characterized with respect to the model, namely as a reduction in the number of disjuncts. We note that the characterization of constraint operations with respect to a model is independent of particular implementation choices such as mapping constraints to SQL queries. For example, checks for non-empty bit vector intersection could be used to match queries against data items. A simple optimization that is available in both these cases is to remove checks for constraints for which all values are acceptable.

3.2 Implementation specifics

Our general goal is a configurable, standards-based framework for managing knowledge sources in dialogue systems. Domain knowledge is structured according

https://doi.org/10.1017/S1351324908004889 Published online by Cambridge University Press

to domain-dependent ontologies. The knowledge manager makes use of OWL, a W3C standard, to represent the ontological relationships between domain entities. In example (1b), `restaurant` refers to a class of domain entities. `PriceLevel`, `Cuisine` and `Reservations` are properties of the `restaurant` class. `japanese` is a subclass of `restaurant`. The actual restaurant entries in the database inherit properties from all their superclasses. Protégé (Gennari *et al.* 2002), a domain-independent ontology tool, is used to maintain the ontology offline.

In addition to the KM module, we employ a content optimization (CO) module that acts as an intermediary between dialogue and knowledge management during the query process. It receives query constraints from the dialogue manager, modifies them if necessary and queries the knowledge manager. The decision about constraint modifications is based on the number of items in the query result set and a number of configurable properties, in particular the domain-dependent relaxation strategy (e.g., `restaurant.relaxationStrategy.cuisine = strategies.OntologyStrategy`), thresholds for different DB result set sizes (see also Section 4.2) and the default feature/constraint order, to be used in lieu of a user-tailored feature order in selecting constraints to modify. Relaxing a hierarchical constraint entails replacing it with its parent-concept in the domain ontology. A linear constraint is relaxed by replacing the current value with an adjacent value, and a binary constraint is relaxed by replacing its value with the opposite value.

When the content optimizer obtains a semantic frame from the dialogue manager, it performs the following operations:

1. Resolve ambiguous property names
2. Merge with previous frame, if this is a query revision
3. Send query to KM
4. Select top-level strategy (e.g., processEmpty, processMedium, processLarge, none) based on number of items returned
5. If top-level strategy=none, return query result
6. Else: select constraint to modify, select specific modification strategy, perform modification and return to step 3.

The merging of frames (step 2) refers to user dialogue turns that modify a previous query, for example by adding or 'overwriting' a constraint, rather than beginning a new query. In our system, user utterances are usually interpreted as revisions, unless the user explicitly issues a new query. For example, if the user's follow-up turn to (1a) is 'How about a Thai restaurant?', this changes the value of constraint `restaurant:Cuisine` from `restaurant:japanese` to `restaurant:thai` but keeps the other constraints rather than 'forgetting' them. The distinction between revision moves and new queries is made by the dialogue manager. However, we do not address the general issue of dialogue move recognition in this paper. More details can be found in (Purver *et al.* 2006).

The content optimizer also calculates statistics for every set of items returned by the knowledge manager in response to a user's query. If the result set is large, these figures can be used by the dialogue manager to give meaningful summary responses (e.g., in the MP3 domain, 'There are 85 songs. Do you want to list them by a genre

https://doi.org/10.1017/S1351324908004889 Published online by Cambridge University Press

such as Rock, Pop or Soul?'). However, the ultimate goal of our dialogue strategies is to arrive at a single item.

The content optimizer also produces constraints that represent meta-knowledge about the ontology:

```
(2) rdfs:subClassOf = restaurant:Cuisine
```

This constraint is produced for a user input 'What cuisines are there?' and allows the system to list meta-knowledge in ways similar to object-level knowledge.

4 Dialogue strategies for dealing with query results

4.1 General dialogue considerations

Quite separate from the general characterization of constraint handling (Section 3.1) is its application during dialogue: the system relaxes constraints automatically, but does not introduce new ones or specialize existing ones unless explicitly triggered by user input. This is motivated by the desire to reduce the burden on the user: without automatic constraint relaxation, the system can only report an empty result set (and make a suggestion; see below). On the other hand, automatically specializing constraints (a) seems risky since it may result in costly correction turns by the user and (b) violates the general HCI guideline to let the user be in control of the interaction. The dialogue strategies are currently manually defined.

Suggestions in our implementation are restricted to new constraints. The system does not make suggestions for constraint value specializations, e.g., 'You could look for a Italian or French restaurant', if the user was originally looking for Mediterranean cuisine. However, the user can always state a more specialized constraint explicitly, which 'overrides' the previous constraint value. Furthermore, explicit removal of constraints is currently not supported ('Forget about cuisine type. I'm hungry!'). However, the user can always explicitly start a new query.

4.2 Presenting query results

Verbalizing the constraints used in a query is crucial for robustness and usability. It provides feedback to the user about what the system understood and what actions it performed (for example, constraint relaxations). This is important since the system may miss or misinterpret constraints, leading to uncertainty on the user's side about what constraints were used. For this reason, a generic system response such as 'I found 9 items' is not sufficient.

In addition to specifying the understood query constraints, the system needs to present the results of the query. Consider the following interaction that follows user turn (1a) (which is shown again below):

```
(3)   U: i want to find an inexpensive Japanese restaurant that takes
         reservations
      S: I found 9 inexpensive Japanese restaurants that take reservations .
         Here are the first few :
      S: GINZA JAPANESE RESTAURANT
```

https://doi.org/10.1017/S1351324908004889 Published online by Cambridge University Press

```
S: OKI SUSHI CAFE
S: YONA SUSHI
S: Should I continue?
```

While the example query has a relatively small result set which can be listed directly, this is not always the case, and we thus need dialogue strategies that deal with different result set sizes. For example, it does not seem sensible to produce 'I found 2,500 restaurants. Here are the first few: … '. At what point does it become unhelpful to list items? We do not have a final answer to this question – however, it is instructive that the (human) wizard in our data collection experiments (Cheng *et al.* 2004) did not start listing when the result set was larger than about ten items. In the implemented system, we define dialogue strategies that are activated at adjustable thresholds.

Even if the result set is large and the system does not list any result items, the user may still want to see some example items returned for the query. This observation is based on comments by subjects in pilot experiments that in some cases it was difficult to obtain any query result at all. For example, speech recognition errors may make it difficult to build up a sufficiently complex query. In response to this, we always give some example items even if the result set is large. (An alternative would be to start listing items after a certain number of dialogue turns.) Furthermore, the system should encourage the user to refine the query by suggesting constraints that have not been used yet. This is done by maintaining a list of unused constraints that gets smaller as the dialogue progresses. This list is roughly ordered according to the frequency of constraint use. For example, using cuisine type is suggested before proposing to ask for information about reservations or credit cards.

Table 1 summarizes the main dialogue strategies for presenting the results of database queries. The strategies are conditioned on the size of the *final* result set and whether or not any modifications were performed. The dialogue strategies represent implicit confirmations and are used if NLU has a high confidence in its analysis of the user utterance; see Varges and Purver (2006) for more details on our handling of robustness issues.

For empty result sets, the system states that it has not found any matching items (Table 1, s1a,b). Small result sets up to a threshold t_1 are listed in a single sentence (s2a,b). For medium-sized result sets up to a threshold t_2, the system starts listing immediately (s3a,b). For large result sets, the generator shows example items and makes suggestions as to what constraint the user may use next (s4a,b). If the CO module performs any constraint modification, the first sentence of the system turns reflects the modification (s1b,s2b,s3b,s4b). 'NP-original' and 'NP-optimized' in Table 1 are used for brevity and are explained in the next section.

5 Response generation

The task of the generator is to produce system responses given input in two dimensions:

https://doi.org/10.1017/S1351324908004889 Published online by Cambridge University Press

Table 1. *Dialogue strategies for dealing with query results*

	$\|result_{final}\|$	mod	Example realization
(s1a)	0	No	I'm sorry but I found no restaurants on Mayfield Road that serve Mediterranean food.
(s1b)	0	Yes	I'm sorry but I found no [NP-original]. I did not even find any [NP-optimized].
(s2a)	Small: $> 0, < t_1$	No	There are 2 cheap Thai restaurants in Lincoln in my database: Thai Mee Choke and Noodle House.
(s2b)	Small	Yes	I found no cheap Greek restaurants that have a formal dress code but there are 4 inexpensive restaurants that serve other Mediterranean food and have a formal dress code in my database: . . . .
(s3a)	Medium: $>= t_1, < t_2$	No	I found 9 restaurants with a two star rating and a formal dress code that are open for dinner and serve French food. Here are the first ones: . . . .
(s3b)	Medium	Yes	I found no [NP-original]. However, there are N [NP-optimized]. Here are the first few: . . . .
(s4a)	Large: $>= t_2$	No	I found 258 restaurants on Page Mill Road, for example Maya Restaurant, Green Frog and Pho Hoa Restaurant. Would you like to try searching by cuisine?
(s4b)	Large	Yes	I found no [NP-original]. However, there are N [NP-optimized]. Would you like to try searching by [Constraint]?

1. Dialogue context: the general response dialogue move as determined by the dialogue move scripts (Mirkovic and Cavedon 2005), and the last user utterance at the word level as recognized by ASR;
2. Content optimizer output: the constraints used in the database query; if any constraint modifications were performed; a set of example database items; a summary of the database items; suggestion for next constraint to use. Not all this information is present in all generator inputs.

The specific presentation strategy (Section 4.2) can be determined by the generator based on this input. There are also other response moves that do not deal with content optimizer output which we do not discuss here. Note that the particular position of the system response node under construction in the dialogue move tree does not directly affect the generator. Rather, it affects the particular selection of response dialogue move and possibly the optimizer constraints (e.g., query revision versus new query).

We follow the bottom-up generation approach for production systems described in Varges (2005) and perform mild overgeneration of candidate system turns, i.e., system utterances including the actual surface forms (words), followed by ranking. The highest-ranked candidate is selected for output. In the following, we first describe the 'base generator' that constructs candidate system turns and then the ranking of these output candidates.

https://doi.org/10.1017/S1351324908004889 Published online by Cambridge University Press

5.1 Generation of candidate system turns

The core of the base generator is a set of productions. Productions are 'if-then' rules that operate over a shared knowledge base of facts. Productions map individual database constraints to phrases such as 'open for lunch', 'within 3 miles' or 'a formal dress code', and recursively combine them into NPs. This includes the use of coordination to produce 'restaurants with a 5-star rating and a formal dress code', for example. NP generation for query constraints can be seen as a form of referring expression generation (see Dale and Reiter 1995, for example). In contrast to Varges (2004), the phrases produced by the chart generator are not directly associated with their extension (the set of objects they denote) since these are held in the database backend rather than in-memory.

The decision of the CO module to relax or remove constraints also affects the generator: there are two sets of constraints, an 'original' one directly constructed from the user utterance and optionally an 'optimized' one produced by the CO module to modify the original constraints (Section 3).

The NPs are integrated into sentence templates that realize dialogue moves. For example, original and relaxed NPs are used to describe a constraint modification: 'I found no [NP-original] but there are [NUM] [NP-optimized] in my database'. Several dialogue moves can can be combined to form a candidate system turn. For example, the constraint realizing sentence above can be combined with a follow-up sentence such as 'You could try to look for [NP-constraint-suggestion]'.

The selection of sentence templates is determined by the dialogue move scripts. Typically, a move-realizing production produces several alternative sentences. The NP generation rules realize constraints regardless of the specific dialogue move at hand. This allows us to also use them for clarification questions based on constraints constructed from classifier information if the parser and associated parse-matching patterns fail; all that is required is a new sentence template, for example 'Are you looking for [NP]?'.

We use 283 productions overall in the restaurant, MP3 and navigation domains, 73 of them to generate NPs and VPs that realize 41 possible input constraints (for all domains). Other productions serve to integrate phrases into sentence templates and also perform tasks such as lexical lookup and consistency checking of the NLG input for robustness.

5.2 Ranking

Ranking of candidate system turns is done by using a combination of factors:

1. **Alignment:** Alignment is a key to successful natural language dialogue (Brockmann *et al.* 2005). We perform alignment of system utterances with user utterances by computing an ngram-based overlap score. For example, a user utterance 'I want to find a Chinese restaurant' is presented by the bag-of-words {'I', 'want', 'to', 'find',...} and the bag-of-bigrams {'I want', 'want to', 'to find',...}. We compute the overlap with candidate system utterances represented in the same way and combine the unigram and bigram match scores.

https://doi.org/10.1017/S1351324908004889 Published online by Cambridge University Press

Words are lemmatized and proper nouns of example items removed from the utterances.

2. **Variation:** We use a variation score to 'cycle' over sentence-level paraphrases. Alternative candidates for realizing a certain input move are given a unique alternation number in increasing order. The actual alternation score is inversely related to recency and normalized to [0...1].
3. **Ngram filtering:** We use an ngram filter based on ngrams extracted from 'bad' examples, removing, for example, 'Chinese cheap restaurants' but keeping 'cheap Chinese restaurant'. For generalization, we replace constraint realizations with semantic tags derived from the constraint names (except for the head noun), for example the trigram 'CUISINE PRICE restaurants'. An alternative is to use a more complex grammar formalism to prevent ungrammatical candidate system utterances.
4. **Score combination:** The final candidate score is a linear combination of alignment and variation scores:

$$score_{final} = \lambda_1 \cdot align_{uni,bi} \quad +(1-\lambda_1) \cdot variation \tag{1}$$

$$align_{uni,bi} = \lambda_2 \cdot align_{uni} \quad +(1-\lambda_2) \cdot align_{bi} \tag{2}$$

where $\lambda_1, \lambda_2 \in \{0, \ldots, 1\}$. A high value of λ_1 places more emphasis on alignment, a low value yields candidates that are more different from previously chosen ones. In our experience, alignment should be given a higher weight than variation, and, within alignment, bigrams should be weighted higher than unigrams, i.e., $\lambda_1 > 0.5$ and $\lambda_2 < 0.5$. (Deriving weights empirically from corpus data is an avenue for future research.)

A major benefit of using an overgeneration-and-ranking approach to NLG in dialogue is the possibility of performing alignment with the user utterance. This is possible even in situations where no corpus data is available. (However, we do have wizard-of-oz data to inform the system design; see Section 7.2.)

Alignment allows us to prefer 'restaurants that serve Chinese food' over 'Chinese restaurants' if the user used a wording more similar to the first. The Gricean Maxim of Brevity, applied to NLG in Dale and Reiter (1995) suggests a preference for the second, shorter realization. However, if the user thought it necessary to use 'serves', maybe to correct an earlier mislabeling by the classifier/parse-matching patterns, then the system should make it clear that it understood the user correctly by using those same words. On the other hand, a general preference for brevity is desirable in spoken dialogue systems: users are generally not willing to listen to lengthy synthesized speech.

Mild overgeneration combined with alignment also allows us to map the constraint `PriceLevel=0-10` in example (1b) above to both 'cheap' and 'inexpensive', and use alignment to 'play back' the original word choice to the user.

6 Evaluation

A number of experiments have been conducted on the CHAT system. In this paper, we concentrate on the restaurant selection task because it is more challenging for

https://doi.org/10.1017/S1351324908004889 Published online by Cambridge University Press

constraint handling and information presentation. Specifically, we describe two sets of experiments for the restaurant selection task: one is a general evaluation to test the performance of the CHAT system on the task; the other is a controlled experiment to test the use of a suggestion strategy. The former is covered in Section 6.1 and the latter in Section 6.2.

6.1 General evaluation on restaurant selection

In the general evaluation, twenty subjects were recruited for the experiment. Each subject in the restaurant selection task was given nine scenario descriptions involving three constraints. Subjects were instructed to use their own words to find a fitting restaurant. The following is an example task:

'You are celebrating your sister's birthday and she wants to try something different. A co-worker has suggested a Russian restaurant in the area. You're not sure whether your stomach will agree with this new fare, so be certain to find a place with a five star rating. Also, you want to go somewhere close by, so make sure the restaurant you choose is in Jackson. Use your own words to find a restaurant that meets your needs.'

A particular challenge is to induce the subjects to use their own language but on the other hand not be too imprecise – however, this is not the focus of this paper.

We use a backend database of 2,500 restaurants containing the following information for each restaurant: restaurant name, cuisine type, city and street names, service level, rating, whether they accept credit cards or reservations, price level, open hours, dress code, additional information (e.g., vegetarian food) and distance from current location. Some of these constraints have a fixed number of values (e.g., service level and dress code), whereas those taking named entities as values are obviously open-ended. Recall from Section 3 that the constraints are organized in different ways (hierarchically, linearly, binary) to trigger different constraint handling rules in the CO module.

Figures 3 and 4 show extracts of two of the dialogues from the experiments. To exemplify the challenges the system is faced with, user turns are shown in the form of the output of the speech recognizer (we currently use the best ASR hypothesis as input to the parser). The task of the first dialogue (Figure 3) was to find a restaurant that has the following constraints: [Reservations=yes, ServiceLevel=excellent, Rating=5star]. Despite some minor speech recognition errors, the system correctly builds a query for the two constraints provided by the user in (1.1) and verbalizes these in (1.2) to show its understanding of the user input. It also gives some example items and makes a suggestion. This suggestion is not taken up by the user – in fact, we find that suggestions are generally not taken up by the user. We believe this is due to the nature of the tasks, which specified exactly which criteria to match. On the other hand, in more open application scenarios, where users may not know what questions can be asked, suggestions may be useful. In (1.3) the user issues a subquery that further constrains the result set. By again summarizing the constraints used, the system confirms in (1.4) that it has interpreted the new constraint as a revision of the previous query. The alternative is for the system to initiate a new query, which would be wrong in this context.

https://doi.org/10.1017/S1351324908004889 Published online by Cambridge University Press

Turn	Speaker	Utterance
1.1	User	I need a restaurant with a five star food rating excellent service
1.2	System	I found 425 restaurants with a five star rating and excellent service, for example EL CAPITAN FISH MARKET, K-NINE COUNTRY CLUB, and KAISERHOF GERMAN AMERICAN. You could try to look for cuisine type.
1.3	User	which restaurants take reservations
1.4	System	I found 353 restaurants that have a five star rating, have excellent service and take reservations, for example GRAND CENTRAL CAFE, CHASIN CHICKEN, and CHON WOO RESTAURANT. Would you like to try searching by price?

Fig. 3. Example dialogue 1 from evaluation (extract; ASR output for user turns).

The task of the second dialogue (Figure 4) was to find a restaurant that meets the constraints [BusinessHours=breakfast, StreetName='bower street', DressCode=casual]. This user tends to give shorter, keyword-style input to the system (2.1, 2.8). In (2.3), the user reacts to a clarification question and adds another constraint which the system summarizes in (2.4). The user turn (2.5) is an ASR error which the system cannot handle (2.6, 2.7). The user constraint of (2.8) is correctly used to revise the query (2.9), but the system/ASR misrecognizes 'yes' as 'british' (2.10). This additional constraint yields an empty result set from which the system recovers automatically by relaxing the hierarchically organized cuisine constraint to 'European food'. In (2.11) the system uses dialogue strategy s3b for medium-sized result sets with constraint modifications (Table 1). The result of both dialogues is that all task constraints are met.

In this experiment with twenty subjects, two of them were restarted in the middle. Overall, 180 tasks were performed involving 1,144 user turns and 1,818 system turns. Two factors contributing to the higher number of system turns are (a) some system turns are counted as two turns, such as 2.6, 2.7 in Figure 4, and (b) restaurants in longer enumerations of result items are counted as individual turns. On average, user utterances are significantly shorter than system utterances ($\bar{x} = 4.9$ words, standard deviation $\sigma = 3.82$ versus $\bar{x} = 15.4$ words, $\sigma = 13.53$). This is a result of the 'constraint summaries' produced by the generator. The high standard deviation of the system utterances can be explained by the above-mentioned listing of individual result items (e.g., utterance (2.12) in Figure 4).

We collected usage frequencies for the dialogue strategies presented in Section 4.2: there was no occurrence of empty final result sets (strategy s1a/b) because the system successfully relaxed constraints if it initially obtained no results. Strategy s2a (small result sets without modifications) was used for 61 inputs, i.e., constraint sets

https://doi.org/10.1017/S1351324908004889 Published online by Cambridge University Press

Turn	Speaker	Utterance
2.1	User	breakfast
2.2	System	Are you looking for a restaurant that is open for breakfast?
2.3	User	yes located on bower street
2.4	System	I found 71 restaurants on bower street that serve breakfast, for example, SOUPLANTATION-TO-GO, RAMAYANI WESTWOOD, and PHILLY WEST. You could try to look for cuisine type.
2.5	User	which
2.6	System	I'm sorry, I do not understand what you mean by: which
2.7	System	Please try rephrasing that.
2.8	User	casual dress
2.9	System	There are 51 restaurants on bower street that serve breakfast and have a casual dress code in my database, for example, BERTY'S, LABOTETEGA, and JAVAN RESTAURANT. Would you like to try searching by price?
2.10	User	british
2.11	System	I found no restaurants on bower street that serve british food, have a casual dress code and are open for breakfast but there are 8 restaurants on bower street that are open for breakfast, serve European food and have a casual dress code in my database. Here are the first few:
2.12	System	HOTEL BEL-AIR

Fig. 4. Example dialogue 2 from evaluation (extract; ASR output for user turns).

constructed from user utterances. Strategy s3a/b (medium-sized result sets) was used for 217 times and required constraint relaxations in five cases. Strategy s4a/b (large result sets) was used for 316 inputs and required constraint relaxations in sixteen cases. Thus, the system performed constraint modifications in twenty-one cases overall. All of these yielded non-empty final result sets. For 573 inputs, no modification was required. There were no empty final result sets despite modifications. The 550 user turns that did not involve any constraints comprised utterances such as 'yes', 'tell me more' or restaurant names, for example.

https://doi.org/10.1017/S1351324908004889 Published online by Cambridge University Press

On average, the generator produced 16 candidate system turns for inputs of two constraints, 160 candidates for typical inputs of three constraints and 320 candidates for four constraints. Such numbers can easily be handled by simply enumerating candidates and selecting the 'best' one. One could handle much larger numbers of generation candidates either by using packing (Langkilde 2000) or by interleaving rule-based generation with corpus-based pruning (Varges and Mellish 2001) if complexity should become an issue when doing overgeneration.

Task completion in the experiments was high: the subjects met all target constraints in 170 out of 180 tasks, i.e., completion rate was 94.44 per cent. An error analysis revealed that the reasons for only partially meeting the task constraints were varied. For example, in one case a rating constraint ('five stars') was interpreted as a service constraint by the system, which led to an empty result set. The system recovered from this error by means of constraint relaxation but the user seems to have been left with the impression that there are no restaurants of the desired kind with a five star rating.

After the experimental study, the subjects were given a questionnaire. To the question 'The responses of the system were appropriate, helpful, and clear' (where, in line with Walker *et al.* (2004), 1 = 'strongly agree', 5 = 'strongly disagree'), the subjects gave the following ratings: 1: 7, 2: 9, 3: 2, 4: 2 and 5: 0, i.e., the mean user rating is 1.95. The subjects were somewhat more critical of the quality of the speech feedback ($\bar{x} = 2.41$, $\sigma = 1.18$) and most critical of the 'pace of interaction' ($\bar{x} = 3.14$, $\sigma = 1.08$; all responses were on a scale of 1–5).

6.2 Evaluating the use of suggestions

We ran an experiment comparing two response strategies that varied in the suggestions given to the user. One strategy made use of the CO module and gave suggestions about how to proceed when a user query returned no results or many results (for this experiment 'many' means 'greater than 30'). The second strategy gave responses without such suggestions. Example responses are shown below. Aside from the times a user query returned many results or no results, the two response strategies were identical.

Suggestions strategy (S):

a. Many matches: 'I found 656 cheap restaurants. You can refine your query by adding criteria such as cuisine type'.
b. No matches: 'I found no restaurants in Jackson that serve Indonesian food. However, there are 25 restaurants in Jackson that serve Southeast Asian food'.

No Suggestions strategy (NS):

a. Many matches: 'I found 656 cheap restaurants'.
b. No matches: 'I found no restaurants in Jackson that serve Indonesian food'.

https://doi.org/10.1017/S1351324908004889 Published online by Cambridge University Press

Table 2. *Experiment design*

Task	Num Matches	Strategy Group A	Strategy Group B	Prediction
1	Few (5)	S	NS	Same
2	None (0)	S	NS	S > NS
3	Many (487)	S	NS	S > NS
4	Few (2)	NS	S	Same
5	None (0)	NS	S	S > NS
6	Many (429)	NS	S	S > NS

6.2.1 Experimental design

Participants were sixteen native English speakers, ranging between 19 and 65 years of age. The experimental procedure included three warm-up tasks followed by six evaluation tasks. For each task, participants read a short scenario description containing three criteria, and were instructed to use their own language to talk to the system and find a restaurant. The six evaluation tasks were designed so that a query containing the three criteria would return (a) few matching restaurants, (b) no matching restaurants or (c) many matching restaurants.

Each participant was randomly assigned to Group A or Group B. Group A received suggestions (S) for tasks 1–3 and no suggestions (NS) for tasks 4–6. To counterbalance, Group B received suggestions (S) for tasks 4–6 and no suggestions (NS) for tasks 1–3. The experiment design is summarized in Table 2.

We predicted that users would prefer strategy S over NS in the tasks with no matches and many matching restaurants, and that there would be no difference in preferences for the tasks with few matching restaurants.

To ensure that task pairs (1, 4), (2, 5) and (3, 6) would be comparable, each pair contained the same three criteria (e.g., cuisine type, location and rating) but differed in the values.

6.2.2 Results

Data was collected for a total of ninety-six dialogues. User satisfaction was gauged by asking users to indicate on a 5-point Likert scale their agreement or disagreement with the following statement: 'The system utterances in this task were easy to understand and provided exactly the information I was interested in when choosing a restaurant', where 1 = strongly disagree and 5 = strongly agree.

The user satisfaction results indicate that responses with suggestions are preferred over responses with no suggestions for queries with no matches and many matches, in line with our predictions. Mean user ratings are summarized in Table 3. Paired sample t-tests were performed for each task category. The difference in means was not significant for the few matches and no match categories, but differences were significant (indicated with an '*') for the many matches category ($p < 0.1$). Contrary

https://doi.org/10.1017/S1351324908004889 Published online by Cambridge University Press

Table 3. *Mean user satisfaction ratings*

	No matches ($N=32$)	Few matches ($N=32$)	Many matches ($N=32$)
S	3.68	4.50	4.25*
NS	3.31	4.75	3.63*

Table 4. *Mean ratings for 'many matches' category*

	Task 2 Many matches ($N=16$)	Task 6 Many matches ($N=16$)
S	4.38*	4.00
NS	3.25*	4.00

Table 5. *Mean number of turns per task*

	Total turns/task ($N=96$)	User turns/task ($N=96$)	System turns/task ($N=96$)
S	14.37	5.29*	9.08
NS	16.69	6.48*	10.21

to our prediction that S and NS would be equally preferred for the few matches category, NS responses were rated slightly higher.

We ran an ANOVA to determine whether any of the user satisfaction ratings differed as a function of subject group, and found that this was not the case (i.e., the order in which users saw the two response strategies did not affect their ratings).

Within the 'many matches' category, we looked at means for individual tasks. The mean user ratings are shown in Table 4. Interestingly, S was rated significantly higher than NS for task 3 ($p < 0.1$), but for task 6, there was no difference.

We also measured whether response strategies had any effect on dialogue length (number of turns). Across all ninety-six dialogues, the mean number of turns per dialogue was 14.37 for S and 16.69 for NS. For S dialogues, mean user turns was 5.29 and mean system turns was 9.08; for NS dialogues, mean user turns was 6.48 and mean system turns was 10.21. These results are shown in Table 5. A paired sample t-test was run on each of the S–NS pairs. The difference in means for user turns per dialogue was significant ($p < 0.01$) (marked with an '*') while the differences among the system turns and the total turns were not significant.

7 Discussion

7.1 Suggestion experiment

The results of the experiment presented in Section 6.2 indicate that users prefer responses with suggestions for cases in which they encounter empty result sets or

https://doi.org/10.1017/S1351324908004889 Published online by Cambridge University Press

very large result sets, and that they prefer responses with no suggestions for queries with few results. This preference is most salient for situations where the query result contains many matching items.

The fact that the S–NS difference in means for task 3 was so pronounced and the means for task 6 were identical indicates that responses with suggestions may only be desirable for new users. One possible explanation is that participants in Group A, who received NS responses in task 6, were sufficiently familiar with the space of possible constraints such that the lack of suggestions made no difference.

The lower number of mean dialogue turns for the S strategy, as compared to the NS strategy, indicates that giving responses with suggestions may lead to more efficient dialogues. While it is promising that the S strategy led to fewer user turns, further analysis is needed to determine whether these dialogues are less cognitively demanding.

7.2 Use of corpus resources

On the basis of wizard-of-oz data, the system alternates specific and unspecific refinement suggestions ('You could search by cuisines type' versus 'Can you refine your query?'). Furthermore, many of the phrases used by the generator are taken from wizard-of-oz data too. In other words, the system, including dialogue manager and generator, is informed by empirical data but does not use this data directly (Reiter and Dale 2000). This is in contrast to generation systems such as the ones described in Langkilde and Knight (1998) and Varges and Mellish (2001) that employ corpora to rank candidate system turns produced by the generator. It should be possible to use the wizard turns of the wizard-of-oz data study for ranking system output. This would be in addition to the user turns of the immediate dialogue context which we currently use for computing the alignment score. Furthermore, the system could back-off to more general large-scale corpora to fill in gaps in linguistic knowledge (Langkilde and Knight 1998; Varges and Mellish 2001). However, an important advantage of the alignment technique presented in this paper is that we can do ranking without any corpus resources, solely based on the dialogue context.

Learning can also be applied in other places, including picking a constraint to relax, deciding what constraints to suggest to the user and the thresholds for the information presentation moves described in Section 4.2.

8 Related work

There has been substantial related work in (at least) the areas of spoken dialogue systems, including language generation in dialogue (textual), natural language interfaces to databases and (textual) open domain question answering. One general aspect that distinguishes our work from the one presented below is the fact that we implemented an end-to-end spoken dialogue system including all the components outlined in Section 2 and evaluated it with subjects in the spoken modality.

https://doi.org/10.1017/S1351324908004889 Published online by Cambridge University Press

8.1 Information presentation in spoken dialogue systems

Qu and Green (2002) present a constraint-based approach to cooperative information dialogue that identifies candidates for constraint relaxation in overconstrained queries and restriction candidates in underconstrained ones. In contrast to our approach, constraint relaxation is not done automatically but instead suggested (in textual form).

Demberg and Moore (2006) describe an approach to structuring complex options in the air travel domain by means of content clustering (see also Pellom, Ward and Pradhan 2000) in combination with a user model. These techniques could be incorporated into our system to generate more sophisticated summaries of large result sets. However, the length of the summaries may be an issue that needs to be investigated using a spoken dialogue system. Kruijff-Korbayova, Karagjosova and Larsson (2002) propose to use conditional responses that could be used to present complex options.

Chung (2004) focuses on the use of simulation techniques for rapid application development. Dialogue management is concerned with relaxation strategies in overconstrained queries, e.g., relaxing a geographical constraint according to a geography ontology. These strategies are also available to our system (in manually specified form). Rieser and Lemon (2007) show how information presentation can be automatically adapted to, for example, different levels of noise and screen sizes, by using reinforcement learning and user simulation.

8.2 Question answering

Work on question answering in the 1970s and 1980s focused on natural language interfaces to databases, for example the LUNAR (Woods, Kaplan and Nash-Webber 1972) and COOP (Kaplan 1983) systems. Kaplan describes 'suggestive indirect responses' that also relax query constraints to present the user with a non-empty result set. In contrast to our approach, the structure of the ontology is not used to determine the constraint relaxation strategy. On the other hand, Kaplan's 'corrective indirect responses' (based on the detection of presupposition failure) go beyond the capabilities of the CHAT system.

Frank *et al.* (2005) describe a recent system in this tradition that uses deep linguistic analysis of the question to access structured knowledge sources in restricted domains. Our approach can be seen as complementary to this line of work: we provide query refinement in dialogue and answer presentation in the speech modality that could be used in combination with more sophisticated QA backends. However, more research is needed on how to interface these components, especially if the QA system is operating in open domains.

Kruschwitz and Al-Bakour (2005) describe a web-search interface that suggests query refinements and relaxations. It uses a domain model constructed from a text collection to, for example, back-off to a root node in the concept hierarchy. This seems similar to our relaxation of hierarchically organized constraints (although we do not back-off to the root node directly). Relaxations are suggested whenever no refinements are available, i.e., the actual size of the result set matters less than in

https://doi.org/10.1017/S1351324908004889 Published online by Cambridge University Press

our system. Furthermore, suggestions for query refinements are based on concrete terms (say, 'Iraq'), whereas we suggest the use of a concept ('search by country').

9 Conclusions

We described strategies for selecting and presenting succinct information in spoken dialogue systems. Verbalizing the constraints used in a query is crucial for robustness and usability – in fact, it can be regarded as a special case of providing feedback to the user about what the system has heard and understood (see Traum 1994, for example). The specific strategies we use include 'backing-off' to more general constraints (automatically by the system) or suggesting query refinements. Constraint relaxation techniques have been widely used before, of course, for example in syntactic and semantic processing. The presented paper details how these techniques, when used at the content determination level, tie in with dialogue and generation strategies. Although we focused on the restaurant selection task, our approach is generic and can be applied across domains, provided that the goal of the dialogue is to identify a specific item out of a potentially large set of items. Our language generator uses overgeneration and ranking techniques for surface realization which facilitates variation and alignment with the user utterance. Moreover, we presented evaluation results for general task performance and for the use of suggestions.

Acknowledgments

This work was supported by the U.S. government's NIST Advanced Technology Program. Collaborating partners were CSLI, Robert Bosch Corporation, VW America and SRI International. The major part of this work was conducted while the first author was at CSLI, and the second and third authors at Robert Bosch Corporation. While preparing this paper, Sebastian Varges was supported by the European Commission Marie Curie Excellence Grant for the ADAMACH project (contract no. 022593).

We thank the many people involved in system design, development and evaluation, in particular (but not exhaustively) Baoshi Yan, Zhe Feng, Florin Ratiu, Madhuri Raya, Yao Meng, Matthew Purver, Annie Lien, Tobias Scheideck, Badri Raghunathan, Feng Lin, Rohit Mishra, Brian Lathrop, Zhaoxia Zhang, Harry Bratt, Stanley Peters and Lawrence Cavedon. We would also like to thank the reviewers of this paper.

References

Bratt, E. O., Schultz, K., Peters, S., Chen, T., and Pon-Barry, H. 2005. Empirical foundations for intelligent coaching systems. In *Proceedings of The Interservice/Industry Training, Simulation and Education Conference (I/ITSEC) '05*, Orlando, FL.

Brockmann, C., Isard, A., Oberlander, J., and White, M. 2005. Modelling alignment for affective dialogue. In *Proceedings of the UM'05 Workshop on Adapting the Interaction Style to Affective Factors*, Edinburgh, UK.

https://doi.org/10.1017/S1351324908004889 Published online by Cambridge University Press

Cheng, H., Bratt, H., Mishra, R., Shriberg, E., Upson, S., Chen, J., Weng, F., Peters, S., Cavedon, Y., and Niekrasz, J. 2004. A Wizard-of-Oz framework for collecting spoken human-computer dialogs. In *Proceedings of Interspeech/ICSLP '04*, Jeju Island, Korea.

Chung, G. 2004. Developing a flexible spoken dialog system using simulation. In *Proceedings of ACL '04*, Barcelona, Spain.

Dale, R., and Reiter, E. 1995. Computational interpretations of the Gricean maxims in the generation of referring expressions. *Cognitive Science* **19**: 233–263.

Demberg, V., and Moore, J. 2006. Information presentation in spoken dialogue systems. In *Proceedings of the European Chapter of the ACL (EACL) '06*, Trento, Italy.

Frank, A., Krieger, H.-U., Xu, F., Uszkoreit, H., Crysmann, B., Jörg, B., and Schäfer, U. 2005. Question answering from structured knowledge sources. *Journal of Applied Logic* **5**(2), 20–48; Special Issue on Questions and Answers: Theoretical and Applied Perspectives, Amsterdam.

Gennari, J. H., Musen, M. A., Fergerson, R. W., Grosso, W. E., Crubézy, M., Eriksson, H., Noy, N. F., and Tu, S. W. 2002. The evolution of protégé: An environment for knowledge-based systems development. Technical Report, Stanford University.

Kaplan, J. 1983. Coorperative responses from a portable natural language database query language. In Michael Brady and Robert C. Berwick (eds.), *Computational Models of Discourse*, pp. 167–208. Cambridge, MA: MIT Press.

Kruijff-Korbayova, I., Karagjosova, E., and Larsson, S. 2002. Enhancing collaboration with conditional responses in information-seeking dialogues. In *Proceedings of 6th Workshop on the Semantics and Pragmatics of Dialogue (EDILOG '02)*, Edinburgh, UK.

Kruschwitz, U., and Al-Bakour, H. 2005. Users want more sophisticated search assistants: Results of a task-based evaluation. *Journal of the Americal Society for Information Science and Technology* **56**(13): 1377–1393.

Langkilde, I. 2000. Forest-based statistical sentence generation. In *Proceedings of the North American Chapter of the ACL (NAACL) '00*, Seattle, WA.

Langkilde, I., and Knight, K. 1998. Generation that exploits corpus-based statistical knowledge. In *Proceedings of COLING/ACL-98*, pp. 704–710. Montreal, Canada.

Larsson, S., and Traum, D. 2000. Information state and dialogue management in the TRINDI dialogue move engine toolkit. *Natural Language Engineering* **6**(3–4): 323–340.

Lemon, O., Gruenstein, A., Battle, A., and Peters, S. 2002. Multi-tasking and collaborative activities in dialogue systems. In *Proceedings of 3rd SIGdial Workshop on Discourse and Dialogue*, Philadelphia, PA.

Mirkovic, D., and Cavedon, L. 2005. Practical plug-and-play dialogue management. In *Proceedings of the 6th Meeting of the Pacific Association for Computational Linguistics (PACLING)*, Tokyo, Japan.

Pellom, B., Ward, W., and Pradhan, S. 2000. The CU Communicator: An architecture for dialogue systems. In *Proceedings of Interspeech/ICSLP '00*, Beijing, China.

Pon-Barry, H., Weng, F., and Varges, S. 2006. Evaluation of content presentation strategies for an in-car spoken dialogue system. In *Proceedings of Interspeech/ICSLP '06*, Pittsburgh, PA.

Purver, M., Ratiu, F., and Cavedon, L. 2006. Robust interpretation in dialogue by combining confidence scores with contextual features. In *Proceedings of Interspeech/ICSLP '06*, Pittsburgh, PA.

Qu, Y., and Green, N. 2002. A constraint-based approach for cooperative information-seeking dialogue. In *Proceedings of the International Workshop on Natural Language Generation (INLG '02)*, New York.

Reiter, E., and Dale, R. 2000. *Building Applied Natural Language Generation Systems*. Cambridge, UK: Cambridge University Press.

Rieser, V., and Lemon, O. 2007. Learning dialogue strategies for interactive database search. In *Proceedings of Interspeech*, Antwerp, Belgium.

https://doi.org/10.1017/S1351324908004889 Published online by Cambridge University Press

Traum, D. 1994. *A Computational Theory of Grounding in Natural Language Conversation.* Ph.D. thesis, Computer Science Department, University of Rochester.

Varges, S. 2004. Overgenerating referring expressions involving relations. In *Proceedings of the Third International Conference on Natural Language Generation (INLG '04)*, Brockenhurst, UK.

Varges, S. 2005. Chart generation using production systems (short paper). In *Proceedings of 10th European Workshop on Natural Language Generation*, Aberdeen, Scotland.

Varges, S., and Mellish, C. 2001. Instance-based natural language generation. In *Proceedings of the 2nd Meeting of the ACL (NAACL '01)*, Carnegie Mellon University, Pittsburgh, PA.

Varges, S., and Purver, M. 2006. Robust language analysis and generation for spoken dialogue systems (short paper). In *Proceedings of the ECAI 06 Workshop on the Development and Evaluation of Robust Spoken Dialogue Systems*, Riva del Garda, Italy.

Walker, M. A., Whittaker, S. J., Stent, A., Maloor, P., Moore, J. D., Johnston, M., and Vasireddy, G. 2004. Generation and evaluation of user tailored responses in multimodal dialogue. *Cognitive Science* **28**: 811–840.

Weng, F., Cavedon, L., Raghunathan, B., Mirkovic, D., Bei, B., Pon-Barry, H., Bratt, H., Cheng, H., Schmidt, H., Mishra, R., Lathrop, B., Zhang, Q., Scheideck, T., Xu, K., Hand-Bender, T., Peters, S., Shriberg, L., and Bergmann, C. 2005. A flexible conversational dialog system for MP3 player. In demo session of *HLT-EMNLP 2005*, Vancouver, Canada.

Weng, F., Cavedon, L., Raghunathan, B., Mirkovic, D., Cheng, H., Schmidt, H., Bratt, H., Mishra, R., Peters, S., Upson, S., Shriberg, E., Bergmann, C., and Zhao, L. 2004a. A conversational dialogue system for cognitively overloaded users. In *Proceedings of Interspeech/ICSLP '04*, Jeju Island, Korea.

Weng, F., Cavedon, L., Raghunathan, B., Mirkovic, D., Cheng, H., Schmidt, H., Bratt, H., Mishra, R., Peters, S., Zhao, L., Upson, S., Shriberg, E., and Bergmann, C. 2004b. Developing a conversational dialogue system for cognitively overloaded users. In *Proceedings of the International Congress on Intelligent Transportation Systems*, San Francisco, CA.

Weng, F., Jin, N., Meng, J., and Zhu, Y. 2001. A novel probabilistic model for link unification grammar. In *Proceedings of the 7th International Workshop on Parsing Technologies (ACL/SIGPARSE)*, Beijing, China.

Weng, F., Yan, B., Feng, Z., Ratiu, F., Raya, M., Lathrop, B., Lien, A., Mishra, R., Varges, S., Lin, F., Purver, M., Meng, Y., Bratt, H., Scheideck, T., Zhang, Z., Raghunathan, B., and Peters, S. 2007. CHAT to your destination. In *Proceedings of 8th SIGdial Workshop on Discourse and Dialogue*, Antwerp, Belgium.

Woods, W. A., Kaplan, R. M., and Nash-Webber, B. L. 1972. *The Lunar Sciences Natural Language Information System: Final Report*. BBN Report No. 2378, Bolt Beranek and Newman Inc., Cambridge, MA. (Available from NTIS as N72-28984.)

Zhang, Q., and Weng, F. 2005. Exploring features for identifying edited regions in disfluent sentences. In *Proceedings of the 9th International Workshop on Parsing Technologies (IWPT-05)*, Vancouver, CA.

Zhang, Q., Weng, F., and Feng, Z. 2006. A progressive feature selection algorithm for ultra large feature spaces and its application to edit region identification. In *Proceedings of ACL '06*, Sydney, Australia.

https://doi.org/10.1017/S1351324908004889 Published online by Cambridge University Press