Open Dialogue Management for Relational Databases

Ben Hixon
Computer Science & Engineering
University of Washington
Seattle, WA 98195, USA
bhixon@cs.washington.edu

Rebecca J. Passonneau
Center for Computational Learning Systems
Columbia University
New York, New York, USA
becky@ccls.columbia.edu

Abstract

We present open dialogue management and its application to relational databases. An open dialogue manager generates dialogue states, actions, and default strategies from the semantics of its application domain. We define three open dialogue management tasks. First, vocabulary selection finds the intelligible attributes in each database table. Second, focus discovery selects candidate dialogue foci, tables that have the most potential to address basic user goals. Third, a focus agent is instantiated for each dialogue focus with a default dialogue strategy governed by efficiency. We demonstrate the portability of open dialogue management on three very different databases. Evaluation of our system with simulated users shows that users with realistically limited domain knowledge have dialogues nearly as efficient as those of users with complete domain knowledge.

1 Introduction

This paper presents *open dialogue management.* An open dialogue manager (ODM) generates dialogue states, actions, and strategies from knowledge it computes about the semantics of its domain. A *dialogue strategy* is the procedure by which a system chooses its next action given the current state of the dialogue. The system's *dialogue policy* completely specifies which strategy to use in any dialogue state. Strategies can be handcrafted or learned. Reinforcement learning, the leading method for dialogue strategy learning, can yield powerful results but relies on small sets of states and actions predefined by the researcher. This reliance on domain expertise limits machine learned dialogue managers to the domains for which they were specifically designed, and contributes to the prevalence of handcrafted strategies over machine learning approaches for dialogue management in commercial applications (Paek & Pieraccini, 2008). We argue that open dialogue management, which exploits the semantics and contents of its database to generate actions, states and default strategies, is a step towards a dialogue manager that operates across domains.

As a first step to open dialogue management we present ODDMER (OPEN-DOMAIN DIALOGUE MANAGER), the first dialogue system to generate its own dialogue strategy from relational databases. ODDMER's vocabulary selection module uses supervised learning to determine each table's *intelligible* attributes, those most likely to be in the user's vocabulary. Its focus discovery module finds *candidate dialogue foci,* tables that have the most potential to address basic user goals. Foci are identified with *schema summarization* through a random walk over the database schema that ranks tables by size, linguistic information, and connectivity. For each candidate focus, ODDMER instantiates a *focus agent* that prompts users for values of intelligible attributes ordered by efficiency.

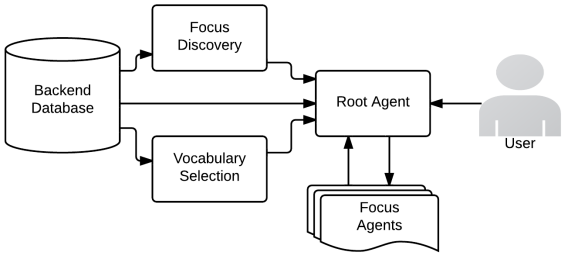

Figure 1. ODDMER uses focus discovery and vocabulary selection to choose its states, actions, and strategy.

Proceedings of NAACL-HLT 2013, pages 1082–1091,
Atlanta, Georgia, 9–14 June 2013.

This paper addresses a particular type of information-seeking dialogue in which the user's goal is to select a tuple from a table. Tuples are identified by *constraints,* attribute-value pairs elicited from a user during the dialogue. A typical user, however, cannot supply all values with equal readiness. For example, attributes such as primary or foreign keys are irrelevant or unintelligible to users. This results in a *vocabulary problem*, a mismatch between system and user vocabulary (Furnas et al., 1987). Furthermore, tables differ in their relevance to users. Tables that contain little semantic information have less potential to address user goals. Dialogue systems for relational databases often rely on manual pre-processing to select the attributes a typical user can most readily supply and identify the tables with the most relevance to basic user goals. An open dialogue system obviates this manual step by exploiting the database semantics.

For example, *Heiskell* is a library database that includes a table for books (BOOK) and a table for book subject headings (HEADING). A typical patron wants a book, not a heading. Due to BOOK's larger size, its greater number of intelligible attributes, and its higher connectivity to other tables, ODDMER recognizes that a BOOK tuple satisfies a more basic user goal. BOOK has 32 attributes, most of which are numeric fields familiar to a librarian but arcane to the user. ODDMER selects the table's intelligible attributes as its vocabulary. It recognizes that a book's author and title are intelligible, but the book's ISBN is not. Consequently, ODDMER will not ask the user for the ISBN.

ODDMER assumes a user of the Heiskell database will be likely to know one or more intelligible attributes of books. ODDMER ranks intelligible attribute-value pairs by their *semantic specificity,* the degree to which they uniquely identify a tuple. To demonstrate the benefit of pre-computing this semantic information, we test ODDMER on three databases with simulated users of two knowledge levels. *Complete-knowledge* users know all attribute values. They have no vocabulary problem, will always be able to supply a requested constraint, and require no vocabulary selection to achieve maximum dialogue efficiency. *Incomplete-knowledge* users have a more realistic vocabulary. They know values for different attributes with different probabilities. Without vocabulary selection, these users have long, inefficient dialogues. Given ODDMER's vocabulary selection and efficient dialogue strategy, these users achieve their goals nearly as efficiently as complete-knowledge users.

2 Related Work

ODDMER is the first dialogue system to examine a database and choose which tables and attributes to use in dialogue. We envision open dialogue management as a suite of domain-independent procedures through which a dialogue manager can exploit its knowledge base. Hastie, Liu, and Lemon (2009) also generate policies from databases. They do not consider multiple tables, and they depend on handcrafted Business Process Models that explicitly specify the dialogue flow for the domain. This limits their method to domains with available models. Polifroni, Chung, and Seneff (2003) also argue for the importance of generic, domain-independent dialogue managers. Their portable information presentation strategies cluster attribute values to summarize database contents for users. Neither of these works considers how to choose attributes or find which domain entities are likely objects of dialogue goals. Chotimongkol and Rudnicky (2008) use unsupervised learning to automatically acquire task-specific information from a corpus of in-domain human-human dialogue transcripts. They require a large corpus whereas we need only the underlying database.

The vocabulary problem has received relatively little attention in dialogue research, and no method to automatically identify intelligible constraints has been previously demonstrated. Demberg and Moore (2006) choose constraints with a user model that records user importance, such as 'price' for a student in a restaurant domain. They require a manually crafted user model and must match model to user. Polifroni and Walker (2006) use attribute entropy to order system initiative prompts, but assume that both the table and the relevant attributes are known a priori. Varges, Weng, and Pon-Barry (2006) develop a WOZ system in which a wizard recommends to real users what constraint to provide that will best narrow down results. Each of these works assumes all constraints are intelligible.

Two recent works concentrate more closely on the vocabulary problem. Janarthanam and Lemon (2010) build a system that determines a user's level of referring expression expertise, but manually determine the set of possible expressions. Selfridge and Heeman (2010) simulate users with different

levels of domain knowledge. A novice has a 10% chance to know any constraint, and an expert a 90% chance. They do not consider users who know different constraints with different probabilities as we do, and do not consider databases that contain attributes likely to be unintelligible to most users.

Reinforcement learning, the leading approach for learning a dialogue strategy, demonstrates powerful results. For example, Rieser and Lemon (2009) find the optimal size of a list of tuples that match a user's constraints and when to display it in different user environments. A dialogue strategy is treated as a *policy*, a function that maps states to actions. Policy optimization is a Markov decision process. Paek and Pieraccini (2008) argue that reinforcement learning is limited by its reliance upon small sets of manually defined states and actions, with no standard method to determine these. ODDMER identifies dialogue states and actions automatically. Its default strategy could be optimized with reinforcement learning.

Portability is an important research area in natural-language interfaces to databases (NLIDBs). An NLIDB parses a user utterance into a logical form, which is transformed into a database query. Users typically know the database structure and contents. TEAM (Grosz, 1983), the first portable NLIDB, questions a domain expert to acquire linguistic knowledge for new databases. More recently, the ORAKEL interface (Cimiano et al., 2008) partially derives a domain-specific lexicon from a generic ontology. Here we do not focus on parsing of user questions, but on the acquisition of dialogue states, actions, and strategies from a database.

3 The ODDMER Dialogue System

ODDMER's *vocabulary selection* module finds each table's intelligible attributes. Its *focus discovery* module identifies candidate foci. A *focus agent* module instantiates dialogue agents for each focus. Their default strategy elicits attribute values from users in order of semantic specificity.

3.1 Vocabulary Selection

A *vocabulary* is the set of words and expressions used to discuss a domain. Domain entities can be identified by their *descriptions*, or sets of attribute-value pairs. In order for a system and a user to profitably engage in a natural language dialogue about database items, descriptions should consist of attributes and values understood by both system and user. We define the *vocabulary selection task* as the automatic selection of attribute-value pairs that the system expects its users will use to describe domain entities.

Successful vocabulary selection solves the vocabulary problem. The vocabulary problem is a bottleneck to portability because the attributes a user is likely to know must be predetermined for existing systems. ODDMER learns a classifier to determine a table's intelligible attributes. An attribute is *intelligible* if its values are in a user's vocabulary. A user interested in a particular item but unfamiliar with the structure of a database is more likely to recognize an intelligible attribute, and to know all or part of the relevant value.

To determine intelligible attributes, ODDMER currently relies on a binary classifier that takes as input the values of each attribute found in a particular instantiation of a relational database. To train the classifier, we labeled a set of 84 attributes belonging to tables taken from the Microsoft AdventureWorks Cycle Company database, a benchmark database packaged with Microsoft SQL Server. An attribute was labeled as intelligible if its values were likely to be known to a user. Four annotators worked independently to label the attributes. Pairwise agreement was 69%, and Krippendorff's alpha (Krippendorff, 1980) was 0.42. The low agreement can be attributed in part to the many ways to interpret the question annotators were to answer. The instructions indicated that the goal was to identify attributes corresponding to common-sense knowledge, but for a given table, annotators were shown all the attributes and asked whether they would know a value. For an employee table, annotators disagreed on attributes such as birthdate, hire date, and organization level. If they had instead been asked whether anyone without access to the table might know a value, there may have been more agreement.

Ratio of unique to total characters in all values
Mean ratio of unique to total characters per value
Ratio of numeric to total characters in all values
Ratio of unique to total values
Ratio of unique to total words in all values
Total number of characters in all values

Table 1. Representative features for attribute classification used in the best-performing intelligibility classifier.

The training data for the classifier consisted of 67 attributes that at least three annotators agreed on (22 intelligible, 45 not intelligible); pairwise agreement was 0.81 and Krippendorff's alpha was 0.61. They represented 8 tables and contained a total of 393,520 values, 123,901 of which were unique. For each attribute we extracted 17 features to represent the linguistic *expressiveness* of the attributes' values. An attribute whose values are more like natural language is more intelligible. Table 1 lists the features of the best classifier.

We tested several binary classifiers in Weka (Hall et al., 2009). ADTree (Freund & Mason, 1999) with ten boosting iterations performed best, with 91% recall and 91% precision under 10-fold cross-validation. However, the ADTree models were overfitted to the AdventureWorks domain. The RIPPER rule-learning algorithm (Cohen, 1995) achieved 77% precision and 78% recall. Because its learned model of three simple rules generalizes better to our domains, ODDMER uses the RIPPER intelligibility classifier.

Given a table, the vocabulary selection module returns which of its attributes should be in the system's vocabulary. For the Heiskell Library domain, the 32 attributes of the BOOK table include many internal codes understood by librarians but not by users. Only the seven attributes shown in Table 2 are classified as intelligible. Dialogues with only intelligible attributes should be more efficient for users with incomplete domain knowledge, because they will be more likely to know their values.

ODDMER's vocabulary selection module also computes the *semantic specificity* score of each attribute (Hixon, Passonneau, & Epstein, 2012). Semantic specificity rates an attribute on a scale from 0 to 1 according to how unambiguously its values map to rows in the database. More specific attributes are expected to be more efficient prompts. Table 2 lists the specificity values of the intelligible attributes for BOOK.

Intelligible Attributes	Specificity
ANNOTATION	0.958
TITLE	0.878
SORTTITLE	0.878
AUTHOR	0.300
NARRATOR	0.018
PUBLISHER	0.016
SERIES	0.003

Table 2. Intelligible attributes for BOOK sorted by specificity. (SORTTITLE is a duplicate of TITLE.)

3.2 Candidate Dialogue Focus Discovery

Information-seeking dialogues address diverse dialogue goals. For example, users may want to identify a tuple in a table ("I want a certain book by Stephen King."), retrieve the value of an attribute for a given tuple ("Is my plane on time?" "Who wrote *Moby Dick*?"), aggregate over a set of tuples ("How many Italian restaurants are in this neighborhood?"), or compare values of different tuples ("Which restaurant is more expensive?"). Each of these dialogue goals represents a distinct information need. However, not all possible information needs in a domain are equally likely. For example, a user is unlikely to ask for the value of a primary key attribute, or to select a tuple from a table that contains only primary and foreign keys. A dialogue system should place less priority on addressing these peripheral dialogue goals.

Given a particular domain, we assume that some goals are more *basic* than others. For example, the basic function of a library is to provide books to borrowers. Some libraries will also provide other material, or perform reference functions, but these are less basic. This notion of a basic goal is related to the basic categories proposed by Rosch (1978), who claimed that not all categories are equally useful for cognition. Basic categories are more differentiated from other categories, and have attributes that are common to all or most members of the category, thus provide us with more information (the principle of *cognitive economy*). Basic categories also mirror the structure of the perceptual and functional attributes of the natural world, thus serve us better in our daily activities. Typically a domain expert will identify the basic dialogue goals in a domain, but we suggest that the basic dialogue goals are *discoverable* in the underlying database. While a difficult problem, we are motivated by work in the database literature to identify and rank the most likely queries for an arbitrary database (Jayapandian & Jagadish, 2008).

We approach the problem of recovering dialogue goals from a database by restricting our attention to the tuple selection task, a commonly studied type of information-seeking dialogue in which the user's goal is to select a tuple from a table. A relational database typically consists of multiple tables, and each table can satisfy different user goals. Given a database composed of multiple tables, an open dialogue system calculates which

tables are larger, have more natural language content, and greater connectivity to other tables. We refer to these tables as *candidate dialogue foci.* This notion of candidate focus for a dialogue is similar to focus of attention (Grosz & Sidner, 1986) in that information-seeking dialogues can be segmented to reflect the table both participants focus their attention on at a given time. We denote the task of identifying candidate foci in a relational database as *focus discovery.*

ODDMER's focus discovery module returns an ordered list of candidate foci, the *focal summary.* The highest ranked focus is the most relevant to basic user goals, those goals that pertain to the most information-rich and intelligible table. For our tuple-selection task, the highest-ranked focus is the table from which the system predicts a user will most likely want to select a tuple. A system that begins a dialogue by first mentioning the most relevant tables communicates the structure of the database better than does a system that lists all tables in a random order. Users with more specialized goals may be interested in more peripheral tables. For these users, more effort will be required to establish a dialogue focus: several tables may be proposed by the system and rejected before the user agrees to consider a given table. In tests with real users, we would expect them to find it acceptable for a specialized goal to take more effort than a basic goal.

We use *schema summarization* to find candidate focus tables. According to Yu and Jagadish (2006), a schema summary should convey a concise understanding of the underlying database schema. They identify table importance and coverage as criteria for good schema summaries. Their summaries for XML databases rank entities with higher cardinality (number of rows) and connectivity (number of joins) as more important. Yang, Procopiuc, and Srivastava (2009) extend schema summarization to relational databases. We closely follow the Yang algorithm but make modifications for dialogue to account for attribute intelligibility.

A database *schema* is an undirected graph G = <R,E> where each node r ∈ R corresponds to a table in a database and each edge e ∈ E denotes a join between two tables. A *schema summary* is a set of the most important nodes in the schema. Yang and colleagues compute the importance of a table as a function of its size, total entropy of its attributes, and connectivity to other tables. To incorporate connectivity, they employ a random walk over the schema graph. The most important tables maintain the highest information content in the steady state of a random walk over the schema.

A significant feature of their algorithm is that a table's high-entropy attributes largely determine its importance. It is possible to artificially inflate a table's importance under the Yang algorithm by introducing a new column of distinct integer values; numeric and linguistic values contribute equally to importance. For dialogue applications, this is undesirable. A table with more intelligible attributes is a more likely candidate focus because it can more readily be discussed. We therefore modify the Yang algorithm to compute table *verbality.* Verbality is similar to importance except that where Yang and colleagues use all attributes, we use intelligible attributes identified by vocabulary selection.

A table's verbality score is a function of its cardinality, the entropy of its intelligible attributes, and its connectivity to other tables. We apply vocabulary selection to find natural language attributes. To calculate the verbality of a table T, let A be the attributes of T and let $A' \subseteq A$ be those attributes of T whose values are intelligible, found by the classifier described previously. For BOOK, A' consists of the attributes shown in Table 2. Define V, the verbal information content of a table, as

$$V(T) = \log(|T|) + \sum_{a \in A'} H(a)$$

where $|T|$ is the cardinality of the table and $H(a)$ is the entropy of the attribute a in A'. The entropy of a is given by $H(a) = -\sum_{k \in K} p_k \log p_k$ where K is the set of distinct values of a and p_k is the fraction of rows in T for which $a=k$. If table T has no joins, $V(T)$ is the final verbality score of T. Table 3 shows $V(T)$ for each T in *Heiskell*.

To incorporate connectivity into verbality, we create a matrix of transition probabilities between every pair of nodes in the schema and determine which table maintains the highest information. Let $J \subseteq A$ be the attributes of T that join to other tables. The *information transfer* (*IT*) over the join $j \in J$ is

$$IT(j) = \frac{H(j)}{V(T) + \sum_{a \in A} q_a H(a)}$$

where q_a is the number of joins in which attribute a participates. Let P(T,R) be the transition probability from table T to table R. For $T \neq R$, P(T,R) is the sum of $IT(j)$ for all joins j between T and R. These

probabilities represent the flow of information between tables over their joins. The diagonal entries of the transition matrix are given by

$$P(T,T) = 1 - \sum_{T \neq R} P(T,R),$$

the information that stays within table T. We then define the *verbality* of table T_i to be the i^{th} element in the stable distribution of a random walk over the NxN matrix whose elements are $P(T_i,T_j)$ for $i,j \in N$. We follow Yang and colleagues and use power iteration to find the stable distribution.

T	V(T)	Verbality(T)
Book	88.2	45.4
Heading	31.7	22.4
BibHeadingLink	19.2	23.6
CirculationHistory	17.9	24.9
Holding Stats	15.2	24.0
Patron Properties	12.7	21.9
Reserve	12.2	25.5
Patron	9.0	18.6

Table 3. Verbalities of *Heiskell*. V(T) is the verbal information of T. Verbality(T) incorporates connectivity.

Table 3 illustrates the verbalities of *Heiskell* before and after information transfer. BOOK clearly dominates. Before information transfer there is more verbal information V in BOOK than in the next four tables combined. After information transfer reaches a steady state, its connectivity with other tables increases their verbality, but BOOK remains the leading candidate by a large margin. HEADING's verbality decreases sharply after information transfer because of its low connectivity.

The focus discovery module returns a *focal summary,* a list of the top k tables with the highest verbalities. The focal summary is similar to the intensional summary described by Polifroni and Walker (2008), which communicates the contents of a single table to a user. A key difference is that the focal summary pertains to the entire database.

ODDMER is currently limited to the table and attribute labels assigned by the database designer. For example, the Heiskell 'BOOK' table was labeled 'BIBREC' by the database designers, for Bibliographic Record. We renamed this table prior to its use as a backend for the dialogue system. But ODDMER has no way to determine if labels are meaningful. In many cases there is a disincentive towards meaningful table names to avoid conflicts with SQL keywords. Future work will explore how to infer more meaningful table and attribute labels from a database instantiation, for example by consulting external knowledge bases to predict the entity a table represents.

3.3 Focus agent generation

Focus discovery fits naturally into an agent-based or agenda-based approach to dialogue management (Bohus & Rudnicky, 2009; Nguyen & Wobcke, 2005). At the onset of a dialogue, ODDMER's root agent announces the focal summary and prompts the user for her goal. Upon receipt of a user reply, it launches the appropriate *focus agent*, a finite state machine based on Information State Update (Traum & Larsson, 2003), whose parameters are a table, its list of intelligible attributes, and their computed specificity scores. The agent elicits constraints from the user until the current goal is satisfied or the user abandons the focus. Control over the dialogue then returns to the root, which queries the user for a new focus or ends the dialogue. Figure 2 shows a sample dialogue.

1. S: Hello, I can help you find a Book. Would you like to find a Book?
2. U: I'd like a Book.
3. S: Do you know the annotation?
4: U: No.
5: S: Do you know the title?
6. U: It's *Gone with the Wind*

Figure 2. ODDMER's root agent gives the focal summary (k=1) in line 1. The BOOK focus agent launches in line 3 and prompts for the value of the most specific intelligible attribute.

The default strategy of the focus agent is to elicit the most specific, intelligible constraints first. While intelligible attributes are more likely to be known by users, they are not equally valuable as item descriptions. As shown in Table 2, the specificities of NARRATOR, PUBLISHER, and SERIES are so low that a strategy involving them will likely lead to inefficient dialogues and unsatisfied users. The focus agent therefore orders its prompts for constraints by their specificity, and requests the most specific attributes first. Because specificity is a function of the database instantiation and not user knowledge, we expect that this strategy will lead to shorter, more efficient dialogues for all users.

The dialogue starts with the root agent in control of the dialogue. The root agent announces the focal summary, the k tables with highest verbality.

Here we let k=1. The root agent parses the user's reply to determine the focus of interest and launches the appropriate agent. The agent interacts with the user to find a tuple from the table. Its default strategy elicits constraints from the user in order of semantic specificity. Because semantic specificity can apply to combinations of attributes, future work will investigate tradeoffs between efficiency and user effort in prompts for multiple attributes.

The focus agent queries the database upon receipt of each new constraint. If the return size is small enough (here, a single tuple) it announces the result. Otherwise it continues to elicit constraints until all intelligible attributes have elicited values, at which point it announces all matching results.

ODDMER deals with three goal setbacks: (1) disambiguation, in which a query is under-constrained and matches multiple tuples; (2) over-constrained queries that have no matching tuple; and (3) attributes whose values are unknown to the user. The third setback is particularly prevalent in real-world databases. Our system addresses these setbacks with specific, intelligible vocabulary.

4 Evaluation

ODDMER finds foci and vocabulary for any relational database. We evaluate it on three very different domains. These databases were chosen for their variety, and are not equally suitable for dialogue. Our primary domain of consideration is *Heiskell,* the database of the Heiskell Library. *Heiskell* has eight tables. The largest table is CIRCULATION HISTORY, which contains 16 attributes with 244,072 rows. However, focus discovery identifies BOOK as the top candidate focus, which matches our intuition. Though BOOK is smaller at only 71,166 rows, it has 32 attributes of which seven are classified as intelligible. The classifier finds none of CIRCULATION HISTORY's attributes intelligible. Manual inspection revealed CIRCULATION HISTORY to consist primarily of alphanumeric codes relevant to the library rather than to users.

The second domain we consider is *Grocery*, a small supermarket database used as a teaching tool in an undergraduate class at Hunter College. Grocery has 20 tables. Their cardinalities range from 7 to 197 rows. The top focus in *Grocery* represents the products sold in the store. It was the largest table with the greatest connectivity, and makes intuitive sense; it is the table a supermarket customer would most likely want to talk about.

To challenge our system we consider *Eve,* a freely available database for the virtual game Eve Online, a massively multiplayer online role-playing game with over 400,000 subscribers. *Eve* has 78 tables and the game's active community regularly accesses it to determine in-game goals and objects of interest. *Eve* is a challenge for ODDMER because it primarily contains numeric data for objects in the game world. These numeric attributes are of great interest to players but confound our assumption that dialogue goals correlate with high verbality. Moreover, connectivity plays no role in table verbality because *Eve* contains no joins. Focus discovery on *Eve* identifies INVTYPES, a table that represents in-game inventory items, as the best focus, even though vocabulary selection identifies only three of its 15 attributes as intelligible. The 12 unintelligible attributes were all numeric. In general, focus discovery and vocabulary selection proved less effective on *Eve* than on *Heiskell*. In *Eve* the verbality scores of the top tables were close together without one outstanding focus candidate.

4.1 Simulating the vocabulary problem

A typical evaluation of a spoken dialogue system provides users with all the information needed to carry out a dialogue. Such a *completely knowledgeable* user can unrealistically describe objects in the domain with the same vocabulary that the system uses. This means it does not experience the vocabulary problem. To test vocabulary selection, we simulate users with *incomplete* domain knowledge. In contrast with Selfridge and Heeman (2010), our limited-knowledge users are more likely to know some attributes than others.

User simulation is often used to stand in for real user dialogues, but it is a concern whether the dialogues are sufficiently realistic (Schatzmann, Georgila, & Young, 2005). Here, we use simulation to exercise each dialogue system with a large number of cases in a highly controlled fashion. For simulated users, we can specify exactly what each user knows about the domain, thus simulation makes it possible to hold everything else the same while contrasting users with complete versus incomplete domain knowledge. We view this as a preliminary step towards evaluation with real users, which we hope to do in future work.

	Heiskell	*Grocery*	*Eve*
C/N/R	15.9 ± 0.4	11.1 ± 0.2	16.3 ± 0.4
C/N/S	9.0 ± 0.0	9.0 ± 0.0	9.0 ± 0.0
C/V/R	11.5 ± 0.2	11.1 ± 0.3	10.6 ± 0.1
C/V/S	9.5 ± 0.1	9.0 ± 0.0	9.0 ± 0.0
L/N/R	25.3 ± 0.8	13.1 ± 0.2	17.7 ± 0.5
L/N/S	16.5 ± 0.6	9.3 ± 0.1	9.4 ± 0.1
L/V/R	12.4 ± 0.2	12.2 ± 0.1	10.7 ± 0.1
L/V/S	11.3 ± 0.2	9.7 ± 0.1	9.0 ± 0.0

Table 4. Mean dialogue length across domains. C/*/*: complete-knowledge user; L/*/*: incomplete-knowledge user; */N/*: no vocabulary selection; */V/*: with vocabulary selection; */*/R: random order; */*/S: order by specificity. Intervals are 95% confidence.

Our simulated user knows each attribute's value with a different probability. Ideally we might estimate these probabilities from a language model of a corpus in the domain. Unfortunately our domains contain many obscure names and non-verbal values for which we need non-zero probabilities. Instead, we estimate a probability for attribute value knowledge by calculating the frequency of total value occurrences in a subset of the New York Times portion of the English Gigaword corpus (Parker et al., 2011), a 4 million word corpus of news articles. These frequency values could have been used during vocabulary selection but we chose to reserve them for evaluation.

The probability that the limited-knowledge user knows a particular value is the normalized frequency that the attribute's values appear in Gigaword. We tokenized attribute values in our databases to remove punctuation and case. We ignored word order so that, for example, the author values "King, Stephen" and "Stephen King" are equivalent. For each value, we counted the articles in which all the value's tokens co-occurred. For each attribute, we took the sum of these counts over all its values, and took its log to represent the probability that the user knows that attribute. We then normalized by the log of the highest-frequency attribute to enforce our assumption that the user usually knows at least one piece of information about her goal. This method is robust to attributes with missing values. Probabilities for the BOOK attributes were 100% for TITLE, 78% for AUTHOR, 75% for PLACE PUBLISHED, and 73% for PUBLISHER. ISBN has a 0% probability because none of its values occur in the corpus. ANNOTATIONS, whose values are brief plot descriptions of each book, has a low 37%. Although its values contained many common words, the words in a single annotation rarely co-occurred in one article.

4.2 Testing the impact of domain knowledge

We evaluate dialogue efficiency with two simulated users as measured by number of turns. The first user, C, has complete domain knowledge and always knows every constraint. The second, L, has limited, incomplete domain knowledge. When confronted with the focal summary, the simulated user always chooses the top suggested focus. The dialogue ends when either a single tuple matching the constraints is found, or all constraints have been requested, in which case all matching tuples are announced. We measure average dialogue length of 1000 simulated dialogues for each user with vocabulary selection (V) and without (N), and with prompts ordered randomly (R) or by specificity (S). Table 4 shows the results. ANOVAs of all pairs of comparisons were highly significant.

The longest dialogues for both users occur without vocabulary selection and with prompts in random order (*/N/R). The more attributes there are, the longer it takes a random order to achieve a constraint combination that forms a key, so C has long dialogues even though it knows every constraint. L experiences much longer dialogues because it is prompted for inefficient constraints, and is unlikely to know most of them. This difference is particularly noticeable in *Heiskell*. On average, L's dialogues are ten turns longer.

Ordering prompts by specificity without vocabulary selection (*/N/S) yields a sharp increase in efficiency for both users. C's dialogues achieve the minimum number of turns because it is immediately asked for the most specific constraint, which it always knows. In *Heiskell*, specificity decreases L's average length from 25.3 to 16.5, a large increase in efficiency but still much worse than C. For *Eve*, L performs better in the absence of vocabulary selection. Specificity alone brings its average efficiency near optimum. This is because for *Eve*'s INVTYPES table, the most specific intelligible attribute is the item's NAME, which our domain knowledge model predicts L will always know.

Vocabulary selection is more effective than specificity for L on *Heiskell*. L is much more likely to know the selected attributes and its efficiency increases even when prompted for intelligible attributes in a random order. Vocabulary selection is

less effective than specificity for C because C knows every attribute, but in general the intelligible attributes are also more specific, so selection increases C's efficiency even with random prompts. Vocabulary selection combined with specificity (*/V/S) leads to a small *decrease* in efficiency for C on *Heiskell* over specificity alone. This is because the most specific intelligible attribute is slightly ambiguous, and C must occasionally supply extra constraints to disambiguate. However, with both specificity and vocabulary selection, L achieves a mean dialogue length of 11.3, requiring only two turns more than C to order a book.

For *Eve*, vocabulary selection and order-by-specificity are each effective individually, and yield similar dialogues for both L and C. This is because INVTYPES has only three intelligible attributes, so the dialogue ends after at most three prompts. Our domain knowledge model predicts close to 100% knowledge for two of these.

A comparison of the order-by-specificity strategy used here with the order-by-entropy strategy described by Polifroni and Walker (2006) yielded no significant difference in dialogue length. The two strategies produce similar attribute orders.

5 Conclusion and Open Questions

We have demonstrated an open dialogue management system, ODDMER, which formulates a dialogue strategy by computing *metaknowledge* about its database: table verbality, attribute intelligibility, and attribute specificity. Candidate dialogue foci are the tables with high verbality. For each candidate focus, ODDMER chooses an intelligible domain vocabulary and generates a default strategy that orders prompts by specificity. A simulated user facing the vocabulary problem achieves more efficient dialogues with vocabulary selection. Our method works well on the Heiskell Library database, which has a particularly prominent candidate focus showing a clear separation between intelligible and unintelligible attributes. Focus discovery and vocabulary selection are less effective for numeric databases without clear dialogue goals. For example, *Eve*'s top focus scored the highest verbality, even though the table contained only three intelligible attributes.

Questions that arise from this work include how to extend focus discovery and vocabulary selection to numerical databases, how to extract strategies for goals other than tuple-selection from a database, and how to automatically infer intelligible table and attribute labels. We are also interested in discovery of less rigid dialogue goals, for example, a library patron who would be satisfied by an alternative book, and goals involving information aggregation where user utterances map to sophisticated queries. We would like to investigate how optimal policies learned through reinforcement learning vary across domains. Future work will also scale to mixed-initiative open dialogue management, explore more sophisticated models of user domain knowledge, and evaluate portability on more databases.

ODDMER uses the semantics of its domain representation to discover what to talk about and how to talk about it. We envision a rich toolkit that enables a system to explore its database for knowledge to exploit in collaborative dialogues with its users.

Acknowledgements

The first author was supported in part by NSF grant IIS-0803481, ONR grant N00014-11-1-0294, and DARPA contract FA8750-09-C-0179. The second author was supported by the NSF grant IIS-0745369. This research was carried out at Columbia University. We thank Susan Epstein, Oren Etzioni, Wlodek Zadrozny, and the anonymous reviewers for helpful comments. We thank Luis Gravano for valuable literature suggestions, and Susan Epstein for use of the Grocery database.

References

Bohus, D., & Rudnicky, A. (2009). The RavenClaw Dialog Management Framework: Architecture and systems. *Computer Speech and Language 23*(3), 332-361.

Chotimongkol, A., & Rudnicky, A. I. (2008). *Acquiring Domain-Specific Dialog Information from Task-Oriented Human-Human Interaction through an Unsupervised Learning*. Paper presented at the Conference on Empirical Methods in Natural Language Processing (EMNLP '08).

Cimiano, P., Haase, P., Heizmann, J., Mantel, M., & Studer, R. (2008). Towards Portable Natural Language Interfaces to Knowledge Bases - The case of the ORAKEL system. *Data & Knowledge Engineering, 65*(2), 325-354.

Cohen, W. W. (1995). *Fast Effective Rule Induction.* Paper presented at the Twelfth International Conference on Machine Learning.

Demberg, V., & Moore, J. D. (2006). *Information Presentation in Spoken Dialogue Systems.* Paper presented at the 11th Conference of the European Chapter of the Association of Computational Linguistics (EACL 2006).

Freund, Y., & Mason, L. (1999). *The Alternating Decision Tree Learning Algorithm.* Paper presented at the Sixteenth International Conference on Machine Learning (ICML).

Furnas, G. W., Landauer, T. K., Gomez, L. M., & Dumais, S. T. (1987). The Vocabulary Problem in Human-System Communication. *Communications of the ACM, 30*(11), 964-971.

Grosz, B. J. (1983). *TEAM: a Transportable Natural-Language Interface System.* Paper presented at the First conference on Applied natural language processing.

Grosz, B. J., & Sidner, C. L. (1986). Attention, intentions, and the structure of discourse. *Computational Linguistics, 12*(3), 175-204.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The WEKA Data Mining Software: An Update. *SIGKDD Explorations, 11*(1).

Hastie, H., Liu, X., & Lemon, O. (2009). *Automatic Generation of Information State Update Dialogue Systems that Dynamically Create Voice XML, as Demonstrated on the iPhone*. Paper presented at the 10th Annual SIGdial Meeting on Discourse and Dialogue (SIGdial 2009).

Hixon, B., Passonneau, R. J., & Epstein, S. L. (2012). *Semantic Specificity in Spoken Dialogue Requests.* Paper presented at the 13th Annual SIGdial Meeting on Discourse and Dialogue (SIGdial 2012).

Janarthanam, S., & Lemon, O. (2010). *Learning to Adapt to Unknown Users: Referring Expression Generation in Spoken Dialogue Systems.* Paper presented at the 48th Annual Meeting of the Association for Computational Linguistics (ACL 2010), Uppsala, Sweden.

Jayapandian, M., & Jagadish, H. V. (2008). *Automated creation of a forms-based database query interface*. Paper presented at the 34th international conference on Very large data bases (VLDB '08).

Krippendorff, K. (1980). *Content Analysis: An Introduction to its Methodology*. Beverly Hills, CA: Sage Publications.

Nguyen, A., & Wobcke, W. (2005). *An Agent-based Approach to Dialogue Management in Personal Assistants*. Paper presented at the 10th international conference on Intelligent user interfaces (IUI '05), New York.

Paek, T., & Pieraccini, R. (2008). Automating Spoken Dialogue Management Design Using Machine Learning: An Industry Perspective. *Speech Communication, 50*, 716-729.

Parker, R., Graff, D., Kong, J., Chen, K., & Maeda, K. (2011). *English Gigaword Fifth Edition*. Philadelphia: Linguistic Data Consortium.

Polifroni, J., Chung, G., & Seneff, S. (2003). *Towards the Automatic Generation of Mixed-Initiative Dialogue Systems from Web Content.* Paper presented at the Eurospeech 2003.

Polifroni, J., & Walker, M. (2006). *Learning Database Content for Spoken Dialogue System Design.* Paper presented at the 5th International Conference on Language Resources and Evaluation (LREC).

Polifroni, J., & Walker, M. (2008). *Intensional Summaries as Cooperative Responses in Dialogue: Automation and Evaluation.* Paper presented at the ACL-HLT.

Rieser, V., & Lemon, O. (2009). Does this list contain what you were searching for? Learning Adaptive Dialogue Strategies for Interactive Question Answering. *Natural Language Engineering, 15*(1), 55-72.

Rosch, E. (1978). Principles of Categorization. In E. Rosch & B. Lloyd (Eds.), *Cognition and Categorization* (pp. 27-48). Hillsdale, NJ: Lawrence Erlbaum.

Schatzmann, J., Georgila, K., & Young, S. (2005). *Quantitative Evaluation of User Simulation Techniques for Spoken Dialogue Systems*. Paper presented at the 6th SIGdial Workshop on Discourse and Dialogue.

Selfridge, E., & Heeman, P. (2010). *Importance-Driven Turn-Bidding for Spoken Dialogue Systems.* Paper presented at the 48th Annual Meeting of the Association for Computational Linguistics (ACL 2010).

Traum, D., & Larsson, S. (2003). The Information State Approach to Dialogue Management. In Jan van Kuppevelt and Ronnie Smith (Eds.), *Current and new directions in discourse and dialogue.* Kluwer*:* 325-353.

Varges, S., Weng, F., & Pon-Barry, H. (2006). *Interactive Question Answering and Constraint Relaxation in Spoken Dialogue Systems.* Paper presented at the 7th Annual SIGdial Meeting on Discourse and Dialogue (SIGdial 2006).

Yang, X., Procopiuc, C. M., & Srivastava, D. (2009). *Summarizing Relational Databases*. Paper presented at the 35th international conference on Very large data bases (VLDB '09).

Yu, C., & Jagadish, H. V. (2006). *Schema Summarization*. Paper presented at the 32nd international conference on Very large data bases (VLDB '06).