PCQL: A Formalism for Human-Like Preference Dialogues*

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

Abstract

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

1 Introduction

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

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

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

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

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

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

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

2 Human-Like Preference Dialogue

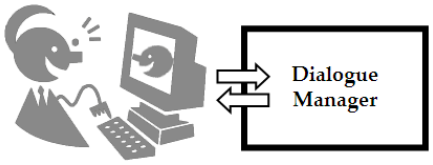
A system of categories describing preference dialogues needs to consider basic dialogue strategies, distribution of dialogue acts, and the way domain entities are referred to. As an empirical base for characterising preference dialogue we have previously collected a human-human dialogue corpus in the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

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

[1] Preference Conversational Query Language.

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

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



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

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

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

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

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

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

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

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

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

When an initial set of preferences have been accumulated, preferences may be refined by introducing comparative statements in phase 3 (e.g. utterance U5 as response to S5a/S5b in Figure 1). Initiative in the third phase is not as clear-cut as in the previous two. The corpus indicates that about half of the recommenders re-gained more control over initiatives

in phase 3 and asked customers comparative questions. The other half simply acknowledged comparative preferences as they were stated by customers. For dialogue system strategy design, this behaviour is thus an open choice. Both approaches are "human-like" using the human-human dialogue corpus as guideline.

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

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

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

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

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

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

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

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

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

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

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

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

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

3 PCQL

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

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

3.1 FP State Formulas

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

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

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

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

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

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

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

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

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

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

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

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

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

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

Factual State Mapping

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

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

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

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

Preference State Mapping

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

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

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

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

3.2 PCQL Action Statements

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

4 Using PCQL in Conversational Recommender Systems

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

4.1 Architecture

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

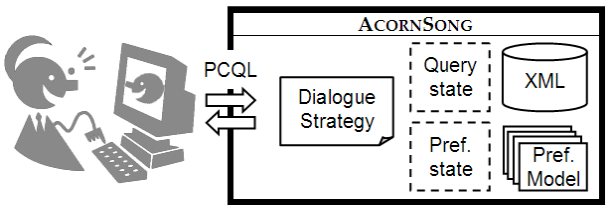


Figure 2: ACORNSONG's dialogue manager sub-modules. The preference state is modelled in preflets. Preflets are data structures used in ACORNSONG that cater for domain preference variation based on FP state formula mappings.

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

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

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

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

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

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

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

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

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

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

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

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

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

4.2 Modeling Preferences using PCQL

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

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

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

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

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

5 Summary

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

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

References

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

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

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

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

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

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

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

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

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

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

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