ISBN 978-83-60810-22-4

Performing Conjoint Analysis within a Logic-based Framework

Adrian Giurca, Ingo Schmitt

Dept. of Databases and Information Technology
Brandenburg University of Technology
P.O. 101344, 03013 Cottbus, Germany
Email: {giurca, schmitt}@tu-cottbus.de

Daniel Baier

Dept. of Marketing and Innovation Management
Brandenburg University of Technology
P.O. 101344, 03013 Cottbus, Germany
Email: daniel.baier@tu-cottbus.de

***Abstract*—Conjoint Analysis is heavily used in many different areas: from mathematical psychology, economics and marketing to sociology, transportation and medicine trying to understand how individuals evaluate products/services and as well as on predicting behavioral outcomes by using statistical methods and techniques. Nowadays is not much agreement about best practice, which in turn has led to many flavors of CA being proposed and applied. The goal of this paper is to offer a solution to perform Adaptive Conjoint Analysis inside CQQL, a quantum logic based information framework. We describe an algorithm to compute a logical CQQL formula capturing user preferences and use this formula to derive decision rules.**

I. Introduction and Motivation

NOWADAYS, the term *Conjoint Analysis* (CA) is used in many different ways. While in the past it was mostly on the interest of marketers and psychologists, today's variants are used in many fields including applied economics, sociology, transportation and medicine. The origins of Conjoint Analysis come from developments in several disciplines, most notably economics ([27], [26]) and mathematical psychology ([30], [31], [2]).

Much of the Conjoint Analysis work was used trying to understand how individuals evaluate products/services and form preferences (see, [18], [22], [33] and possibly others). In the last thirty years the CA literature focused more on predicting behavioral outcomes by using statistical methods and techniques ([3]) and this resulted in a widespread variation in CA practice. Recently, applications in innovation market were developed ([4]). The result today is that there is not much agreement about best practice, which in turn has led to many flavors of CA being proposed and applied.
The regular case involves a compositional model (see [17], [20], [16]) where the respondents provide the necessary information to be able to compute *local utilities*. Various ACA interpretations define a *conjoint*(or *utility*) function U that aggregates the local utilities to an *overall utility* i.e., if o is a product representation then $U(o)$ denotes the overall utility of this product. The main condition for such a function is to be monotonic with respect of preferences i.e., $U(o_1) \geq U(o_2)$ whenever $o_1 \succeq o_2$ (see [17], [8], [9]). There is a large literature concerning the way the overall score of a product should be computed (see [3], [13], [17], [19], [16], [20], and many other). A well known model for such a utility function is the additive linear model (see [3] for a survey of models). Basically, the overall utility is an additive linear combination on local utilities adjusted with attribute weights and compensated with a constant depending on interview:

$$U_p(o_j) = \mu + \sum_{k=1}^{n} \sum_{l=1}^{n_k} \beta_{kl} \cdot x_{jkl} + e_j$$

where μ is the mean preference value across all profiles, $U_p(o_j)$ – is the total score on product profile o_j with respect to respondent p, β_{kl} – is the weight of value a_{kl} of attribute A_k, $x_{jkl} = \begin{cases} 1, & if\ o_j.A_k = a_{kl} \\ 0, & otherwise \end{cases}$ and e_j is the measurement error.
Therefore the problem reduces to find all β_{kl} and μ. We need estimates and not crisp values because we cannot offer all possible product profiles in a user interview simply because they can be too many([22]). Traditional conjoint uses full profiles (complete product descriptions) as basis for user surveys, but such an approach suffers by description complexity in the presence of many attributes ([22], [23], [24] and [25]), the adaptive conjoint introduces a "partial evaluation" i.e., requesting respondents to evaluate only partial profiles, named *stimuli* by using trade-off matrix approaches, with emphasis on pair comparisons.

A solution of the above model will entitle experts to compute a measure of acceptance of a product by the consumer. However, such a solution is not able to provide explanations on product features and how they may influence the consumer final decision. Logical languages offers qualitative and symbolic methods to complement these standard approaches of economic decision theory.

The goal of this paper is to offer a heuristic to perform Adaptive Conjoint Analysis inside a logic based framework. We use the Commuting Quantum Logic Language,

This work is supported by German Federal Ministry of Education and Research, ForMaT project (Forschung für den Markt im Team), Phase II, Innovationslabor: Multimediale Ähnlichkeitssuche zum Matchen, Typologisieren und Segmentieren, http://www.unternehmen-region.de/de/4818.php
and by DFG Project SQ-System: Entwicklung von Konzepten für ein quantenlogikbasiertes Retrieval-Datenbank-Anfragesystem: Anfragesprache, interaktive Suchformulierungsowie effiziente Anfrageauswertung

TABLE I
PRODUCT ATTRIBUTES AND USER RATINGS

AttrId	Attr Weight	Attr. Name	Attr. Value	ValueId	Value Weight
1	6	Operating system	Android	1	7
			Windows Phone	2	6
			other(proprietary)	3	3
2	7	WiFi	yes	1	6
			no	2	3
3	7	Screen size	less than 3.5"	1	3
			3.5"-4.0"	2	7
			greater than 4.0"	3	8
4	5	Battery life	12 hours	1	9
			6 hours	2	6
			4 hours	3	5
			2 hours	4	1
5	8	PriceLevel	less than 150 EUR	1	9
			150 EUR - 250 EUR	2	8
			250 EUR - 500 EUR	3	5
			greater than 500 EUR	4	0

CQQL([35]), to learn a logical formula U such that the CQQL evaluation of this formula against a specific set of database objects (product profiles) should monotonically correlate with the user preferences on objects i.e. if o_1 and o_2 are two product profiles $eval(U, o_1) \geq eval(U, o_2) \Leftrightarrow o_1 \succeq o_2$. Next section shows an illustrating example of our approach.

A. A simple scenario: Which smartphone they may prefer?

Example 1 (Which smartphone do you prefer?): A smartphone manufacturer aims understanding which smartphone is preferred by a specific target group. It performs adaptive conjoint analysis with CQQL using a product decomposition and user scores to compute a logical formula fulfilling the user preferences. This formula can be easily interpreted towards obtaining design decisions.
The initial data consists of a product decomposition into a set attribute values (e.g., "Operating System" is a product attribute with three possible values: "Android", "Windows Phone" and "other(proprietary)") and an initial user rating of attributes and values (e.g., attribute "WiFi" has score 7 and value "Android" has score 7) as shown in the Table I.

Product decomposition is an important stage in conjoint analysis. Many papers emphasize that the decomposition should be chosen carefully to satisfy a property known as *preferential independence*. Preferential independence is extremely important because if each set of attributes is preferentially independent of its complement set, then the attribute utility can be represented by an additive or multiplicative decomposition ([26]). If the attributes are not independent, according to [14], [15], and [5], the utility is more difficult to be estimated. It is not the goal of this paper to analyze methodologies and techniques for product decomposition, therefore we assume an initial a set of attributes together with their possible values.

The last step of our ACA-CQQL learning process yields the following DNF formula[1]:

$$U = (A_1 \wedge \overline{A_2} \wedge A_3 \wedge \overline{A_4} \wedge A_5) \vee (\overline{A_1} \wedge A_2 \wedge A_3 \wedge \overline{A_4} \wedge \overline{A_5})$$

[1]for sake of simplicity, in this example, U covers the two most important minterms.

After logical transformations such as distributivity, double negation, transforming to implication and dual implication the following set of decision rules are produced:
$R_1 : \overline{A_2} \vee A_5 \to A_1$; $R_2 : A_2 \to \overline{A_1}$; $R_3 : A_1 \vee \overline{A_2} \to A_5$; $R_4 : A_2 \to \overline{A_5}$;
$R_5 : \overline{A_2} \vee A_5 \to A_1$; $R_6 : A_2 \to \overline{A_1} \wedge \overline{A_5}$; $R_7 : A_5 \to A_1$; $R_8 : \overline{A_2} \to A_1 \wedge A_5$;
$R_9 : A_3$, $R_{10} : \overline{A_4}$
These rules are interpreted according with the user's level of importance of attribute values, e.g., R_9 and R_{10} requests that your product should definitely have a high level of A_3 (i.e. "large screen") and may have a low level of A_4 ("2 hour"). R_1 means "whenever you have a low level of A_2 and a high level of A_5, you should have a high level of A_1 too" that is *"if the phone has no WIFI and a low price then it should have OS Android"*.
In addition, each minterm of the the learned formula U defines a logical interpretation satisfying U. Each such interpretation recommends product profiles. For example, $\mathcal{I} = \{\overline{A_1}, A_2, A_3, \overline{A_4}, \overline{A_5}\}$ (introduced by the second minterm in our example) recommends
$("proprietary\ OS", "WIFI", " > 4.0''", "2\ hours", "high\ price")$
While these rules are suitable to be processed by rule engines, we can also derive logically equivalent representations tailored to human interpretation:
$F_1 : A_3 \wedge \overline{A_4}$;
$F_2 : A_1 \oplus \overline{A_2}$;
$F_3 : A_1 \leftrightarrow A_5$;
$F_4 : \overline{A_2} \oplus A_5$;

II. OVERVIEW OF OUR APPROACH

Let $\mathcal{A} = \{A_1, \ldots, A_n\}$ be a set of mutually independent attributes. Let $dom(A_i)$ the value space of attribute A_i. Let $\mathcal{O} = \{(a_1, ..., a_n) | a_k \in dom(A_k), k = 1, ..., n\}$ a set of possible objects built over $\mathcal{A}$. Let $\mathcal{S}$ be a finite set of *incomplete objects* or *stimuli* i.e. $s \in \mathcal{S}$ is a database tuple with nulls. The *null* interpretation related to these objects is in the sense of missing information, e.g., considering 6 attributes a

possible stimuli is the tuple $s = (a_1, null, null, a_4, null, null)$ where a_i are specific attribute values. $\mathcal{S}$ is built by extracting incomplete objects from $\mathcal{O}$. Let $\mathcal{P}$ be a set of respondents.
The *ACA-CQQL conjoint representation* is the weighted full DNF[2], $U = \bigvee_{w_k} m_k$ where m_k denotes the k-minterm and $w_k \in [0,1]$ is the weight of minterm m_k. Each minterm is a conjunction of exactly n literals (positive attribute occurrence or negative attribute occurrence) each corresponding to one of the n attributes of ACA-CQQL problem.

The overall approach we use is briefly described below:

1) For each respondent $p \in \mathcal{P}$, use an interview to derive a preference relation
$P_{\mathcal{S},p} : \mathcal{S} \times \mathcal{S} \longrightarrow \{0, 1, unknown\}$;
2) Use a stimuli preference relation $P_{\mathcal{S},p}$ to compute a rank $\rho_{\mathcal{S},p}$ on $\mathcal{S}$. A stimuli s is better ranked by $\rho_{\mathcal{S},p}$ when the user likes s better than other one (with a lower rank);
3) Derive an induced rank on $\mathcal{O}$, $\rho_{\mathcal{O},p}$; This step computes a rank on full objects by considering the computed rank in Step 2.
4) Use CQQL evaluation to obtain $P_{\mathcal{M},p}$ a preference relation on minterms of U; This step allows to express a respondent preference relation on minterms and by consequence to compute an overall preference as described in Step 5.
5) Compute an overall minterm preference $P_{\mathcal{M}}$ by aggregation of all $P_{\mathcal{M},p}$;
6) Use $P_{\mathcal{M}}$ to compute, $\rho_{\mathcal{M}}$, a rank on CQQL formula minterms; This is one of the core steps of our solution. Ranking minterms of the full DNF allows to select significant minterms and as such to derive an approximation of interests.
7) Create and interpret decision rules considering the best ranked minterms.

Methods for learning and predicting preferences are addressed by the machine learning community [21] and recommender systems [1]. These communities provides solutions such as approximating the scoring function by using interviews (*preference elicitation*) to *collaborative filtering*, where the user preferences are estimated from the preferences of other users.
The steps 1–3 are described in Section III while steps 4–7 are described in Section V.

III. Ordering from Preferences on Incomplete Information

Conjoint Analysis associates each respondent $p \in \mathcal{P}$, with a set of *pairwise comparisons* (2–stimuli questions), each comparison being rated on a Likert scale [28]. Each question may have its own scale. Likert scales are bipolar scaling methods therefore we can straightforward derive a preference from ratings.
Let $p \in P$ be a respondent. Let (s_i, s_j) be a comparison rated with r on a Likert scale $1..K_p$, $K_p \in \mathbb{N}$. One traditional scale is 1..9 but other choices can be used too (e.g. 1..5). On scale 1..9, $r = 1$ means "I like very much s_i" , $r = 5$ is neutral ("I equally like both s_i and s_j") while $r = 9$ means "I like very much s_j". Then

$$P(s_i, s_j) = \begin{cases} 0, r < \lfloor K_p/2 \rfloor + 1, \ (s_i \succ s_j) \\ unknown, r = \lfloor K_p/2 \rfloor + 1 \\ 1, r < \lfloor K_p/2 \rfloor + 1, \ (s_j \succ s_i) \end{cases}$$

The first step is to use the given initial preference $P_{\mathcal{S}}$ to learn an ordering function $\rho_{\mathcal{S}}$. We use the Algorithm 1, a greedy ordering introduced by [6].
This algorithm is based on viewing preferences as a directed weighted graph where the initial set of vertices V is equal of set of stimuli $\mathcal{S}$ and each edge $u \to v$ has weight $P_{\mathcal{S}}(u, v) = 1$ (v is preferred to u). Each vertex v gets a "potential" $\pi(v)$ which is the sum of the incoming edges minus the sum of outgoing edges. The Algorithm 1 picks some node t that has a maximum potential, assigns it a rank $\rho_{\mathcal{S}}(t) = |V|$ and then ordering in the same way the remaining nodes, after updating the nodes potentials.
The potentials define a stratification of the directed weighted graph into node sets of the same potential. The stratum

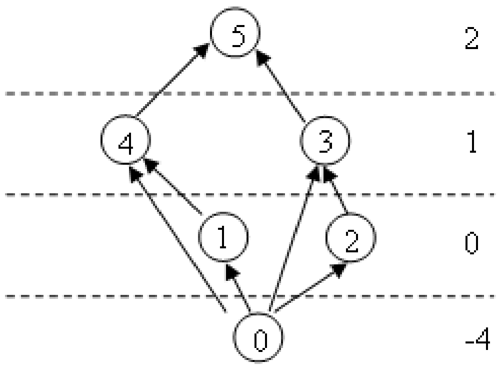

Fig. 1. Graph of preferences

```
Data: $\mathcal{S}$, binary preference $P_{\mathcal{S}}$
Result: $\rho_{\mathcal{S}}$
Let $V = \mathcal{S}$;
foreach $v \in V$ do
    $\pi(v) = \sum_{u \in V} P_{\mathcal{S}}(v, u) - \sum_{u \in V} P_{\mathcal{S}}(u, v)$;
end
while $V \neq \emptyset$ do
    Let $t = argmax_{u \in V} \pi(u)$;
    Let $\rho_{\mathcal{S}}(t) = |V|$;
    $V = V - \{t\}$;
    foreach $v \in V$ do
        $\pi(v) = \pi(v) + P_{\mathcal{S}}(t, v) - P_{\mathcal{S}}(v, t)$;
    end
end
```
Algorithm 1: $orderByPrefs(\mathcal{S}, P)$, [6]

[2]Any logical formula can be converted to a disjunctive normal form (DNF) by using logical equivalences, such as the double complement elimination, De Morgan's laws, and the distributive law. Recall that CQQL provides rewriting rules to translate from weighted operations to into CQQL Boolean calculus. Therefore we are going to learn a formula directly in the DNF form.

with the highest potential is processed first. However when processing the nodes in the same stratum all of them have the same potential, therefore the algorithm must choose between them. For example, considering the preferences as in Figure 1 (the initial potential of each stratum is shown too) it is easy to see that node 5 will be processed first (it gets a rank of 4) but then we have a nondeterministic selection because either 4 or 3 can be a choice (both will have the same updated potential). Therefore the output ordering provided by Algorithm 1 depends on the $argmax$ implementation[3]. Among others, orderings $\{0,1,2,3,4,5\}$ and $\{0,2,1,4,3,5\}$ can be produced.

Definition 1 (Incomplete Objects Equality and Membership): Two incomplete objects $s_1, s_2 \in \mathcal{S}$ are *equal*, and we write $s_1 = s_2$, when all their correspondent values are the same i.e. if $s_1 = (a_1, ..., a_n)$ and $s_2 = (b_1, ..., b_n)$ then $s_1 = s_2$ iff $a_k = b_k$ for all $k = 1, \dots, n$. The values equality interpretation is defined as in the below table:

$=$	v	$null$
v	true	false
$null$	false	true

Let $s = (a'_1, ..., a'_n) \in \mathcal{S}$ and $o = (a_1, ..., a_n) \in \mathcal{O}$. We say that s *is contained by* o (or o *contains* s) and we denote $s \subseteq o$ iff $a'_k \neq null$ implies $a'_k = a_k$ for all $k = 1, \dots, n$. Therefore an incomplete object is contained by an object when its non-null values are the same in the related object.

The following definition introduces an extension from an order on incomplete objects to an order on complete objects.

Definition 2 (Ordering Extension): Let $\mathcal{O}$ be a set of complete objects, $\mathcal{S}$ be a set of incomplete objects and $P_\mathcal{S}$ a binary preference. Let $\rho_\mathcal{S}$ obtained by Algorithm 1. Then, the ordering $\rho_\mathcal{O}$ defines the *extension of* $\rho_\mathcal{S}$ *from* $\mathcal{S}$ *to* $\mathcal{O}$:

$$\rho_\mathcal{O}(o) = \sum_{s_i \in \mathcal{S},\ s_i \subseteq o} u_{s_i} \rho_\mathcal{S}(s_i)$$

where u_{s_i} is the stimuli utility. If no stimuli belong to object o then $\rho_\mathcal{O}(o) = 0$.

Stimuli utility is an additive linear combination of attribute value weight i.e. if m_s is the number of distinct attribute values of $s \in \mathcal{S}$, then $u_s = \frac{1}{m_s} \sum_{i=1}^{N} \theta_i \cdot (\sum_{k=1}^{n_i} \theta_{ik} \cdot x_{ik})$ where $x_{ik} = 1$ when A_i is present (with value a_{ik}) in stimuli s, otherwise is 0 (null values are ignored). $\theta_i \in [0,1]$ is the weight of attribute A_i and $\theta_{ik} \in [0,1]$ is the weight of value $a_{ik} \in dom(A_i)$. The reader may notice that θ_i and θ_{ik} are obtained from the respondent initial ratings of attributes and attribute values as requested by traditional ACA typically by using a $[0,1]$ mapping from ratings on Likert scales.

Proposition 1: Let $o_1, o_2 \in \mathcal{O}$ such that $s_1, ..., s_k \subseteq o_1$ and $s_1, ..., s_k, s_{k+1} \subseteq o_2$. Then $\rho_\mathcal{O}(o_2) > \rho_\mathcal{O}(o_1)$.

Proof: Easy to see that $\rho_\mathcal{O}(o_2) = \rho_\mathcal{O}(o_1) + u_{s_{k+1}} \rho_\mathcal{S}(s_{k+1})$. ∎

[3]While Algorithm 1 successfully applies when we have a binary preference as input (see [7] for an optimality proof), in practical cases we work with non binary preferences i.e., there are pairwise comparisons (x, y) which are rated neutral, therefore $P_\mathcal{S}(x, y) = unknown$. In this work we will ignore neutral rated questions.

Any ordering induce a preference as below:

$$P_\rho(x,y) = \begin{cases} 1, \ \rho(x) < \rho(y) \\ 0, \ \rho(x) > \rho(y) \\ unknown, \ \rho(x) = \rho(y) \end{cases}$$

When ρ is strict then, P_ρ is binary (the $unknown$ case does not occur).

In the rest of the paper we assume as given $\rho_\mathcal{O}$ and $P_\mathcal{O}$. The next step is to use CQQL minterm evaluation to obtain a preference on formula minterms. The goal of the next section is to give basic information about CQQL language, particularly on CQQL evaluation rules.

IV. Commuting Quantum Query Language (CQQL)

Introduced in [35], CQQL is an extension of the relational calculus using quantum logic paradigm which defines *metric*(or *similarity*) predicates, weighted conjunction ($\wedge_{\theta_1,\theta_2}$), weighted disjunction ($\vee_{\theta_1,\theta_2}$) and quantum negation. CQQL extends relational calculus by allowing for complex logical formulas mixing classical first-order logic predicates with metric predicates. Score values from $[0,1]$ results from the evaluation of metric predicates on data objects. On the other hand, traditional database predicates (*non-metric* predicates) provide 1 for true and 0 for false. A significant advantage of the language derives from the capabilities of quantum measurement results to be interpreted as probability values. Therefore conjunction, disjunction and negation conforms with the probability calculus. As a consequence, many concepts of information retrieval, already embedded into linear algebra and probability theory, can be addressed.

Processing *non-metric attributes* i.e. attributes not requiring any degree of neighborhood between their values, therefore we follow the classical database query processing. When processing *metric attributes* i.e. data for which we are interested in distinguishing comparisons between two values which are close neighbors from those which lie far away from each other we follow the CQQL approach of similarity evaluation based on quantum measurement. An example of a *non-metric attribute* (or *database attribute*) is *"Operating System"* as shown in Section I-A.

Attributes such as *"Price"* comes naturally as *metric attributes* (or similarity attributes) because the users are definitely interested in low prices, therefore they are evaluated by means of similarity predicates: it becomes easy to derive an order of the values of these attributes and to use a continuous, and monotonic predicate $p_{Price} : dom(Price) \longrightarrow [0,1]$ to associate their values with truth degrees.

Definition 3 (Database Values Evaluation): Let o be a database object. If A is a non-metric attribute such that $o.A = a$, then

$$eval(A = v, o) = \begin{cases} 1, \ if \ v = a \\ 0, \ otherwise \end{cases}$$

whenever $a = null$ or $v = null$ we take
$eval(A = v, o) = unknown$.

Let A be a metric attribute. Any value $a \in dom(A)$, is mapped by quantum encoding into a vector state $\overline{a}$. This give the user a means to assign its own semantics to the resulting proximity values. The CQQL framework allows evaluation of similarity values by using any similarity measure s satisfying the following conditions:

1) for all $a, b \in dom(A)$, $s(\overline{a}, \overline{b}) \in [0,1]$.
2) for all $a, b \in dom(A)$, $s(\overline{a}, \overline{b}) = 1 \Leftrightarrow \overline{a} = \overline{b}$.
3) for all $a, b \in dom(A)$, $s(\overline{a}, \overline{b}) = s(\overline{b}, \overline{a})$.

A widely used such similarity measure is the cosine similarity $s(\overline{a}, \overline{b}) = \frac{\overline{a} \cdot \overline{b}}{\|\overline{a}\| \|\overline{b}\|}$.

Definition 4 (Retrieval Values Evaluation):

$$eval(A \approx v, o) = \begin{cases} s(\overline{v}, \overline{o.a}), \; if v \neq null \; and \; a \neq null \\ unknown, \; otherwise \end{cases} \quad (1)$$

Before a CQQL formula can be evaluated it has to be normalized. The normalization requires a special syntactical form starting with a formula that is in the prenex and disjunctive normal form. Then all common atoms φ are removed by applying the rule $(\varphi \wedge \varphi_1) \vee (\varphi \wedge \varphi_2) = \varphi \wedge (\varphi_1 \vee \varphi_2)$ (derived from distributivity and absorbtion).The normalization algorithm is based on Boolean transformation rules and is described in [35].

Definition 5 (Formula Evaluation): Let $\varphi_1 \wedge \varphi_2$, $\varphi_1 \vee \varphi_2$ and φ normalized formulas. Then

$eval(\varphi_1 \wedge \varphi_2, o) = eval(\varphi_1, o) * eval(\varphi_2, o)$
$eval(\varphi_1 \vee \varphi_2, o) = eval(\varphi_1, o) + eval(\varphi_2, o) -$
$eval(\varphi_1, o) * eval(\varphi_2, o)$
$eval(\neg\varphi, o) = 1 - eval(\varphi, o)$

Applying CQQL evaluation against a set of database objects we get a rank of all objects according to their score value.
Integrating weights into CQQL can be achieved simply. The core idea is the direct transformation of a weighted conjunction or disjunction into a logical expression in which weight values are converted into 0-ary predicates using the rewriting rules below:

$\varphi_1 \wedge_{\theta_1,\theta_2} \varphi_2 \Rightarrow (\varphi_1 \vee \neg\theta_1) \wedge (\varphi_2 \vee \neg\theta_2)$
$\varphi_1 \vee_{\theta_1,\theta_2} \varphi_2 \Rightarrow (\varphi_1 \wedge \theta_1) \vee (\varphi_2 \wedge \theta_2)$

These rules provide a way to transform weighted conjunction, weighted disjunctions into CQQL Boolean calculus. More technical details on these transformations and examples can be found in [37]. An intuitive interpretation of rewriting rules when considering only Boolean weights (0 and 1)is described in the below table.

θ_1	θ_2	$\varphi_1 \wedge_{\theta_1,\theta_2} \varphi_2$	$\varphi_1 \vee_{\theta_1,\theta_2} \varphi_2$	Explanation
0	0	1	0	none of φ_1 and φ_2 counts
0	1	φ_2	φ_2	only φ_2 counts
1	0	φ_1	φ_1	only φ_1 counts
1	1	$\varphi_1 \wedge \varphi_2$	$\varphi_1 \vee \varphi_2$	both φ_1 and φ_2 counts

Existential quantification and universal quantification in a CQQL query are evaluated by computing maximum respectively minimum of the weight values of the appropriate objects.

V. Conjoint Analysis with CQQL

This scenario is centered on a weight learning algorithm, an adaptation of the weighted majority algorithm proposed in [29] and discussed in [6] and [7]. The main assumption, introduced by [29] and conforming with conjoint analysis principles ([22], [23]) is the compositional approach to overall preference as a weighted sum of individual preferences.

Let $\mathcal{A} = \{A_1, \ldots, A_n\}$ be a set of mutually independent attributes. Let $dom(A_i)$ the value space of attribute A_i. Let $\mathcal{O} = \{(a_1, ..., a_n) | a_k \in dom(A_k), k = 1, ..., n\}$ a set of possible objects built over $\mathcal{A}$. Let $\mathcal{S}$ be a set of stimuli and $\mathcal{M}$ be the set of all minterms over $\mathcal{A}$. The overall idea of the conjoint process is to obtain an aggregated preference on possible minterms and use this preference to compute an ordering on $\mathcal{M}$. Given minterm preferences $P_{\mathcal{M},p}$ for all respondents $p \in \mathcal{P}$ the algorithm learn optimal $\{w_p \in [0,1] | p \in \mathcal{P}\}$ such that

$$P_{\mathcal{M}}(m_1, m_2) = \sum_{p \in \mathcal{P}} w_p \cdot P_{\mathcal{M},p}(m_1, m_2), \; m_1, m_2 \in \mathcal{M}$$

such that a specific loss function is minimized. This solution uses a loss function defined as the [0,1] normalization of the number of discordant preferences between $\widehat{P}_{\mathcal{M}}$ – a preference function computed from experts evaluations, and $P_{\mathcal{M}}$ – the preference computed by the learner:

$$Loss(P_{\mathcal{M}}, \widehat{P}_{\mathcal{M}}) = \frac{\sum_{\widehat{P}_{\mathcal{M}}(x,y)=1} (1 - P_{\mathcal{M}}(x,y))}{|\widehat{P}_{\mathcal{M}}|}$$

As largely discussed in [6] and [7], this loss function has a probabilistic interpretation: if $P_{\mathcal{M}}(x,y)$ is interpreted as the probability that y is preferred to x then $Loss(P_{\mathcal{M}}, \widehat{P}_{\mathcal{M}})$ is the probability of disagreement of $P_{\mathcal{M}}$ with the feedback on (x,y) from $\widehat{P}_{\mathcal{M}}$.
As we use this algorithm on minterm preferences, at first, we need a mechanism to compute minterm preferences from object preferences.
Recall that the goal is to learn a formula as a disjunction of most important minterms. It defines a logical representation of the respondent interests and offers an explanation on user's interests. To compute it we compute a ranking of formula minterms from preferences on database objects and take the most dominant minterms.

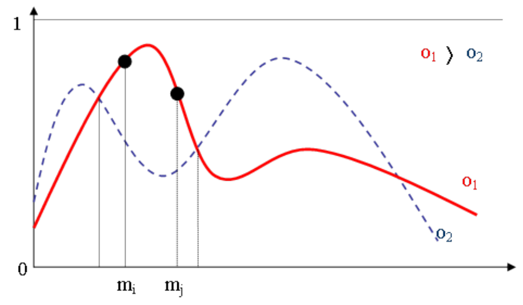

Fig. 2. Geometric Interpretation of Rule 1 on minterm preferences

A. Computing Minterm Preferences

Definition 6 (Minterm Preference): Let $P_{\mathcal{O}}$ be a preference. Let $\mathcal{M}$ be the set of minterms of U. Let $o_1, o_2 \in \mathcal{O}$ and $m_i, m_j \in \mathcal{M}$. The minterms preference $P_{\mathcal{M}}$ is computed according with the following rules:

$$\left.\begin{array}{l} P_{\mathcal{O}}(o_1,o_2)=0 \\ eval(m_i,o_1) > eval(m_j,o_2) \\ eval(m_i,o_1) > eval(m_j,o_1) \end{array}\right\} \Rightarrow P_{\mathcal{M}}(m_i,m_j)=0$$
$$\left.\begin{array}{l} P_{\mathcal{O}}(o_1,o_2)=0 \\ eval(m_j,o_1) > eval(m_i,o_2) \\ eval(m_j,o_1) > eval(m_i,o_1) \end{array}\right\} \Rightarrow P_{\mathcal{M}}(m_i,m_j)=1$$

All remaining unassigned preferences are considered $unknown$.
Basically, *"minterms that are more similar in CQQL evaluation against preferred objects are preferred"*. This heuristic has a geometric interpretation. Figure 2 shows minterms m_i and m_j such that $P_{\mathcal{M}}(m_i,m_j)=0$. We consider only the regions were the CQQL minterm evaluation of the preferred object (o_1 in Figure 2) is better that minterm evaluation on the less preferred object (o_2) and inside these regions we compute preferences induced by the the ordering produced by CQQL minterm evaluation.

Proposition 2: The minterm preference and object preference are monotonic with respect of CQQL evaluation i.e.,
$P_{\mathcal{O}}(o_1,o_2)=P_{\mathcal{M}}(m_i,m_j)=0$ iff
$eval(m_i,o_1) > eval(m_j,o_2)$
and
$P_{\mathcal{O}}(o_1,o_2)=P_{\mathcal{M}}(m_i,m_j)=1$ iff
$eval(m_j,o_2) > eval(m_i,o_1)$.

Proof: The first relation yields directly using the first rule from Definition 6. The second comes from the preference symmetry ($P(x,y)=0 \Leftrightarrow P(y,x)=1$) and second rule. ■

B. Performing Conjoint Analysis

The conjoint process takes place in a sequence of rounds, one for each interview question and for all respondents. The complete description is depicted in Algorithm 2. Basically, on the q-th iteration we have $\mathcal{M}$ the set of minterms and $P^{(q)}_{\mathcal{M},p}$ the (computed) respondents preferences on $\mathcal{M}$. Then, the learner receives feedback from the environment. We assume that this feedback comes as a preference function on minterms, previously computed using the same techniques from experts

```
Data: $\mathcal{O}$, $\mathcal{S}$, $\mathcal{Q}$ interview
Data: $r \in [0,1]$. Initial $w^{(1)} \in [0,1]^{|\mathcal{P}|}$ with
      $\sum_{j=1}^{|\mathcal{P}|} w_j^{(1)} = 1$
Result: $U$, $R_1, ..., R_l$
foreach $q \in \mathcal{Q}$ do
    Update preferences $\{P^{(q)}_{\mathcal{S},p} | p \in \mathcal{P}\}$;
    Compute minterm preferences $\{P^{(q)}_{\mathcal{M},p} | p \in \mathcal{P}\}$;
    Aggregate
    $P_{\mathcal{M}}(m_i,m_j) = \sum_{p\in\mathcal{P}} w_p^{(q)} \cdot P^{(q)}_{\mathcal{M},p}(m_i,m_j)$;
    $L_{(q)} = Loss(P^{(q)}_{\mathcal{M}}, \widehat{P}_{\mathcal{M}})$;
    Update $w_p^{(next(q))} = \frac{1}{C^{(q)}} w_p^{(q)} \cdot r^{L_{(q)}}$, for all $p \in \mathcal{P}$;
end
Compute $k\_means\_Lloyd(P_{\mathcal{M}}$, $[0,1]$,
$\{0, 1, unknown\})$;
Compute ordering $\rho_{\mathcal{M}} = orderByPrefs(\mathcal{M}, P_{\mathcal{M}})$;
Compute $U$ as a disjunction of the first k minterms
ordered by $\rho_{\mathcal{M}}$;
Compute decision rules $R_1, ..., R_l$;
```
Algorithm 2: CA inside a Logical Framework

inputs. At each step we update the new weight vector as $w_p^{(next(q))} = \frac{1}{C^{(q)}} w_p^{(q)} \cdot r^{Loss(P^{(q)}_{\mathcal{M}}, \widehat{P}_{\mathcal{M}})}$ for all $p \in \mathcal{P}$ where $r \in [0,1]$ is a calibration constant and $C^{(q)}$ is an normalization constant chosen so that $\sum_{p\in\mathcal{P}} w_p^{(next(q))} = 1$.
The "majority weighted" solution for preference aggregation is largely used by machine learning community. It obtains a partial preference $P_{\mathcal{M}} : \mathcal{M} \times \mathcal{M} \longrightarrow [0,1]$. However, Algorithm 1 works with binary preferences, therefore we have to perform a clustering from $P_{\mathcal{M}}(m_1,m_2) \in [0,1]$ to $P_{\mathcal{M}}(m_1,m_2) \in \{0,1,unknown\}$. There is large literature on clustering methods ([34] is a recent survey). By now, because our clustering space is Euclidean and one dimensional we considered standard k-means clustering procedure (Lloyd's algorithm) to produce three clusters: *"0"*, *"0.5"("unknown")* and *"1"*.

C. Creating and Interpreting Decision Rules

Following the above learning process we get a formula U as a disjunction of the most K dominant minterms, $m_{i_1}, \ldots, m_{i_K}$. To obtain a ruleset based on U:

1) *Compute a conjunctive normal form.* Applying the distributivity laws to U, then eliminate duplicates ($L \vee L \Rightarrow L$) and tautology (delete $L \vee \overline{L}$). As a result of this step we obtain U in the CNF form.
2) *Find unit clauses.* (i.e. the clauses containing only one literal). All these unit clauses (or facts) are mandatory rules of our ruleset.
3) *Simplify unit clauses.* All unit clauses must be true therefore we replace the corresponding literals from U, then apply the usual Boolean computation. As a result all clauses of U do not contain any literals from unit clauses. Let $\mathcal{C}$ be the set of all clauses of U.
4) *Transform each clause to a rule.* For each $C \in \mathcal{C}$, let $C = L_1 \vee \ldots \vee L_j$ and let L_1 be the desired conclusion of

the rule[4]. Then the rule corresponding to C is obtained by simple transformation to implication i.e. $\overline{\left(\overline{(L_2 \vee \ldots \vee L_j)}\right)} \vee L_1 \Rightarrow \overline{\left(\overline{L_2} \wedge \ldots \wedge \overline{L_j}\right)} \vee L_1 \Rightarrow R_C : \overline{L_2} \wedge \ldots \wedge \overline{L_j} \rightarrow L_1$.
Eliminate possible double negation of the rule literals (apply $\overline{\overline{L}} \Rightarrow L$).

VI. Experimental Results

Our experiments were related to an use case considering 15 attributes. We considered one scale (1..9) for all attribute ratings and the experiment uses a slightly modified variant of logistic function $f(n) = \frac{1}{(1+e^{-n+5})}$ for rating mapping. This maps the scale 1..9 into 0.0179, 0.0474, 0.1192, 0.2689, 0.5, 0.7310, 0.8808, 0.9525, 0.9820.
The training data had 16 objects previously ranked by experts, from which we derived 32 stimuli. Each interview had 15 pairwise comparisons containing stimuli of similar utility, i.e., a pair comparison (s_1, s_2) was generated iff $|u_{s_1} - u_{s_2}| < \varepsilon_p$ where ε_p is a threshold depending on the respondent.All questions used the same evaluation scale, 1..9. We computed a CQQL formula based on 1,2,3,5, and 10 most dominant minterms. The respondents scores and preferences were set up at random and we performed 100 simulations.
The result quality was measured by using Spearman's correlation, to compare the training data rank obtained by CQQL evaluation with the experts evaluation as shown in the below table:

No. of minterms	Best Spearman	Worse Spearman	Average
1	0.4814	0.1231	0.4002
2	0.7871	0.3612	0.72
3	0.91	0.7713	0.8724
5	0.9159	0.7023	0.8813
10	0.9921	0.7718	0.8763

The actual results show that using the first 3 or 5 dominant minterms give a correlation comparable with the case when the first 10 most dominant minterms were used. When creating decision rules the main complexity parameter is the number of minterms to be used: using the first two dominant minterms as a starting base will produce the simplest rules, as the ones described in Section I-A[5]. Such rules can be easily interpreted both by a human expert or a rule engine. However, when four or more dominant minterms were used then a much larger ruleset is computed and the complexity of each rule increase too. Typically such a ruleset will be processed by means of a rule engine although if a human expert is interested only in partial decisions he can use only a subset of these rules ([12]). The settings of the decision rules creation process considered the first two dominant minterms of the obtained full DNF formula and the first three user high rated attributes as rule conclusions.

[4]This choice is related to the human expert interests on one or other attribute. Other solutions may consider the user's high rated attributes as a choice.

[5]Notice that the rules produced as described in Section V-C can also be grouped by head towards a much compact writing. This is easy when two or three minterms were used in the CQQL formula, but becomes not so much useful when more minterms were considered.

VII. Conclusions and Future Research

This article has shown how conjoint analysis can be modeled using the tools offered by CQQL, a logic-based similarity query language. The advantage of the approach lies in the expressive power and flexibility of logic to encode the conjoint model. We obtained an ACA-CQQL interpretation as ranking of potential products according with some preexistent pattern (CQQL query) and used some learning algorithms to compute the solution. Doing conjoint analysis inside a logic-based framework creates opportunities to apply such data analysis strategy to other kind of problems such as delivering recommendations inside social networks, and deriving user's profiles from mining social activities.

We plan to extend our approach. The model formulation offers the opportunity to tune the conjoint problem by performing a choice on a number of parameter as discussing below.

Attribute classification and preferences normalization. CQQL supports both metric and non-metric attributes therefore both crisp and non-crisp data can be handled. In addition, each metric attribute may come with his own rating scale. The length of the scale might have to be considered while creates much more granularity of weights. The actual experiment uses classical exponential utility but other normalization approach should be considered in the future research.

Stimuli and pairwise comparisons. While in our experiments we've extracted stimuli on a heuristic base, possibly a more systematic approach (using techniques already provided in economics research) may yield to better results. Stimuli with two conjuncts are very easy to be understood by respondents, but using 3 or more conjuncts in the stimuli will improve the quality of the extended ordering on the training objects (see Definition 2). Secondly, CA also considers questions with more than 2 stimuli choices: in this case, we cannot use bipolar scales, therefore the preference order induced by question rating is not unique[6]. Finally, in the present work we ignore neutral rated comparisons but we aim to investigate other approaches in further research.

The interview creation. This work does not impose any restrictions on the interview creation, therefore the list of questions composing the respondent interview may support various orderings such as *"by question importance"* or *"by question difficulty"*. In all cases a strategy should not violate the transitivity property of preferences: non-transitive pairs creates inconsistent problems (not all preferences can be simultaneously satisfied). An interview creation strategy

[6]Receiving a question $q = (s_1, s_2, s_3)$ after scoring $r_1 < r_2 < r_3$ we get $s_1 \preceq s_2 \preceq s_3$ but we have to understand how to derive ratings for pairs such as $s_1 \preceq s_2$.

should detect this issue in the interview creation stage (immediately after the user rated a new pair comparison) and not during the learning stage. In addition, because during an interview, traditional ACA may also ask respondents to rate individual products(complete objects) on a purchase likelihood scale, we aim to generate these by considering intermediary learned formulas for tuning the preferences set towards fasten the convergence to an acceptable solution.

The learning strategy. Actually, the algorithm uses a loss function previously tested by other research. However we intend to investigate the performance of other loss functions such as

$$Loss(P, \widehat{P}) = \frac{\sum_{(x,y)\in X\times X}(\widehat{P}(x,y) - P(x,y))}{|X \times X|}$$

which is similar with Kendall's τ rank correlation coefficient. The learning strategy implementation supports parallelization by distributing processing on persons or person groups, therefore an ACA-CQQL application should be able to offer fast answers. We also intend to investigate other learning algorithm such as ones described by [21]. Obtaining various ACA-CQQL conjoint representation that can be interpreted and explained is one of the powerful achievements of the proposed solution. We get explanations both in terms of user's most preferred attributes/values, and preferred products/services. In future research alternative solutions to derive minterm preferences have to be examined.

Decision rules. The obtained rule set can be subject of interpretation in various ways from simpler (as seen in Example 1) to solutions involving different semantics such as incomplete/imprecise information, [36], probabilistic models [32], or plausibility-based models [10], [11]). We plan to address some of these alternatives in subsequent research.

REFERENCES

[1] G. Adomavicius, and A. Tuzhilin, Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions, IEEE Transactions on Knowledge and Data Engineering, Vol. 17, No. 6, June 2005, pp. 734-749.
[2] N.H. Anderson. Foundations of information integration theory. Acad Press, 1981.
[3] D. Baier and M. Brusch (Eds.) Conjointanalyse, Methoden - Anwendungen - Praxisbeispiele, Springer, Berlin, 2009.
[4] D. Baier. Conjoint Measurement in der Innovationsmarktforschung, in: Baaken, Thomas; Höft, Uwe; Kesting, Tobias (Hrsg.), Marketing für Innovationen - Wie innovative Unternehmen die Bedürfnisse ihrer Kunden erfüllen, Harland Media, Münster, ISBN-13 978-3-938363-42-3.
[5] F. J. Carmone and P. E. Green. Model Misspecification in Multiattribute Parameter Estimation. Journal of Marketing Research, 18, 1981, pp.87-93.
[6] W. Cohen, R.E. Schapire and Y. Singer. Learning to Order Things. Advances in Neural Information Processing Systems 10, Morgan Kaufmann, 1998.
[7] W. Cohen, R.E. Schapire and Y. Singer. Learning to Order Things. Journal of Artificial Intelligence Research 10 (1999), pp. 213-270.
[8] P.C. Fishburn. Methods of Estimating Additive Utilities. Management Science, Vol. 13, 1967, pp.435-453.
[9] P.C. Fishburn. Utility Theory. Management Science, Vol. 14, 1968, pp. 335-367.
[10] N. Friedman, and J.Y. Halpern. Plausibility measures and default reasoning. Journal of the ACM, 48:648-685, 2001.
[11] A. Giurca. A Logic with Plausibility. Annales of Craiova University, Mathematics and Computer Science Series, XXVII, pp.105-115, 2000.
[12] A. Giurca, D. Gasevic, and K. Taveter Handbook of Research on Emerging Rule-Based Languages and Technologies: Open Solutions and Approaches, Vol 1-2, IGI Publishers, 2009, ISBN13: 978-1-60566-402-6.
[13] P. E. Green. Hybrid Models for Conjoint Analysis: An Expository Review. Journal of Marketing Research, Vol. 21, 1984, pp. 155-169.
[14] P. E. Green and M. T. Devita. A Complementary Model of Consumer Utility for Item Collections. Journal of Consumer Research, 1, (December), 1974, pp.56-67.
[15] P. E. Green and M. T. Devita. An Interaction Model of Consumer Utility. Journal of Consumer Research, 2, (September), 1975, pp.146-153.
[16] P. E. Green, S. M. Goldberg, and M. Montemayor. A Hybrid Utility Estimation Model for Conjoint Analysis. Journal of Marketing, Vol. 45, Winter 1981, pp.33-41.
[17] P. E. Green and A. M. Krieger. Conjoint Analysis with Product-Positioning Applications. J. Eliashberg, and G.L. Lilien (Eds.): Handbook in Operations Research and Management Science, Vol. 5, North-Holland, 1993, pp. 467-515.
[18] P. E. Green and V. Rao. Conjoint measurement for quantifying judgmental data. Journal of Marketing Research, 8, 1971, pp.355-363.
[19] P. E. Green and V. Srinivasan. Conjoint Analysis in Consumer Research: Issues and Outlook. Journal of Consumer Research, Vol. 5, September 1978, pp. 103-123.
[20] R.T. Hoepfl and G. P. Huber. A Study of Self-Explicated Utility Models. Behavioral Science, Vol. 15, 1970, pp. 408-414.
[21] E. Hüllermeier, J. Fürnkranz, W. Cheng, and K. Brinker. Label ranking by learning pairwise preferences, Artificial Intelligence, Vol. 172:16-17, Nov. 2008, pp. 1897-1916.
[22] R. M. Johnson. Tradeoff Analysis of Consumer Values. J. of Marketing Research, 1974, pp. 121-127.
[23] R. M. Johnson. Accuracy of Utility Estimation in ACA. Working Paper, Sawtooth Software, Sequim, WA, April 1987.
[24] R. M. Johnson. Comment on Adaptive Conjoint Analysis: Some Caveats and Suggestions. Journal of Marketing Research, 28, 1991, pp. 223-225.
[25] R. M. Johnson. Comments on Studies Dealing With ACA Validity and Accuracy, With Suggestions for Future Research, 1991 published by Sawtooth Software.
[26] R.L. Keeney and H. Raiffa. Decisions with multiple objectives: Preferences and value tradeoffs. Wiley Series in Probability and Mathematical Statistics. NY: John Wiley & Sons, 1976.
[27] K. Lancaster. A new approach to consumer theory. J. of Political Economy, 74, 1966, pp.132-157.
[28] R. Likert. A Technique for the Measurement of Attitudes. Archives of Psychology 140, 1932, pp.1-55.
[29] N. Littlestone and M. Warmuth. The weighted majority algorithm. Information and Computation, 108 (2), 1994, pp. 212-261.
[30] R.D. Luce and J. W. Tukey. Simultaneous Conjoint Measurement: A New Type of Fundamental Measurement. J. of Mathematical Psychology, 1, 1964, pp.1.27.
[31] R.D. Luce and P. Suppes. Preference, utility and subjective probability. in Luce, R.D., Bush, R.R., and Galanter, E. (Eds.), Handbook of Mathematical Psychology, III, New York: Wiley, 1965, pp.235-406.
[32] N. J. Nilsson. Probabilistic logic. Artificial Intelligence 28(1):1986, pp.71-87.
[33] K.L. Norman and J.J. Louviere. Integration of attributes in public bus transportation: two modeling approaches. Journal of Applied Psychology, 59, 6, 1974, pp.753-758.
[34] L. Rokach. A survey of Clustering Algorithms. Data Mining and Knowledge Discovery Handbook, 2nd ed. Springer 2010, pp. 269-298, ISBN 978-0-387-09822-7.
[35] I. Schmitt. QQL: A DB&IR Query Language. VLDB J., 17(1):39-56, 2008.
[36] G. Wagner. Logic Programming with Strong Negation and Inexact Predicates. Journal of Logic and Computation 1(6): 835-859 (1991).
[37] D. Zellhöfer and I. Schmitt. A preference-based approach for interactive weight learning: learning weights within a logic-based query language. Distributed and Parallel Databases (2010) 27: 31-51, DOI 10.1007/s10619-009-7049-4.