Solving "Greeting a Customer with Unknown Data" Challenge with Epistemic DMN*

Đorđe Marković[1,2,*], Marc Denecker[1,2]

[1]KU Leuven, Celestijnenlaan 200a, Leuven, 3001, Belgium
[2]Leuven.AI - KU Leuven Institute for Artificial Intelligence

Abstract

Despite the variety of existing rule-based decision modeling languages, the "*Greeting a Customer with Unknown Data*" challenge (posted in 2016 on the Decision Management Community [1]) has only one candidate solution [2].

In this paper, we demonstrate that a correct rule-based modeling of the challenge requires epistemic rule conditions. Furthermore, we provide a solution to the challenge using the epistemic DMN modeling language, an extension of Decision Model and Notation with epistemic operators.

Keywords

Reasoning with Uncertainty, Decision Model and Notation, Epistemic logic, Undefinedness

1. Introduction

The "*Greeting a Customer with Unknown Data*" challenge was posted in 2016 as the part of Decision Management Community [1] challenges list. Eight years later there is still only one candidate solution [2] proposed by Feldman based on OpenRules [3]. The essence of this challenge is in representing and making decisions with missing data. For example, information that the time at the customer's location is either 10 or 11 AM is sufficient to make the decision to use the greeting "Good morning". However, state-of-the-art rule-based decision modeling languages provide restricted support for representing uncertainty about the input variables (e.g., time is either 10 or 11 AM), and decision rules with epistemic conditions (e.g., if Time at the customer's location is unknown, "Hello" is used as a greeting). The mismatch appears because conditions in decision rules are interpreted as purely objective (i.e., constraining the state of affairs). While intuitively humans often think of them epistemically, meaning that decision rules are constraining what is known about the state of affairs by some agent.

Inspired by these issues, in our previous work [4] we proposed an epistemic interpretation of rule-based decision modeling languages and demonstrated its suitability for handling missing data. Additionally, we developed a proof-of-concept *epistemic* DMN (eDMN) decision modeling language based on the state-of-the-art Decision Model and Notation (DMN) language and using Ordered Epistemic Logic (OEL) to provide model semantics of the language. Decision Model and Notation (DMN) [5] is a modeling language designed by the Object Management Group (OMG) that targets business decisions and focuses on human readability. For modeling decisions, DMN uses a tabular notation for rule-based decision modeling. Ordered Epistemic Logic (OEL)[6] is an extension of classical first-order logic that includes an epistemic operator K, where $K[\psi]$ means that ψ is known to be true. Unlike standard epistemic logic, OEL restricts the K operator to operate externally by referencing other theories. Specifically, it allows expressing that some statement ψ is known according to some theory T, written

RuleML+RR'24: Companion Proceedings of the 8th International Joint Conference on Rules and Reasoning, September 16–22, 2024, Bucharest, Romania

*This work was partially supported by the Flemish Government under the "Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen".

*Corresponding author.
dorde.markovic@kuleuven.be (Đ. Marković); marc.denecker@kuleuven.be (M. Denecker)
https://djordje.rs (Đ. Marković); https://people.cs.kuleuven.be/~marc.denecker/ (M. Denecker)
0000-0002-1932-975X (Đ. Marković); 0000-0002-0422-7339 (M. Denecker)

as $K[T][\psi]$. Additionally, the use of the K operator must be stratified, meaning there are no circular references of theories through the K operator, which is the reason for the term "ordered".

In this paper, our goal is to address the problem of uncertainty in the "*Greeting a Customer with Unknown Data*" challenge using eDMN. Rather than merely solving this specific challenge, we aim to demonstrate that the epistemic interpretation of rule-based decision modeling languages is essential for correctly addressing issues arising from any form of uncertainty.

The rest of the paper is structured as follows: Section 2 provides a detailed description of the "*Greeting a Customer with Unknown Data*" challenge. The analysis of this challenge is presented in Section 3. Section 4 specifies the criteria for a correct solution to the challenge and the criteria for a rule-based decision modeling language that supports reasoning with uncertainty. Section 5 introduces eDMN, and Section 6 explains the solution to the challenge using eDMN. The results demonstrating the correctness and completeness of the proposed solution are shown in Section 7, while Section 8 compares our solution with that of Feldman and eDMN with other tools. Finally, Sections 9 and 10 provide notes on the implementation of the eDMN system and the conclusion.

2. Challenge description

The challenge "*Greeting a Customer with Unknown Data*" first appeared on the DMN Challenge list in August 2016 [1]. The problem gravitates around situations where decisions should be made while the exact state of affairs is not fully known. More detailed, a fictional company needs a system for the automatic generation of personalized messages to their clients. This system has access to data of users that is available at the moment, e.g., first and last name, date of birth, phone number, address, etc. The generated messages should start with the greeting of the customer, e.g., "Good evening, Mr. Bond". The problem boils down to making decisions about the greeting based on the time zone of the customer, and salutation based on gender and marital status. The information about the gender and marital status of a customer is either available or not, while the time zone can be acquired in different ways with different precision. For example, a customer's address would provide the exact time zone, and information on the country might provide a range of time zones but still, a good guess can be made. Furthermore, country information can be extracted from the phone number, etc. Another issue with the greeting message is that the estimated time at the customer's location is not the only factor but also the fact of whether it is summer or winter time. Based on this information the thresholds for the different phases of the day change.

Since the focus of this paper is on decision modeling with uncertainty, we abstract away the problem of determining the possible time(s) at the customer's location because it does not contribute to the complexity of the model. Hence, we assume that the system is given an estimate of the *Current time* and *Summer/Winter time* information about the customer's location.

2.1. Specification of the problem

Greeting decision Following is the precise specification for making the *Greeting* decision:

G1 - Use **"Good Morning"** if it is known that at the customer's location:

a) it is *Summer time* and the *Current time* is between $[0..11)$, or

b) it is *Winter time* and the *Current time* is between $[0..12)$, or

c) *Summer/Winter time* is unknown and the *Current time* is between $[0..11)$

G2 - Use **"Good Afternoon"** if it is known that at the customer's location:

a) it is *Summer time* and the *Current time* is between $[11..17)$, or

b) it is *Winter time* and the *Current time* is between $[12..16)$, or

c) *Summer/Winter time* is unknown and the *Current time* is between $[12..16)$

G3 - Use **"Good Evening"** if it is known that at the customer's location:

a) it is *Summer time* and the *Current time* is between $[17..22)$, or

b) it is *Winter time* and the *Current time* is between $[16..21)$, or

c) *Summer/Winter time* is unknown and the *Current time* is between $[17..21)$

G4 - Use **"Good Night"** if it is known that at the customer's location:

a) it is *Summer time* and the *Current time* is between $[22..24)$, or

b) it is *Winter time* and the *Current time* is between $[21..24)$, or

c) *Summer/Winter time* is unknown and the *Current time* is between $[22..24)$.

G5 - Use **"Hello"** if the information known about *Current time* and *Summer/Winter time* is not sufficient to derive one of the greetings using one of the previous rules.

Notice that information about *Summer/Winter time* is irrelevant in cases when *Current time* is inside both ranges for a particular greeting (reflected in (c) rules). However, there are situations in which it is impossible to make one of the first four decisions (e.g., *Current time* is only known to be between 10 and 14). In these cases greeting "Hello" shall be used (rule **G5**).

Salutation decision Following is the precise specification of how the decision about *Salutation* is made based on what is known about customer's *Gender* and *Marital status*:

S1 - Use **"Mr."** if it is known that *Gender* is *Male*.

S2 - Use **"Mrs."** if it is known that *Gender* is *Female* and *Marital status* is *Married*.

S3 - Use **"Ms."** if it is known that *Gender* is *Female* and *Marital status* is *Single*.

S4 - Use **"Ms."** if the *Gender* is known to be *Female* and *Marital status* is unknown.

S5 - Use **"Customer"** if the *Gender* is unknown.

2.2. Extension of the problem

We introduce an extension of the application to show the difference between uncertainty about the value of a variable, and the absence of a value of a variable. Suppose the company provides computer hardware support services, selling, among many other products, Graphic Processing Unit (GPU) cards. Suppose that the message to be sent to customers is promoting material on a new GPU. Depending on the information about the hardware of a customer's personal computer, this message would have different content. Important for this example is that it is possible to have a computer without a dedicated GPU card, working only with integrated graphics. This means that the system can be in (at least[1]) four different epistemic states about the GPU of a customer.

- **Situation 1** - It is known that customer has a *GPU* card, and it is known which one.
- **Situation 2** - It is known that customer has a *GPU* card, but it is not known which one.
- **Situation 3** - It is known that customer does not have a *GPU* card.
- **Situation 4** - It is not known whether customer has *GPU*.

There is a difference between not knowing the value of a variable and knowing that it has no value (e.g., GPU). The decision may differ depending on whether it is known that there is an unknown value or whether it is not known whether there is a value.

[1] It is possible for the system to be in many more epistemic states, for example, if it is known that the customer has a GPU card, but not exactly which one, it is possible to rule out some options based on the information on the other components of the computer.

Message decision Following are the *Message* decisions for each of the situations:

M1 - For situation 1 - Send a message to the customer with the advertisement for a new GPU and provide a performance comparison between the new and the card they own (**Msg1**).

M2 - For situation 2 - Send a message to the customer with the advertisement for a new GPU and provide a form for performance comparison with the new GPU card (**Msg2**).

M3 - For situation 3 - Send a message to the customer with the advertisement for a new GPU, and a discount offer (**Msg3**).

M4 - For situation 4 - Send a poll to determine which, if any, GPU the customer has (**Msg4**).

3. Analysis of the problem

The "*Greeting a Customer with Unknown Data*" challenge is specific as it requires decisions to be made under uncertainty. This would require a practical system equipped with the following:

1. Support for specifying exact knowledge about the input variables.
2. Support for decision rules of the form: "*If the value is unknown the decision is such and so*".
3. Mechanisms for deriving decisions under uncertainty.

To the best of our knowledge, all state-of-the-art rules-based decision modeling languages fail to meet at least some of these requirements. We believe that one of the reasons is the objective interpretation of decision rules (i.e., if the value of the input variable is V, then ...). Alternatively, rules can be interpreted epistemically (i.e., if the value of an input variable is known to be V, then ...). Indeed, decisions must be taken on the basis of, not what **are** the values of the variables in the real world, but what **the user knows** about those values. Of course in the many applications where decisions are taken in the context of complete knowledge, there is no difference. But in applications such as the salutation problem where decisions under uncertainty must be taken, the difference surfaces. Various engineering approaches have been developed to circumvent the limitations of the objective interpretation in practical systems. The main three are discussed in the following paragraphs.

Default values - By default values, we mean the mechanism that would derive a default decision if none of the rules are applicable (due to ignorance). The first issue with this approach is that it is impossible to constrain in the rules that something is unknown. Hence, it is impossible to derive different decisions when different variables are unknown. The "*Greeting a Customer with Unknown Data*" challenge is designed to expose this issue. In particular, *Salutation* has two levels of decision-making with ignorance reflected in rules **S4** and **S5**.

Deriving "safe" decisions - This approach requires some way of expressing that the value of an input variable is unknown. Further, the idea is to compute decisions for every possible value of the unknown variable. If all these decisions coincide then the decision is "safe" to be used. The main downside of this approach is that it does not support custom decisions in case of ignorance (e.g., the decision of the *Salutation* rule **S5**).

Using NULL (or any other value) for representing ignorance - To mitigate the limitations of the previous approach, engineers retreat to polluting the domain of values by introducing an artificial value *Unknown* representing ignorance. This new value informally represents the epistemic state of total ignorance about a variable. The solution proposed by Feldman [2] utilizes this idea (more in Section 8). This approach is significantly limited in that it can express only exact epistemic states or total ignorance (i.e., it is impossible to represent refined levels of epistemic state). In the motivating example, this is needed to represent that Time is known to be between two values. Engineers sometimes use reserved value NULL to represent ignorance. This is dangerous because in most of the languages NULL usually represents the absence of the value. We introduce the extension of the challenge (Section 2.2) so this issue becomes apparent.

Given the overview of existing approaches and their issues, we seek a system that supports the three features mentioned at the beginning of the section without encountering the problems associated with current methods.

4. Evaluation criteria

This section provides evaluation criteria for the particular challenge regarding the correctness and completeness of decision-making and criteria for decision modeling languages supporting uncertainty and undefinedness in general.

4.1. Correctness and completeness

A solution of the "*Greeting a Customer with Unknown Data*" is *correct* if it derives the decision from available knowledge of the values of input variables according to specification from Section 2 (in particular *Greeting* decision, *Salutation* decision, and *Message* decision).

A solution of the "*Greeting a Customer with Unknown Data*" is *complete* if a decision of the three decision variables (i.e., *Greeting*, *Salutation*, and *Message*) are derived in any possible epistemic state about the input variables.

4.2. Criteria for decision modeling language supporting uncertainty

A rule-based decision modeling language supporting uncertainty and undefinedness should meet the following criteria:

C1 - In the input, it must be possible to express enriched information about the value of a variable: a) the exact value; or b) the value is known to be in a set of values; or c) the value is unknown; or d) the value is unexisting.

C2 - In the conditions of rules, it must be possible to express the following constraints about a variable: a) it is known that the variable has a certain value; b) it is known that the value of the variable is (not) in a set of values; c) the exact value of the variable is unknown; d) the variable is known to be (un)defined.

C3 - The system can derive a decision with missing values for input variables.

5. Epistemic DMN (eDMN)

Epistemic DMN (eDMN) [4] was introduced in our previous work with the main goal of defining a precise model semantics of any rule-based decision modeling language. In that paper, we interpret constraints in decision rules as epistemic. Consequently, the language is extended with epistemic constructs. The system supporting eDMN is implemented as an extension of existing *Constraint DMN* (cDMN) [7] language. Hence, we proceed with explaining cDMN.

5.1. Constraint DMN (cDMN)

Constraint DMN (cDMN) [7] was introduced as an extension of classical DMN notation aiming for more expressivity and conceptually clearer modelings. In cDMN there are different kinds of tables for expressing different segments of the model. In the following text, we will briefly discuss these used for solving the challenge in this paper.

Glossary tables are used to define the ontology of the problem, i.e., to specify the symbols that will be used for formalizing the decision. The two types of *Glossary tables* used in this paper are *Types* and *Constants tables*. The *Types table* serve for declaring new types by providing their name, base type (*Integer*, *Real*, *String*, or *Datestring*), and possible values. The *Constants table* declares a set of constants

provided their names and types. The notation of these tables is presented in Figure 1, where type "*Year*" and constant "*Birth year*" are declared.

Types		
Name	**DataType**	**Possible Values**
Year	Int	[1900..2100]

Constants	
Name	**DataType**
Birth year	Year

Figure 1: Example of cDMN *Glossary tables* declaring "*Year*" type and "*Birth year*" constant

Decision tables are the central part of the DMN and cDMN, they are used for specifying decisions in the tabular format. One such table is presented in Figure 2. This table specifies whether a person is entitled to a retirement based on the year of birth. The top-left cell specifies the name of the table. The

Retirement condition		
U	Birth year	Entitled to retirement
1	≤ 1960	Yes
2	> 1960	No

Figure 2: Example of an cDMN *Decision table*

Given Birth year	
D	Birth year
1	1973

Figure 3: Example of an cDMN *Data table*

cell immediately below it (on the leftmost side) contains a hit policy of the table (details in the next paragraph). Green cells contain the names of input variables and blue of the output variables. Columns with green headers are called input columns, while those with blue headers are called output columns. The rows under these columns represent decision rules. The numbers below the hit policy field are the indexes of these rules.

The rules in these tables are interpreted in cDMN as follows: if the constraints (cells of input columns) of some rule are satisfied for a given value of input variables, then the decision variable is derived to have the value of the corresponding rule. Since it is possible that multiple rules are satisfied for the same value of input variables, hit policies are introduced. The hit policy "U" stands for "*Unique*", and it mandates that there is exactly one rule satisfied for given values of input variables. Another hit policy, that is used in this paper, is "F" standing for "*First*", and it uses the first rule (with the lowest index) that is applicable to derive a decision.

The constraints supported in cells of input columns, relevant to this paper, are: equality, comparison, interval, and no constraint. Equality constraint expresses that a variable has a certain value, and it is expressed by filling in the value in the cell. Comparison constraints are used for numerical variables, and they are expressed with symbols $\geq, \leq, <, >$. Interval constraint $[n..m]$ for a numerical variable expresses $\geq n$ and $\leq m$ (open interval is expressed as $(n..m)$). The absence of constraint is represented with the symbol "$-$". Constraints can be connected with "," in the same cell, meaning that at least one of them holds.

Data tables in cDMN are used for expressing values of input variables. In this way, it is possible to separate the decision model from a particular instance of the problem. Such tables use the "D" hit policy to be distinguished. Figure 3 shows one such a table specifying the "*Birth year*" variable to be 1973. Provided this information, it is possible to run the cDMN solver to decide the value of the "Entitled to retirement" decision variable using the *Decision table* from Figure 2, which would yield the decision "No".

5.2. Epistemic interpretation

In DMN (and cDMN) decisions are made based on the objective values of the input variables. Therefore constraints in *Decision tables* are satisfied only when the variable denotes a value that satisfies the constraint. We argued earlier that decisions are made based on epistemic state. An epistemic state can be formally represented as a set of objective worlds. For example, the set of two worlds, in one "*Birth*

year" is 1959 and in another 1961, is one epistemic state where it is known that the "*Birth year*" is either 1959 or 1961. In epistemic interpretation, a constraint is satisfied when it is satisfied in each objective world possible according to an epistemic state.

This shift in the view makes a significant difference. For example, neither of the constraints from *Decision table* in Figure 2 is satisfied for the epistemic state mentioned above, hence there is no decision derived. However, in DMN (and cDMN) there is a decision in each of the world, "Yes" when the "*Birth year*" is 1959 and "No" when it is 1961.

5.3. Extending cDMN with non-denoting constants and epistemic constructs

First, we extend cDMN with partial constants (i.e., constants that do not always denote) by introducing the keyword "*Partial*" which can prefix the declaration of the constant. Following the approach from our previous work [8], a partial constant named "*Var name*" is paired with another constant "*D_Var name*" which takes value Def if "*Var name*" is denoting and $Undef$ when it does not. To ensure that the partial constant is properly used it is required that its "*D_*" predicate is constrained to be Def. Otherwise, no constraint must be applied to the variable. We selected this approach because using reserved value NULL instead of introducing a new constant ("*D_*") leads to difficulties in differentiating between constraints "*It is unknown if the variable is defined*" and "*Variable is known to be defined, but its value is unknown*". This idea is presented in Figure 4. Table 4b declares the partial constant "*First contract*" standing for the year of the first contract. In *Decision table* 4c decision whether a person is entitled to a bonus is defined in terms of "*First contract*". Since it is possible that the first contract does not exist, constant "*D_First contract*" is constrained first.

The next extension introduced with eDMN is support for epistemic constraints in *Decision tables*. The following two constraints are introduced: "$|K|$" expressing that the value of the constrained variable is known, and "$\neg|K|$" expressing that the value of the constrained variable is unknown. An example of "$\neg|K|$" constraint is presented in 4a as a constraint in rule 4 for variable "*D_First contract*" expressing that it is unknown whether the first contract exists.

Bonus for the first contract			
U	D_First contract	First contract	Bonus
1	Def	≤ 1980	Yes
2	Def	> 1980	No
3	Undef	–	No
4	$\neg\|K\|$	–	No

(a) Example of an eDMN *Decision table*

Constants	
Name	**DataType**
Partial First contract	Year

(b) Example of an eDMN *Constants table*

Known about Birth year	
K	Birth year
1	1959,1961

(c) Example of an eDMN *Known table*

Figure 4: Example of eDMN *Glossary*, *Decision*, and *Known tables*

In order to represent different epistemic states of input variables we replace *Data tables* with *Known tables*. Accordingly, the hit policy "D" is replaced with "K". Figure 4c shows an example of *Known table* declaring that "*Birth year*" is known to be either year 1959 or 1961.

6. Solving the challenge using eDMN

The solution to the challenge, described in Section 2, is presented in three parts. The first part describes how the problem ontology is declared (i.e., *Types* and *Constants tables*). The second part is concerned with modeling decisions using eDMN *Decision tables*. The last part is concerned with modeling epistemic stats using *Known tables*.

6.1. Types and Constants declaration

First, we compose the ontology (i.e., type and constant symbols) of the challenge based on the description from Section 2. Additionally, we identified the possible values of the types. For example, *Current time* ranges over integer numbers between 0 and 23. Recall that all variables denote a value with the exception of *GPU* which does not have to exist. This information is formally declared in eDMN with the two *Glossary tables* presented in Figure 5.

Types		
Name	**DataType**	**Possible Values**
time	Int	[0..23]
sw_time	String	Summer, Winter
greeting	String	Good Morning, Good Afternoon, Good Evening, Good Night, Hello
gender	String	Female, Male
marital status	String	Single, Married
salutation	String	Mr, Mrs, Ms, Customer
gpu	String	GA, GB, GC, GD
message	String	Msg1, Msg2, Msg3, Msg4

(a) eDMN *Types table*

Constants	
Name	**DataType**
Time	time
SW Time	sw_time
Greeting	greeting
Gender	gender
M Status	marital status
Salutation	salutation
Message	message
Partial GPU	gpu

(b) eDMN *Constants table*

Figure 5: "*Greeting a Customer with Unknown Data*" eDMN *Glossary tables*

6.2. Modeling decisions

Section 2.1 summarizes the decision making process of the greeting customer challenge. The extension of the problem is specified in Section 2.2. The defined decision variables are *Greeting*, *Salutation*, and *Message*. An important observation is that these specifications define decisions in terms of what is *known* about the input variables. Accordingly, this information is modeled with the three eDMN *Decision tables*, represented in Figure 6.

The eDMN *Greeting* table 6a models the decision process of the *Greeting* decision variable. Rules 1-4 specify greetings for the time intervals for the summer time, and rules 5-8 do the same for the winter time. Rules 9-12 define greeting given the time intervals when the exact Winter/Summer times is unknown (represented with $\neg|K|$). Notice that in this case interval is the intersection of intervals from the rules when the Winter/Summer times is known. Finally, rule 13 specifies that the greeting "Hello" can always be used. However, thanks to the "*First*" (F) hit policy, this rule gets active only if none of the previous rules is satisfied.

The eDMN table 6b specifies decisions for the *Salutation*. The first rule covers the case when *Gender* is known to be male and *Marital status* is irrelevant. Rules 2 and 3 correspond to the exact epistemic state where both *Gender* and *Marital status* of the customer are known. Rules 4 and 5 cover the situations of partial knowledge as described in Section 2.1.

The eDMN table 6c specifies the decision process for the *Message*. Specific for this table is that *GPU* is a partial constant, meaning that it does not necessarily denote a value. Such constants are equipped with a built-in constant symbol derived from the name of the original symbol by adding *D_* in front of the name (as described in Section 5). Here, value *Def* for the constant *D_GPU* means that it is known that constant *GPU* denotes. Similarly, *Undef* stands for epistemic state where it is known that *GPU* does not exist. An important constraint for partial constants (in this case *GPU*) is that their value may only be constrained when they are known to denote (i.e., in this case *D_GPU* is *Def*). The first rule in this table is interesting, it states that it is known that *GPU* exists and moreover, it is known exactly which one ($|K|$ operator) but there is no constraint on the exact value of the *GPU*.

Greeting			
F	Time	SW Time	Greeting
1	$[0..11)$	$Summer$	Good Morning
2	$[11..17)$	$Summer$	Good Afternoon
3	$[17..22)$	$Summer$	Good Evening
4	$[22..24)$	$Summer$	Good Night
5	$[0..12)$	$Winter$	Good Morning
6	$[12..16)$	$Winter$	Good Afternoon
7	$[16..21)$	$Winter$	Good Evening
8	$[21..24)$	$Winter$	Good Night
9	$[0..11)$	$\neg\|K\|$	Good Morning
10	$[12..16)$	$\neg\|K\|$	Good Afternoon
11	$[17..21)$	$\neg\|K\|$	Good Evening
12	$[22..24)$	$\neg\|K\|$	Good Night
13	-	-	Hello

(a) eDMN *Greeting Decision table*

Salutation			
U	Gender	M Status	Salutation
1	Male	-	Mr
2	Female	Married	Mrs
3	Female	Single	Ms
4	Female	$\neg\|K\|$	Ms
5	$\neg\|K\|$	-	Customer

(b) eDMN *Salutation Decision table.*

Message			
U	D_GPU	GPU	Message
1	Def	$\|K\|$	Msg1
2	Def	$\neg\|K\|$	Msg2
3	Undef	-	Msg3
4	$\neg\|K\|$	-	Msg3

(c) eDMN *Message Decision table*

Figure 6: "*Greeting a Customer with Unknown Data*" eDMN *Decision tables* - (a) Defining the value for *Greeting* in terms of what is known about *Time* and *Summer/Winter time* at the customer's location. (b) Defining the value for *Salutation* in terms of what is known about *Gender* and *Marital status* of the customer. (c) Defining the value for *Message* in terms of what is known about *GPU* card.

6.3. Representing the epistemic state

After formally specifying the ontology (i.e., types and constants) of the problem and *Decision tables*, it remains to describe the means of expressing an exact epistemic state about the input variables. This is done with the *Known tables* which allow constraining the value of the constants. One such epistemic state is represented in Figure 7.

K. Time	
K	Time
1	[8..10]

(a) eDMN Time *K table*

K. SW	
K	SW Time
1	Summer

(b) eDMN SW Time *K table*

K. Gender	
K	Gender
1	Male

(c) eDMN Gender *K table*

K. D_GPU	
K	D_GPU
1	Def

(d) eDMN D_GPU *K table*

K. GPU	
K	GPU
1	GA,GB

(e) eDMN GPU *K table*

Figure 7: eDMN *Known tables* - Specification of the epistemic state of the input variables.

Here, table 7a expresses that it is known that constant *Time* ranges between 8 and 10. Tables 7b and 7c express the exact known values of *SW Time* and *Gender*, which are respectively *Summer* and *Male*. Similarly, table 7d expresses that it is known that constant *GPU* is denoting. Finally, table 7e states that it is known that *GPU* is either *GA* or *GB*. Notice that nothing is stated about *Marital Status*, meaning that nothing is known about it.

7. Results

In this section, we provide justification for the correctness and completeness of the proposed solution of the "*Greeting a Customer with Unknown Data*" challenge.

Correctness Towards showing the correctness of the eDMN specification of the challenge (Section 6), Table 1 presents results of computing decisions in five different epistemic states. The rows in the table represent the relation between what is known about the input variables (i.e., Time, Gender, etc.) and values for decision variables (i.e., *Greeting*, *Salutation*, and *Message*). The table contains "$\neg|K|$" when nothing is known about the value of some variable. Multiple values in the cell correspond to the epistemic state where the variable can be any of the listed values. For example, row 2 in the table corresponds to the example from Figure 7. While this table provides a good insight into the correct

Table 1
Results of computing decisions of the eDMN model in different epistemic states.

	Epistemic states						Decisions		
	Time	S/W Time	Gender	M Status	D-GPU	GPU	Greeting	Salutation	Message
1	11	Summer	Male	Married	Yes	GA	G. M.	Mr	Msg1
2	$[8..10]$	Summer	Male	$\neg\|K\|$	Yes	GA, GB	G. M.	Mr	Msg2
3	$[8..10]$	Winter	$\neg\|K\|$	Married	No	-	G. M.	Cus.	Msg3
4	$[8..10]$	$\neg\|K\|$	Female	$\neg\|K\|$	$\neg\|K\|$	-	G. M.	Ms	Msg4
5	$[9..13]$	$\neg\|K\|$	$\neg\|K\|$	Single	Yes	GB	Hello	Cus.	Msg1

behavior of the proposed solution, demonstrating that the modeling is indeed correct is not trivial. The difficulty arises because computing decisions and verifying their values for all possible epistemic states is computationally unfeasible. To illustrate this, consider that if there are n possible states of affairs for the input variables (i.e., assignments of exact values), then there are 2^n epistemic states. In this particular challenge, where Time can be assigned 24 different values, GPU 4 values, and three other variables[2] each 2 values, the total number of possible assignments is 768. Consequently, there are $1.552518092 \times 10^{231}$ possible epistemic states. Because of this limitation, we will show correctness by showing that the tables in Figure 6 accurately model the constraints described in Section 2 and that the semantics of these tables is sound, meaning that rules are deriving decision only if their constraints are true in a given epistemic state.

We show that eDMN tables from Figure 6 accurately model rules specified in Section 2 by providing one-to-one correspondence between the rules:

- The eDMN table 6a models decision process described in **Greeting decision** list from Section 2.1. Rules 1 to 4 in the eDMN table correspond to points **G1-a**, **G2-a**, **G3-a**, and **G4-a** in the list. Rules 5 to 8 in the eDMN table to points **G1-b**, **G2-b**, **G3-b**, and **G4-b**. Rules 9 to 12 in the eDMN table to points **G1-c**, **G2-c**, **G3-c**, and **G4-c**. Rule 13 in the eDMN table corresponds to the rule **G5** from the list.
- The eDMN table 6b models decision process described in **Salutation decision** list from Section 2.1. Each rule i in the eDMN table corresponds to the rule **S**i from the specification.
- The eDMN table 6c models decision process described in **Message decision** list from Section 2.2. Each rule i in the eDMN table corresponds to the rule **M**i from the specification.

It remains to show that the truth value of constraints in an eDMN table according to the formal semantics (defined in [4]) coincides with the description from Section 5.2. Recall, that an epistemic state is a set of possible worlds and constraint is true in some epistemic state if and only if it is true in every possible world of that epistemic state. We show this by analyzing the semantics of eDMN according to [4]:

[2]The D_GPU variable is not included as it is redundant to GPU.

1. An epistemic state is represented as a first-order logic theory T_E constraining the values of input variables. Formally, the set of models (i.e., assignments to variables satisfying the theory) of this theory represents an epistemic state.
2. An eDMN constraint ψ is translated to an epistemic atom $K[T_E][\psi]$.
3. According to OEL, $K[T_E][\psi]$ is true if and only if ψ is entailed by T_E (i.e., $T_E \vdash \psi$) which is equivalent to ψ being true in every possible world that satisfies T_E, which are forming the epistemic state E.

From here it is clear that the semantics of eDMN as defined in [4] coincide with the description from Section 5.2.

Completeness Showing the completeness of the solution is to justify that tables are deriving decisions in every possible epistemic state. This property is achieved by the design of *Decision tables*. The following are justifications of completeness per table:

- *Greeting* - The "*First*" (F) hit-policy is used and the last rule has no constraints. The last rule catches all epistemic states not covered by some of the previous rules.
- *Salutation* - We shall show that each epistemic state of "*Gender*" is covered by some rule (Condition 1), and that for each set of rules with the same condition for "*Gender*" all possible epistemic states are covered for "*Marital Status*" (Condition 2).
  - Gender is constrained with "Male" (rule 1), "Female" (rules 2,3,4), and "$\neg|K|$" (rule 5). Since "Male" and "Female" are the only possible exact epistemic states and "$\neg|K|$" covers all the others, Condition 1 holds.
  - The rules with the same constraint for "*Gender*" are grouped as {1}, {2,3,4}, and {5}. For groups {1} and {5}, "*Marital Status*" is not constrained and hence all epistemic states are covered, i.e., Condition 2 holds for rules 1 and 5. For the group {2,3,4}, rule 2 constrains "*Marital Status*" to "Single", rule 3 to "Married", and rule 4 to "$\neg|K|$". Since "Single" and "Married" are the only possible exact epistemic states and "$\neg|K|$" covers all the others, Condition 2 holds for rules 2, 3, and 4.
- *Message* - Justification is similar to the one for *Salutation Decision table*. Constraints for "*D_GPU*" are "Def", "Undef", and "$\neg|K|$", covering all the possible epistemic states. Constraint is absent for "*GPU*" in rules 3 and 4, while rules 1 and 2 (when "*D_GPU*" is "Def") capture all epistemic states with conditions "$|K|$" and "$\neg|K|$".

An additional requirement for completeness is that *Decision tables* comply with their hit policy. The *Greeting* table uses the "*First*" hit policy which has no issues with multiple rules being satisfied as the first one is selected. However, for the tables with "*Unique*" hit policy no two rules should be satisfied for the same epistemic state. Thanks to the simplicity of *Salutation* and *Message Decision tables*, it is easy to see that all the rules are mutually exclusive, which ensures uniqueness.

8. Related work

As mentioned earlier, the only solution to the challenge is proposed by Feldman [2] using the OpenRules [3] system. This solution utilizes the idea of using the special value $Unknown$ to represent missing data. In Section 3 we already explained that the downside of this approach is the restriction to only total ignorance (i.e., it is not possible to represent different epistemic levels). Additionally, this approach lacks systematicity causing complexities in modeling and prudency for errors. These issues are eliminated in our system as epistemic operators are built in making the modeling more readable and conceptually correct. It is worth noting that our system is a proof of concept, while OpenRules is a more mature

system capable of dealing with other technical issues for this particular problem (e.g., getting current time from the operating system, or fetching customer data from some external source).

Since there are no other solutions to the challenge, we compare other systems with eDMN using the criteria for decision modeling language supporting uncertainty, introduced in Section 4.2. The systems selected for comparison are those most frequently used for solving challenges in the Decision Management Community: Corticon [9], OpenRules [3], DMN [5], and cDMN [7]. Table 2 shows the comparison between the systems (in columns) and the different criteria (in rows). eDMN is the only system supporting reasoning with uncertainty (**C3**). To the best of our knowledge, there are no other

Table 2
Comparison of the existing formalisms regarding modeling criteria from Section 4.2; (✓) indicates the criterion is satisfied, (✗) criterion is not satisfied, and (✱) indicates that criterion is partially satisfied. "Cor" stands for Corticon and "OR" stands for OpenRules.

	Cor	OR	DMN	cDMN	eDMN		Cor	OR	DMN	cDMN	eDMN
C1 a)	✓	✓	✓	✓	✓	**C2** a)	✓	✓	✓	✓	✓
C1 b)	✗	✗	✗	✱	✓	**C2** b)	✗	✗	✗	✗	✓
C1 c)	✱	✱	✱	✱	✓	**C2** c)	✱	✱	✱	✱	✓
C1 d)	✓	✓	✓	✓	✓	**C2** d)	✱	✱	✱	✱	✓

rule-based decision modeling systems supporting uncertainty.

9. Implementation

The implementation of the proof-of-concept eDMN system for this paper is open-source and available in the git repository[3]. This system implements an eDMN translation engine targeting the Order Epistemic Logic (OEL) language, an extension of FO(.) language with hierarchical epistemic operators. The reasoning with OEL specifications is made possible with the IDP-Z3 [10] knowledge base system.

Most of the eDMN system was implemented for the purposes of our previous work [4]. The changes made for this challenge are extensions with partial concepts and support for *Known tables*. The verification of conditions for using partial constants has yet to be implemented. Instructions specific for the modeling described in Section 6 are available in the file[4].

10. Conclusion

In this paper, we use the eDMN decision modeling language to formally model the "*Greeting a Customer with Unknown Data*" challenge. We demonstrate that the proposed solution is both correct and complete. Additionally, we highlight the importance of an epistemic interpretation in rule-based decision modeling languages for properly treating uncertainty. This is reflected in the table comparing eDMN with other state-of-the-art systems.

Acknowledgments

Thanks to Simon Vandevelde and Joost Vennekens for valuable discussions.

References

[1] D. Community, Greeting a customer with unknown data, 2016. URL: https://dmcommunity.org/challenge/challenge-aug-2016/.

[3] https://gitlab.com/krr/idp-z3-oel
[4] https://gitlab.com/krr/idp-z3-oel/-/blob/main/examples/RuleMLRR(2024)/README.md

[2] J. Feldman, Openrules response to dmcommunity challenge aug-2016, 2016. URL: https://dmcommunity.org/wp-content/uploads/2016/09/openrules-response-to-dmcommunity-challenge-aug-20161.pdf.
[3] OpenRules, Inc., Openrules, https://openrules.com/intro.htm, 2024.
[4] D. Marković, S. Vandevelde, L. Vanbesien, J. Vennekens, M. Denecker, An epistemic logic for modeling decisions in the context of incomplete knowledge, in: Proceedings of the 39th ACM/SIGAPP Symposium on Applied Computing, SAC '24, Association for Computing Machinery, New York, NY, USA, 2024, p. 789–793. URL: https://doi.org/10.1145/3605098.3636176. doi:10.1145/3605098.3636176.
[5] Object Modelling Group, Decision model and notation, https://www.omg.org/spec/DMN/, 2023.
[6] H. Vlaeminck, J. Vennekens, M. Bruynooghe, M. Denecker, Ordered epistemic logic: Semantics, complexity and applications, in: Thirteenth International Conference on the Principles of KRR, 2012, pp. 369–379.
[7] S. Vandevelde, B. Aerts, J. Vennekens, Tackling the dm challenges with cdmn: A tight integration of dmn and constraint reasoning, Theory and Practice of Logic Programming 23 (2023) 535–558.
[8] D. Marković, M. Bruynooghe, M. Denecker, Towards systematic treatment of partial functions in knowledge representation, in: S. Gaggl, M. V. Martinez, M. Ortiz (Eds.), Logics in Artificial Intelligence, Springer Nature Switzerland, Cham, 2023, pp. 756–770.
[9] Progress, Corticon, https://www.progress.com/corticon, 2024.
[10] P. Carbonnelle, S. Vandevelde, J. Vennekens, M. Denecker, Interactive configurator with fo(.) and idp-z3, arXiv preprint arXiv:2202.00343 (2022).