Semantic DMN: Decisions Models with Background Knowledge

Diego Calvanese
Joint work with *Marco Montali, Marlon Dumas, Frabrizio M. Maggi*

KRDB Research Centre for Knowledge and Data
Free University of Bozen-Bolzano, Italy

Department of Computing Science
Umeå University, Sweden

KRDB Summer Online Seminars
12 June 2020 – Bolzano, Italy

Outline

Outline

Decision Model and Notation (**DMN**)

Recent OMG standard providing constructs for building decision models:

- First version in September 2015.
- Current version: DMN 1.2 (January 2019).

Decision

Logic used to determine an output value from a number of input values, using one or more rules. Graphically shown in a **decision table**.

Decision requirement graph

Network of DMN decisions, where outputs of some decisions are bound to input of other decisions. Graphically shown in a **decision requirement diagram (DRD)**.

Promotes separation of concerns and integration with BPMN.

Old wine in new bottle?

Yes . . .

Around since end '60s.
[Pooch 1974, ACM Comp. Surv.]
Repeated standardization efforts [CODASYL Decision Table Task Group 1982; Vanthienen and Dries 1994].

. . . with two key provisos:

Standard rule language

Friendly Enough Expression Language.
Two flavours:

- **FEEL** – powerful and textual.
- **S-FEEL** – simple and graphical.

Wide industry adoption

- DMN compliance is a must.
- Steep increase in tools: Oracle, IBM, FICO, Signavio, Camunda, Activiti, Trisotech, OpenRules, Sparkling logic, Red Hat, . . .

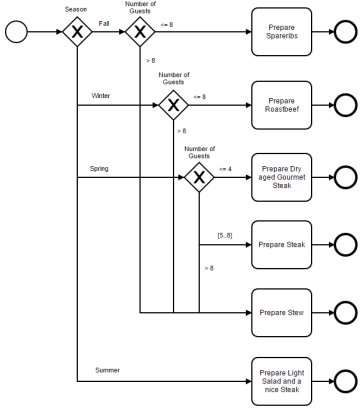
We focus on S-FEEL!

Success factor #1: Timeliness

Organizations are increasingly process-oriented.

- DMN encourages separation of concerns between the process logic and the decision logic.
- Clarity, modularity, reusability.

From BPMN ...

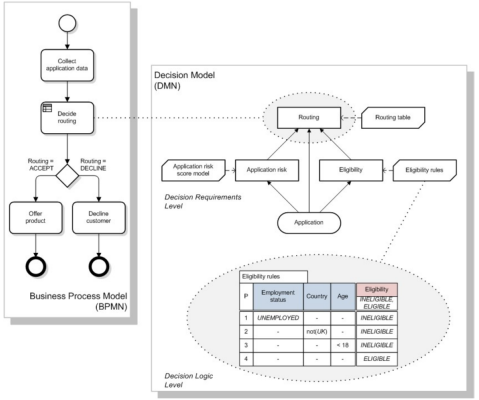


Success factor #1: Timeliness

Organizations are increasingly process-oriented.

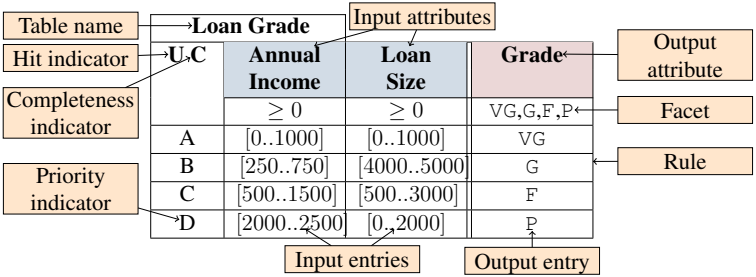
- DMN encourages separation of concerns between the process logic and the decision logic.
- Clarity, modularity, reusability.

. . . to BPMN+DMN



Success factor #2: Understandability

S-FEEL rules have a simple graphical representation in the form of a table.



(Single) hit policies:

- **unique** hit policy (U) – rules do not overlap;
- **any** hit policy (A) – multiple overlapping rules triggered simultaneously compute exactly the same output values;
- **priority** hit policy (P) – whenever multiple overlapping rules simultaneously trigger, the matching rule with highest output priority is considered.

Outline

A simple DMN S-FEEL decision table

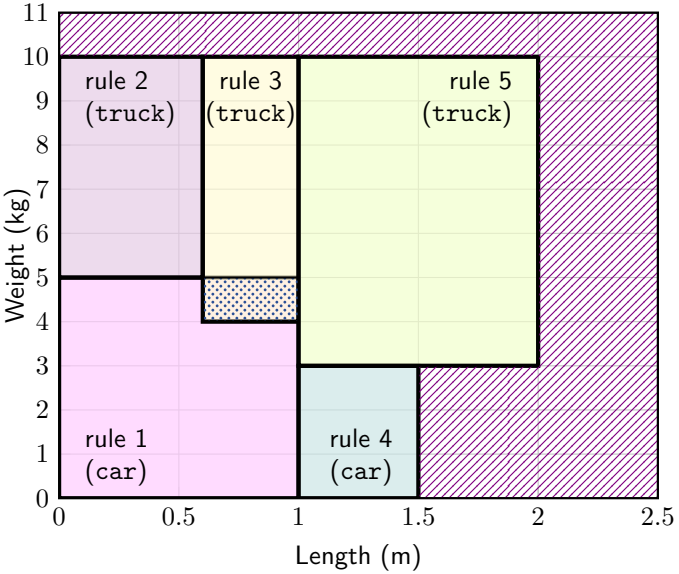
Use the physical features of a package to decide shipment mode.

Package Shipment			
P	Length (m)	Weight (kg)	ShipBy
	> 0	> 0	`car, truck`
1	(0.0,1.0]	(0, 5]	`car`
2	(0.0,0.6]	(5,10]	`truck`
3	(0.6,1.0]	(4,10]	`truck`
4	(1.0,1.5]	(0, 3]	`car`
5	(1.0,2.0]	(3,10]	`truck`

Question

What can we say about the logic of this decision?

Geometric Intuition



Incomplete

There are inputs with no matching rule.

Multiple hits

There are overlapping rules with different outputs.
P is a reasonable hit policy.

DMN semantics and analysis [__, Dumas, et al. 2016, 2018, BPM, IS]

1. Logic-based semantics of S-FEEL tables

Multi-sorted FOL encoding of S-FEEL conditions and table rules.

2. Logic-based formalization of analysis tasks

3. Implementation

DMN semantics and analysis [__, Dumas, et al. 2016, 2018, BPM, IS]

1. Logic-based semantics of S-FEEL tables

2. Logic-based formalization of analysis tasks

Quantified formulae capturing table properties:

- compatibility between conditions and attribute facets;
- completeness;
- adequacy of hit policy: does the indicated policy reflect the table semantics?

3. Implementation

DMN semantics and analysis [__, Dumas, et al. 2016, 2018, BPM, IS]

1. Logic-based semantics of S-FEEL tables

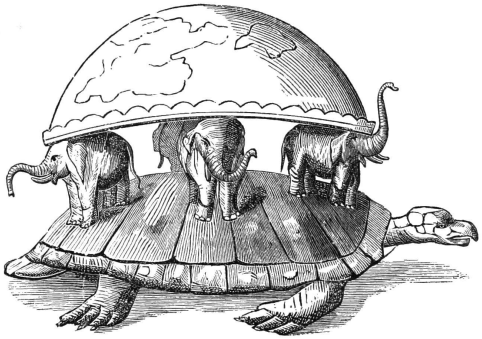
2. Logic-based formalization of analysis tasks

3. Implementation

- In principle, 1+2 directly enable the use of SMT solvers for analysis.
- In practice:
  - We interpret rules geometrically (hyperrectangles).
  - We apply state-of-the-art sweep-line algorithms to the analysis and simplification of tables. Complexity: linear in columns, (sub)quadratic in rules.
  - Impressive performance. E.g., detecting missing rules requires
    - from 160ms for tables with 500 rules and 3 cols ...
    - ... to 11mins for tables with 1500 rules and 15 cols.

Decisions are not alone!

Organization



Strategic Management	Business Process Management	Master Data Management	Enterprise Decision Management
Goals and resources	Operational processes	Relevant facts	Strategic decisions

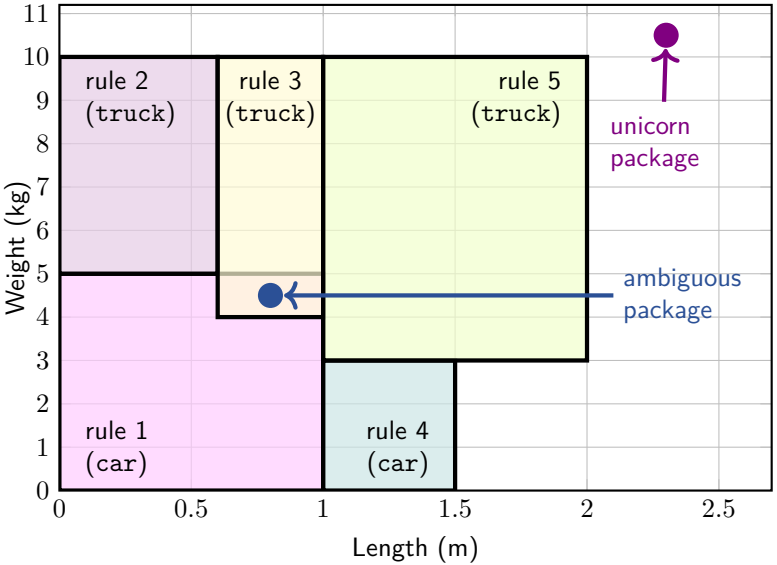
Putting decisions in perspective

Key questions

- How to **integrate** decision models and other organizational pillars?
- What is the **impact on** the **decision logic**?
- Which **analysis tasks** emerge?
- What is their **decidability / complexity**?
- How to **algorithmically** attack them?

Outline

Which packages exist?



Packages within an organization

Delivery companies never offer packages of *arbitrary type*.

Background knowledge about packages

A1	There are *only* two types of packages: standard and special.
A2	The minimum weight for a package is 0.5 kg.
A3	A standard package has a length of 0.5 m and bears at most 8 kg.
A4	A special package has a length of 1.2 m and bears at most 9 kg.

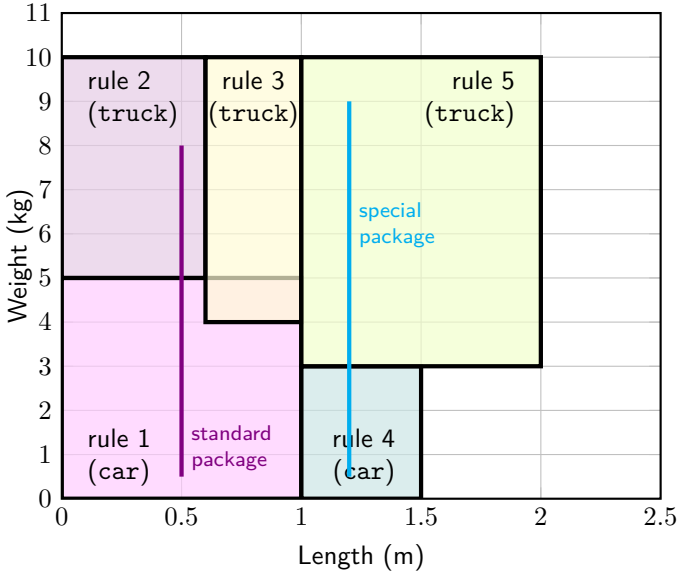
Warning

This is **not** a decision table. This is an **ontology of packages**!

Question

What happens if the package shipment table is interpreted in the context of this background knowledge?

Decision in the context of background knowledge



Complete

A standard / special package always matches with a rule.

Unique hit

A standard / special package with a given weight matches with a single rule.
Hence, **P** is a useless policy.

Output

Computable from package type + weight.

A more complex example

Inspired by the Ship and Port Facility Security Code:

- Ship clearance in the Netherlands.
- March 2016 challenge at `dmcommunity.org`.

Knowledge of ships

There are several types of ships, characterized by:
length (in m); draft size (in m); capacity (in TEU).

Ship KB

Ship Type	**Short Name**	**Length** (m)	**Draft** (m)	**Capacity** (TEU)
Converted Cargo Vessel	*CCV*	135	0 – 9	500
Converted Tanker	*CT*	200	0 – 9	800
Cellular Containership	*CC*	215	10	1000 – 2500
Small Panamax Class	*SPC*	250	11 – 12	3000
Large Panamax Class	*LPC*	290	11 – 12	4000
Post Panamax	*PP*	275 – 305	11 – 13	4000 – 5000
Post Panamax Plus	*PPP*	335	13 – 14	5000 – 8000
New Panamax	*NP*	397	15.5	11 000 – 14 500

Warning!

This is **not** a decision table!
This is a set of **constraints** relating the ship types with corresponding possible dimensions.

Clearance rules

A vessel may enter a port if:

- it is equipped with a valid certificate of registry;
- it meets the safety requirements.

Valid certificate of registry

Certificate expiration date $>$ current date.

Safety requirements

Based on ship characteristics and the amount of residual cargo:

- Small ships (with length < 260 m and draft < 10 m) may enter only if their capacity is < 1000 TEU.
- Ships with a small length (< 260 m), medium draft (≥ 10 m and ≤ 12 m), and capacity < 4000 TEU, may enter only if their carried residuals have $\leq 0.75\,\mathrm{mg/cm^2}$ dry weight.
- Medium-sized ships (with length ≥ 260 m and < 320 m, and draft > 10 m and ≤ 13 m), and with a capacity < 6000 TEU, may enter only if their carried residuals have $\leq 0.5\,\mathrm{mg/cm^2}$ dry weight.
- Big ships (with length ≥ 320 m and < 400 m, and draft ≥ 13 m), and capacity > 4000 TEU, may enter only if their carried residuals have $\leq 0.25\,\mathrm{mg/cm^2}$ dry weight.

Clearance rules in DMN S-FEEL

Vessel Clearance						
C U	***Cer. Exp.*** (date)	***Length*** (m)	***Draft*** (m)	***Capacity*** (TEU)	***Cargo*** (mg/cm^2)	***Enter***
	≥ 0	≥ 0	≥ 0	≥ 0	≥ 0	**Y,N**
1	$\leq$ today	–	–	–	–	**N**
2	$>$ today	$<$260	$<$10	$<$1000	–	**Y**
3	$>$ today	$<$260	$<$10	$\geq$1000	–	**N**
4	$>$ today	$<$260	[10,12]	$<$4000	$\leq$0.75	**Y**
5	$>$ today	$<$260	[10,12]	$<$4000	$>$0.75	**N**
6	$>$ today	[260,320)	(10,13]	$<$6000	$\leq$0.5	**Y**
7	$>$ today	[260,320)	(10,13]	$<$6000	$>$0.5	**N**
8	$>$ today	[320,400)	$\geq$13	$>$4000	$\leq$0.25	**Y**
9	$>$ today	[320,400)	$\geq$13	$>$4000	$>$0.25	**N**

Key questions

- Is the hit indicator correct?
- Is the table complete?
- Do we need all the input data for a ship to apply the decision?

Clearance rules in DMN S-FEEL

Vessel Clearance						
C U	***Cer. Exp.*** (date)	***Length*** (m)	***Draft*** (m)	***Capacity*** (TEU)	***Cargo*** (mg/cm^2)	***Enter***
	≥ 0	≥ 0	≥ 0	≥ 0	≥ 0	**Y,N**
1	$\leq$ today	–	–	–	–	**N**
2	$>$ today	$<$260	$<$10	$<$1000	–	**Y**
3	$>$ today	$<$260	$<$10	$\geq$1000	–	**N**
4	$>$ today	$<$260	[10,12]	$<$4000	$\leq$0.75	**Y**
5	$>$ today	$<$260	[10,12]	$<$4000	$>$0.75	**N**
6	$>$ today	[260,320)	(10,13]	$<$6000	$\leq$0.5	**Y**
7	$>$ today	[260,320)	(10,13]	$<$6000	$>$0.5	**N**
8	$>$ today	[320,400)	$\geq$13	$>$4000	$\leq$0.25	**Y**
9	$>$ today	[320,400)	$\geq$13	$>$4000	$>$0.25	**N**

Hit indicator

Unique hit: **yes**!

Completeness

- **no** if table considered *in isolation*;
- **yes** if understood *in the context* of the **ship KB**.

Clearance rules in DMN S-FEEL

Vessel Clearance						
C U	***Cer. Exp.*** (date)	***Length*** (m)	***Draft*** (m)	***Capacity*** (TEU)	***Cargo*** (mg/cm^2)	***Enter***
	≥ 0	≥ 0	≥ 0	≥ 0	≥ 0	**Y,N**
1	$\leq$ today	–	–	–	–	**N**
2	$>$ today	$<$260	$<$10	$<$1000	–	**Y**
3	$>$ today	$<$260	$<$10	$\geq$1000	–	**N**
4	$>$ today	$<$260	[10,12]	$<$4000	$\leq$0.75	**Y**
5	$>$ today	$<$260	[10,12]	$<$4000	$>$0.75	**N**
6	$>$ today	[260,320)	(10,13]	$<$6000	$\leq$0.5	**Y**
7	$>$ today	[260,320)	(10,13]	$<$6000	$>$0.5	**N**
8	$>$ today	[320,400)	$\geq$13	$>$4000	$\leq$0.25	**Y**
9	$>$ today	[320,400)	$\geq$13	$>$4000	$>$0.25	**N**

Do we need all physical characteristics of a ship for clearance?

- From **ship type**, using the ship KB one can **infer partial information** about length, draft, capacity.
- Combined with **certificate expiration** and **cargo residuals**, **this is enough** to unambiguously apply the decision table!

Sources of decision knowledge

- **S-FEEL DMN Decisions**. Defined by the standard.
- **Knowledge Base**. Multi-sorted FOL theory FOL($\mathfrak{D}$).
  - Quantification domain: objects Δ + data values from different sorts $\mathfrak{D}$ capturing S-FEEL data types (with comparison predicates).
  - Class: unary predicate interpreted over Δ.
  - Role: Binary predicate relating pairs of objects from Δ.
  - Feature: Binary predicate relating objects from Δ to data values from a selected data type in $\mathfrak{D}$.

  Closed formulae interpreted as axioms.

Example

Ship Type	**Short Name**	**Length** (m)	**Draft** (m)	**Capacity** (TEU)
...	*CCV*	135	0 – 9	500

$$\forall s.\mathit{CCV}(s) \rightarrow \mathit{Ship}(s) \wedge \forall \ell.(\mathit{length}(s,\ell) \rightarrow \ell = 135) \wedge$$
$$\forall d.(\mathit{draft}(s,d) \rightarrow d \geq 0 \wedge d \leq 9) \wedge \forall c.(\mathit{capacity}(s,c) \rightarrow c = 500)$$

Combining decisions and KBs in 3 steps

Step 1. Decision tables apply to objects of some class

Identification of the "**bridge class**" that is subject at once to the constraints of the KB and the decision logic.

Example

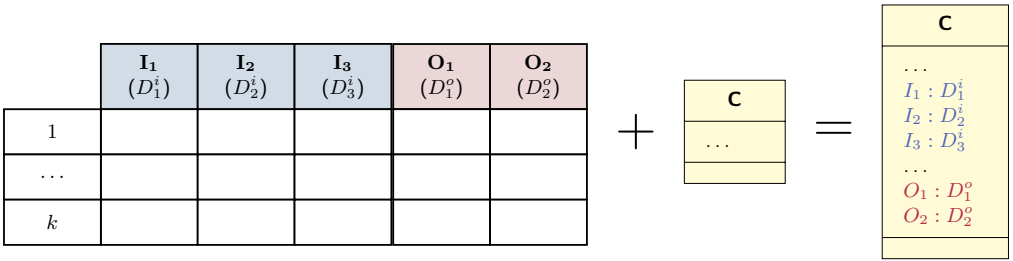
Ship is the bridge class linking the Ship KB to the Vessel Clearance decision table.

Combining decisions and KBs in 3 steps

Step 2. Decision tables enrich the vocabulary of the KB

Table inputs/outputs denote features of the bridge class:

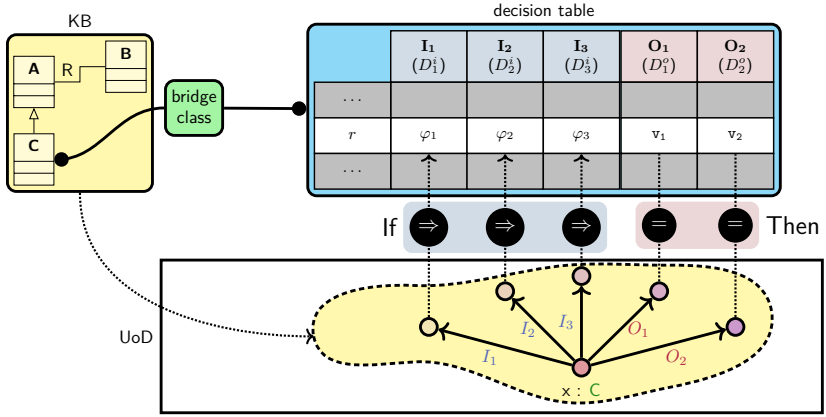
- Each input I becomes an input feature I.
  - If already used in the KB: type compatibility.
- Each output O becomes an output feature O.
  - A new feature, not already used in the KB.



Combining decisions and KBs in 3 steps

Step 3: Combined reasoning

- KB: constrains (some) of the table input features.
- Decision: relates constrained input features to output features.



Ships strike back

Vessel Clearance

Cer.Exp. (date) Real	**Length** (m) Real	**Draft** (m) Real	**Capacity** (TEU) Real	**Cargo** (mg/cm^2) Real	**Enter** `Y`, `N` Bool
9 rules					

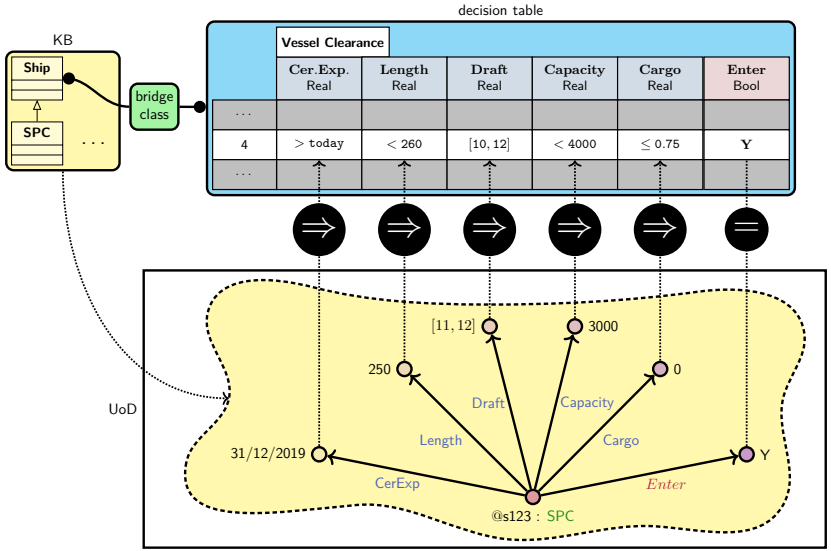
+

Ship
Length : Real
Draft : Real
Capacity : Real

=

Ship
Length : Real
Draft : Real
Capacity : Real
Cer.Exp. : Real
Cargo : Real
Enter : Bool

An empty Panamax Ship approaches the harbor . . .



Decision knowledge bases

A **decision knowledge base** over datatypes $\mathfrak{D}$ ($\mathfrak{D}$-DKB, or DKB for short) ...

... is a tuple $\langle \Sigma, \mathcal{T}, \mathcal{M}, C, A \rangle$, where:

- $\mathcal{T}$ is a FOL($\mathfrak{D}$) intensional KB with signature Σ.
- $\mathcal{M}$ is a DMN decision table that satisfies the following two typing conditions:
  output uniqueness: no output attribute of $\mathcal{M}$ is part of Σ;
  input type compatibility: for every binary predicate $P \in \Sigma$ whose name coincides with an input attribute of $\mathcal{M}$, their types coincide.
- $C \in \Sigma$ is the *bridge class*.
- A is an ABox over the extended signature $\Sigma \cup \mathcal{M}.I$.

Input/output configuration

Input/output configurations for $\mathcal{M}$ are now simply set of facts over an object of type C.

Reasoning tasks: Compatibility with Hit Indicators

Compatibility with Unique Hit

Input: DKB $\mathcal{X} = \langle \Sigma, \mathcal{T}, \mathcal{M}, C, \emptyset \rangle$ (intensional, no data).
Question: Is it the case that no two rules in $\mathcal{M}$ overlap?

Compatibility with Any Hit

Input: DKB $\mathcal{X} = \langle \Sigma, \mathcal{T}, \mathcal{M}, C, \emptyset \rangle$ (intensional, no data).
Question: Is it the case that no two rules in $\mathcal{M}$ that produce different outputs overlap?

Compatibility with Priority Hit

Input: DKB $\mathcal{X} = \langle \Sigma, \mathcal{T}, \mathcal{M}, C, \emptyset \rangle$ (intensional, no data).
Question: Is it the case that no rule in $\mathcal{M}$ is masked by another rule?

Table completeness

Input: DKB $\mathcal{X} = \langle \Sigma, \mathcal{T}, \mathcal{M}, C, \emptyset \rangle$ (intensional, no data).
Question: Does every possible input configuration match a rule in $\mathcal{M}$?

Reasoning tasks: I/O behavior

I/O relationship

Input:
- DKB $\mathcal{X} = \langle \Sigma, \mathcal{T}, \mathcal{M}, C, A \rangle$,
- object $\mathtt{o} \in \Delta$ of type C,
- output attribute $\mathbf{b}$ of $\mathcal{M}$,
- value $\mathtt{v}$ with type that of $\mathbf{b}$.

Question: Is it the case that $\mathcal{X}$ assigns value $\mathtt{v}$ to object $\mathtt{o}$ for attribute $\mathbf{b}$?

Output coverage

Input:
- DKB $\mathcal{X} = \langle \Sigma, \mathcal{T}, \mathcal{M}, C, \emptyset \rangle$ (intensional, no data),
- output attribute $\mathbf{b}$ of $\mathcal{M}$,
- value $\mathtt{v}$ with type that of $\mathbf{b}$.

Question: Is there an input configuration that leads to assign $\mathtt{v}$ to $\mathbf{b}$?

Output determinability

Input:
- DKB $\mathcal{X} = \langle \Sigma, \mathcal{T}, \mathcal{M}, C, \emptyset \rangle$ (intensional, no data),
- unary formula $\varphi(x)$ characterising an input template.

Question: Does $\mathcal{M}$ assign an output to each object of type C that satisfies the formula $\varphi(x)$?

Outline

How to reason?

Question

Is a DKB different from a conventional KB?

Observation

Decision table = a set of additional axioms over the bridge class.

From a DKB to a KB

Given a DKB $\langle \Sigma, \mathcal{T}, \mathcal{M}, C, A \rangle$, construct a conventional KB as follows:

1. Take $\mathcal{T}$ as the initial KB.
2. Encode the attributes of $\mathcal{M}$:
   a. Expand the vocabulary Σ of $\mathcal{T}$ with input/output features from $\mathcal{M}$.
   b. Generate typing and facet axioms for such features.
3. Encode the rules of $\mathcal{M}$: each rule becomes an axiom.

Goal

Reasoning over DKBs as standard reasoning over KBs.

Encoding of attributes (1/2)

Extending the signature

- A feature for each input attribute of the decision that is not already used in the KB.
- A feature for each output attribute.

Example

Vessel Clearance					
Cer.Exp. Real	**Length** Real	**Draft** Real	**Capacity** Real	**Cargo** Real	**Enter** Bool

- Attributes *Length*, *Draft*, *Capacity* correspond to compatible facets in the background KB;
- 2 new features for *CerExp* and *Cargo*;
- 1 new feature for *Enter*.

Encoding of attributes (2/2)

Constraining the features

For each input/output feature, add:

- Typing axiom: the domain of the feature is the bridge concept.
- Functionality axiom: no two attributes of the same kind.
  - For input features: non-ambiguous application of rules.
  - For output features: simply asserts that an output cell contains a single value.

Example

$$\forall x, y.\mathit{length}(x, y) \rightarrow \mathit{Ship}(x)$$
$$\forall x, y, z.\mathit{length}(x, y) \wedge \mathit{length}(x, z) \rightarrow y = z$$

Encoding of S-FEEL conditions

An S-FEEL condition is a compact representation of a unary FOL($\mathfrak{D}$) formula applied to data values.

S-FEEL translation function

Given an S-FEEL condition Q, function $\tau^x(Q)$ builds a unary FOL($\mathfrak{D}$) formula that encodes the application of Q to x.

$$\tau^x(Q) \triangleq \begin{cases} true & \text{if } \mathcal{Q} = \text{“}-\text{”} \\ x \neq \mathtt{v} & \text{if } \mathcal{Q} = \text{“}\mathtt{not(v)}\text{”} \\ x = \mathtt{v} & \text{if } \mathcal{Q} = \text{“}\mathtt{v}\text{”} \\ x \approx \mathtt{v} & \text{if } \mathcal{Q} = \text{“}\approx \mathtt{v}\text{” and } \approx \in \{<, >, \leq, \geq\} \\ x > \mathtt{v}_1 \wedge x < \mathtt{v}_2 & \text{if } \mathcal{Q} = \text{“}(\mathtt{v}_1..\mathtt{v}_2)\text{”} \\ \ldots & \text{(similarly for the other types of intervals)} \\ \tau^x(\mathcal{Q}_1) \vee \tau^x(\mathcal{Q}_2) & \text{if } \mathcal{Q} = \text{“}\mathcal{Q}_1,\mathcal{Q}_2\text{”} \end{cases}$$

Encoding of attribute facets

Restrict the acceptable values

For each input/output feature, add:

- Facet axiom: restricts the acceptable values of the feature range.
  - The facet is an S-FEEL condition: just translate it to get the constraint.

Example

Length Real
≥ 0

$$\forall x, y.\mathit{length}(x, y) \rightarrow \tau^{y}(' \geq 0')y \geq 0$$

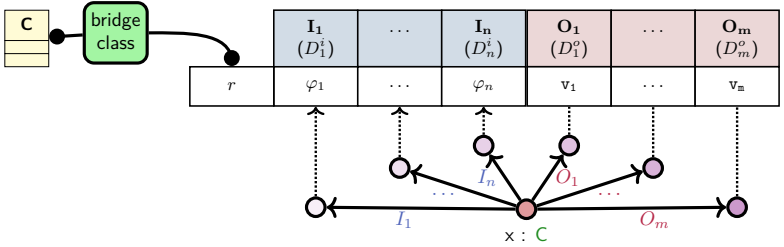
Encoding of rules

Rules as logical implications

For every instance of the bridge class:

if each input feature satisfies the corresponding input cell condition

then each output feature points to the value in the corresponding output cell.



$$\forall x.C(x) \wedge \underbrace{\bigwedge_{j\in\{1,\ldots,n\}} (\exists y_j.I_j(x,y_j) \wedge \tau^{y_j}(\varphi_j))}_{r.\mathsf{If}} \rightarrow \underbrace{\bigwedge_{k\in\{1,\ldots,m\}} (\exists z_k.O_k(x,z_k) \wedge z_k = \mathtt{v_k})}_{r.\mathsf{Then}}$$

Encoding of rules – Example

Example

	Vessel Clearance					
	Cer.Exp. Real	**Length** Real	**Draft** Real	**Capacity** Real	**Cargo** Real	**Enter** Bool
2	$>$ today	< 260	< 10	< 1000	–	Y

Encoding of rule #2

$$\forall x, e, l, d, c.\ \mathit{cerExp}(x, e) \ \wedge\ e > \texttt{today} \ \wedge\ \mathit{length}(x, \ell) \ \wedge\ \ell < 260 \ \wedge\ \mathit{draft}(x, d) \ \wedge\ d < 10 \ \wedge\ \mathit{capacity}(x, c) \ \wedge\ c < 1000 \ \rightarrow\ \exists o.\mathit{enter}(x, o) \ \wedge\ o = \texttt{Y}.$$

Encoding reasoning tasks: Compatibility with Hit Indicators (1/2)

Compatibility with Unique Hit

Input: DKB $\mathcal{X} = \langle \Sigma, \mathcal{T}, \mathcal{M}, C, \emptyset \rangle$ (intensional, no data).
Question: Is it the case that no two rules in $\mathcal{M}$ overlap?

$$\tau(\mathcal{X}) \stackrel{?}{\models} \bigwedge_{r_1, r_2 \in \mathcal{M}.R \text{ s.t. } r_1 \neq r_2} \neg \exists x. \Big(\tau^x(r_1.\mathsf{If}) \wedge \tau^x(r_2.\mathsf{If}) \Big)$$

Compatibility with Any Hit

Input: DKB $\mathcal{X} = \langle \Sigma, \mathcal{T}, \mathcal{M}, C, \emptyset \rangle$ (intensional, no data).
Question: Is it the case that no two rules in $\mathcal{M}$ that produce different outputs overlap?

$$\tau(\mathcal{X}) \stackrel{?}{\models} \bigwedge_{\substack{r_1, r_2 \in \mathcal{M}.R \text{ s.t.} \\ r_1 \text{ and } r_2 \text{ differ in an output}}} \neg \exists x. \Big(\tau^x(r_1.\mathsf{If}) \wedge \tau^x(r_2.\mathsf{If}) \Big)$$

Encoding reasoning tasks: Compatibility with Hit Indicators (2/2)

Compatibility with Priority Hit

Input: DKB $\mathcal{X} = \langle \Sigma, \mathcal{T}, \mathcal{M}, C, \emptyset \rangle$ (intensional, no data).
Question: Is it the case that no rule in $\mathcal{M}$ is masked by another rule?

$$\tau(\mathcal{X}) \stackrel{?}{\models} \bigwedge_{r_1, r_2 \in \mathcal{M}.R \text{ s.t. } r_1 \prec r_2} \exists x. \Big(\tau^x(r_2.\mathsf{If}) \wedge \neg \tau^x(r_1.\mathsf{If}) \Big)$$

Table completeness

Input: DKB $\mathcal{X} = \langle \Sigma, \mathcal{T}, \mathcal{M}, C, \emptyset \rangle$ (intensional, no data).
Question: Does every possible input configuration match a rule in $\mathcal{M}$?

$$\tau(\mathcal{X}) \stackrel{?}{\models} \forall x. C(x) \rightarrow \bigvee_{r \in \mathcal{M}.R} \tau^x(r.\mathsf{If})$$

Encoding reasoning tasks: I/O behavior (1/2)

I/O relationship

Input:
- DKB $\mathcal{X} = \langle \Sigma, \mathcal{T}, \mathcal{M}, C, A \rangle$,
- object $\mathtt{o} \in \Delta$ of type C,
- output attribute $\mathbf{b}$ of $\mathcal{M}$,
- value $\mathtt{v}$ with type that of $\mathbf{b}$.

Question: Is it the case that $\mathcal{X}$ assigns value $\mathtt{v}$ to object $\mathtt{o}$ for attribute $\mathbf{b}$?

$$\tau(\mathcal{X}) \stackrel{?}{\models} \mathbf{b}(\mathtt{o}, \mathtt{v})$$

Output coverage

Input:
- DKB $\mathcal{X} = \langle \Sigma, \mathcal{T}, \mathcal{M}, C, \emptyset \rangle$ (intensional, no data),
- output attribute $\mathbf{b}$ of $\mathcal{M}$,
- value $\mathtt{v}$ with type that of $\mathbf{b}$.

Question: Is there an input configuration that leads to assign $\mathtt{v}$ to $\mathbf{b}$?

$$\tau(\mathcal{X}) \stackrel{?}{\models} \exists x.\mathbf{b}(x, \mathtt{v})$$

Encoding reasoning tasks: I/O behavior (2/2)

Output determinability

Input:
- DKB $\mathcal{X} = \langle \Sigma, \mathcal{T}, \mathcal{M}, C, \emptyset \rangle$ (intensional, no data),
- unary formula $\varphi(x)$ characterising an input template.

Question: Does $\mathcal{M}$ assign an output to each object of type C that satisfies the formula $\varphi(x)$?

$$\tau(\mathcal{X}) \stackrel{?}{\models} \forall x.C(x) \wedge \varphi(x) \rightarrow \bigwedge_{\mathbf{b} \in \mathcal{M}.O} \exists y.\mathbf{b}(x, y)$$

Outline

Reasoning over DKBs as standard reasoning over FOL KBs

Fact

All DKB reasoning tasks can be turned into **logical implication** tests in FOL($\mathfrak{D}$).

Computationally, this is of no help.

Goal

Investigate suitable fragments of FOL($\mathfrak{D}$) that:

- Are expressive enough to encode DMN DRGs + S-FEEL decisions.
- Are computationally feasible (with complexity guarantees).

Setting

Description logics with data types are the natural candidate for this.

The logic $\mathcal{ALCH}(\mathfrak{D})$ [Ortiz et al. 2008], [__, Montali, et al. 2019, TPLP]

Based on the well-known DL $\mathcal{ALC}$ extended with multiple data types that do not interact with each other.

Theorem

Let $\mathfrak{D}$ be a set of datatypes such that for all datatypes $\mathcal{D} \in \mathfrak{D}$ checking $\mathcal{D}$-satisfiability is decidable in ExpTime. Then, reasoning over $\mathcal{ALCH}(\mathfrak{D})$ KBs is ExpTime-complete.

$\mathcal{ALCH}(\mathfrak{D})$ DKBs

Decision Knowledge Bases where background knowledge is expressed as an $\mathcal{ALCH}(\mathfrak{D})$ ontology.

Key Observation

- All constraints seen so far can be encoded in $\mathcal{ALCH}(\mathfrak{D})$.
- Each S-FEEL rule becomes a subsumption assertion in $\mathcal{ALCH}(\mathfrak{D})$.

Encoding S-FEEL rules into $\mathcal{ALCH}(\mathfrak{D})$

Example

	Vessel Clearance					
	Cer.Exp. Real	**Length** Real	**Draft** Real	**Capacity** Real	**Cargo** Real	**Enter** Bool
2	$>$ today	< 260	< 10	< 1000	$-$	Y

Encoding of rule #2 in FOL($\mathfrak{D}$)

$$\forall x, e, l, d, c.\ \mathit{cerExp}(x, e) \ \wedge\ e > \texttt{today} \ \wedge\ \mathit{length}(x, l) \ \wedge\ l < 260 \ \wedge\ \mathit{draft}(x, d) \ \wedge\ d < 10 \ \wedge\ \mathit{capacity}(x, c) \ \wedge\ c < 1000 \ \rightarrow\ \exists o.\mathit{enter}(x, o) \ \wedge\ o = \texttt{Y}.$$

Encoding of rule #2 in $\mathcal{ALCH}(\mathfrak{D})$

$$\forall \mathit{cerExp}.\mathit{real}[>_{\texttt{today}}] \ \sqcap\ \forall \mathit{length}.\mathit{real}[<_{260}] \ \sqcap\ \forall \mathit{draft}.\mathit{real}[<_{10}] \ \sqcap\ \forall \mathit{capacity}.\mathit{real}[<_{1000}] \ \sqsubseteq\ \exists \mathit{enter}.\mathit{string}[=_{\texttt{Y}}]$$

Main results: Complexity

Theorem

Consider an $\mathcal{ALCH}(\mathfrak{D})$ DKB. The encoding into FOL($\mathfrak{D}$) is logically equivalent to the encoding into $\mathcal{ALCH}(\mathfrak{D})$.

Theorem

All DKBs reasoning tasks can be decided in ExpTime for $\mathcal{ALCH}(\mathfrak{D})$ DKBs.

Proof.

Reduction from each reasoning task to a polynomial number of instance or subsumption checks w.r.t. an $\mathcal{ALCH}(\mathfrak{D})$ KB, each of which can be decided in ExpTime. □

UML + S-FEEL DMN = OMG^2

Similar results can be obtained using $\mathcal{ALCQI}$ as the base logic.
$\mathcal{ALCQI}$ is the DL that captures UML class diagrams.

Main Results: Actual Reasoning

OWL 2 standard reasoners work

- $\mathcal{ALCH}(\mathfrak{D})$ datatypes come with unary predicates only.
- Hence $\mathcal{ALCH}(\mathfrak{D})$ DKBs can be directly represented as OWL 2 ontologies.

Datatypes fading away

All reasoning tasks over intensional $\mathcal{ALCH}(\mathfrak{D})$ DKBs (no data) can be encoded into standard $\mathcal{ALCH}$ reasoning tasks without datatypes.

- In the compilation process, datatype reasoning is invoked.
- Open whether this gives an improvement over OWL 2 reasoners.

Lightweight DKBs

S-FEEL decisions: expressible in the lightweight DL *DL-Lite*$_{bool}^{(\mathcal{HN})}(\mathfrak{D})$.

- Not enough to capture DRGs.
- Lightweight DLs with datatypes are less investigated than their more expressive companions.

Outline

Conclusions

- We have introduced **Decision Knowledge Bases** (DKBs), as a conceptual framework to integrate DMN complex decisions with background knowledge.
- We have provided a formalization of DKBs and their reasoning tasks in multi-sorted FOL.
- When the background knowledge is expressed in DLs, we have shown how to encode DKBs in an expressive DL with (unary) datatypes:
  - Reasoning stays in ExpTime (and is ExpTime-complete).
  - We can use state-of-the-art OWL 2 reasoners for effective inference.
- We have presented the formalization and encoding only for complex DMN decisions, but the framework extends also to Decision Requirement Graphs (DRGs) – See [__, Montali, et al. 2019, TPLP].
- We are also investigating the possibility to use a lightweight DL extended with datatypes for the encoding, which would lead to polynomial reasoning.

unibz

Thank you for your attention!

References I

[1] U. W. Pooch. "Translation of Decision Tables". In: *ACM Computing Surveys* 6.2 (1974), pp. 125–151.

[2] CODASYL Decision Table Task Group. *A Modern Appraisal of Decision Tables: a CODASYL Report*. Tech. rep. 1982.

[3] J. Vanthienen and E. Dries. "Illustration of a Decision Table Tool for Specifying and Implementing Knowledge Based Systems". In: *Int. J. on Artificial Intelligence Tools* 3.2 (1994), pp. 267–288.

[4] ___, M. Dumas, Ü. Laurson, F. M. Maggi, M. Montali, and I. Teinemaa. "Semantics and Analysis of DMN Decision Tables". In: *Proc. of the 14th Int. Conf. on Business Process Management (BPM)*. Vol. 9850. Lecture Notes in Computer Science. Springer, 2016, pp. 217–233.

[5] ___, M. Dumas, Ü. Laurson, F. M. Maggi, M. Montali, and I. Teinemaa. "Semantics, Analysis and Simplification of DMN Decision Tables". In: *Information Systems* 78 (2018), pp. 112–125.

[6] M. Ortiz, M. Simkus, and T. Eiter. "Worst-case Optimal Conjunctive Query Answering for an Expressive Description Logic without Inverses". In: *Proc. of the 23rd AAAI Conf. on Artificial Intelligence (AAAI)*. AAAI Press, 2008, pp. 504–510.

References

References II

[7] ___, M. Montali, M. Dumas, and F. M. Maggi. "Semantic DMN: Formalizing and Reasoning About Decisions in the Presence of Background Knowledge". In: *Theory and Practice of Logic Programming* 19.4 (2019), pp. 536–573.