CONSTRUCTION AND REPRESENTATION OF CONCEPTS IN THE FRAME-BASED LANGUAGE OBJLOG+ : FROM PROBABILISTIC CONCEPTS TO PROTOTYPES

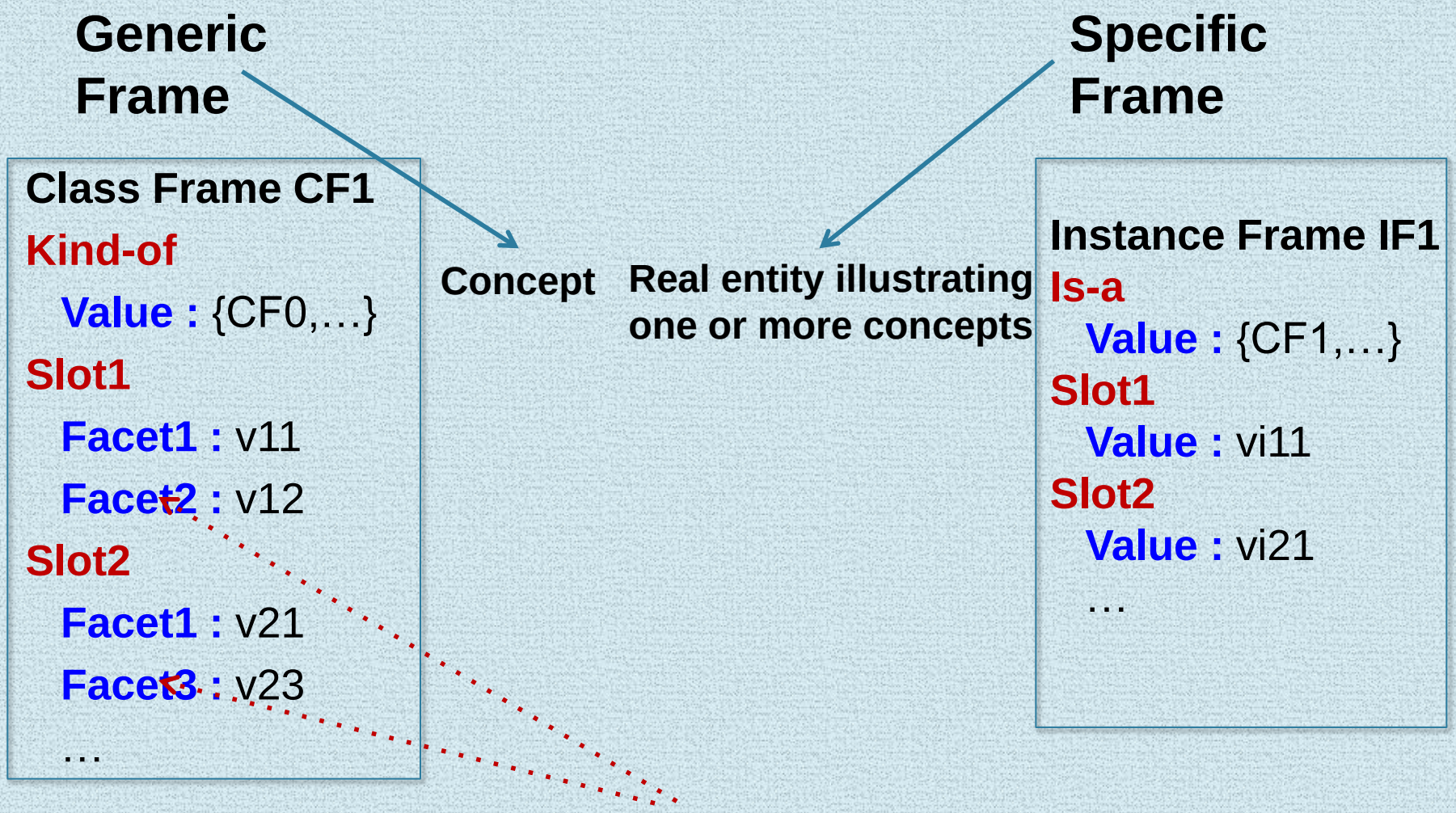
Colette Faucher

LSIS, UMR CNRS 7296

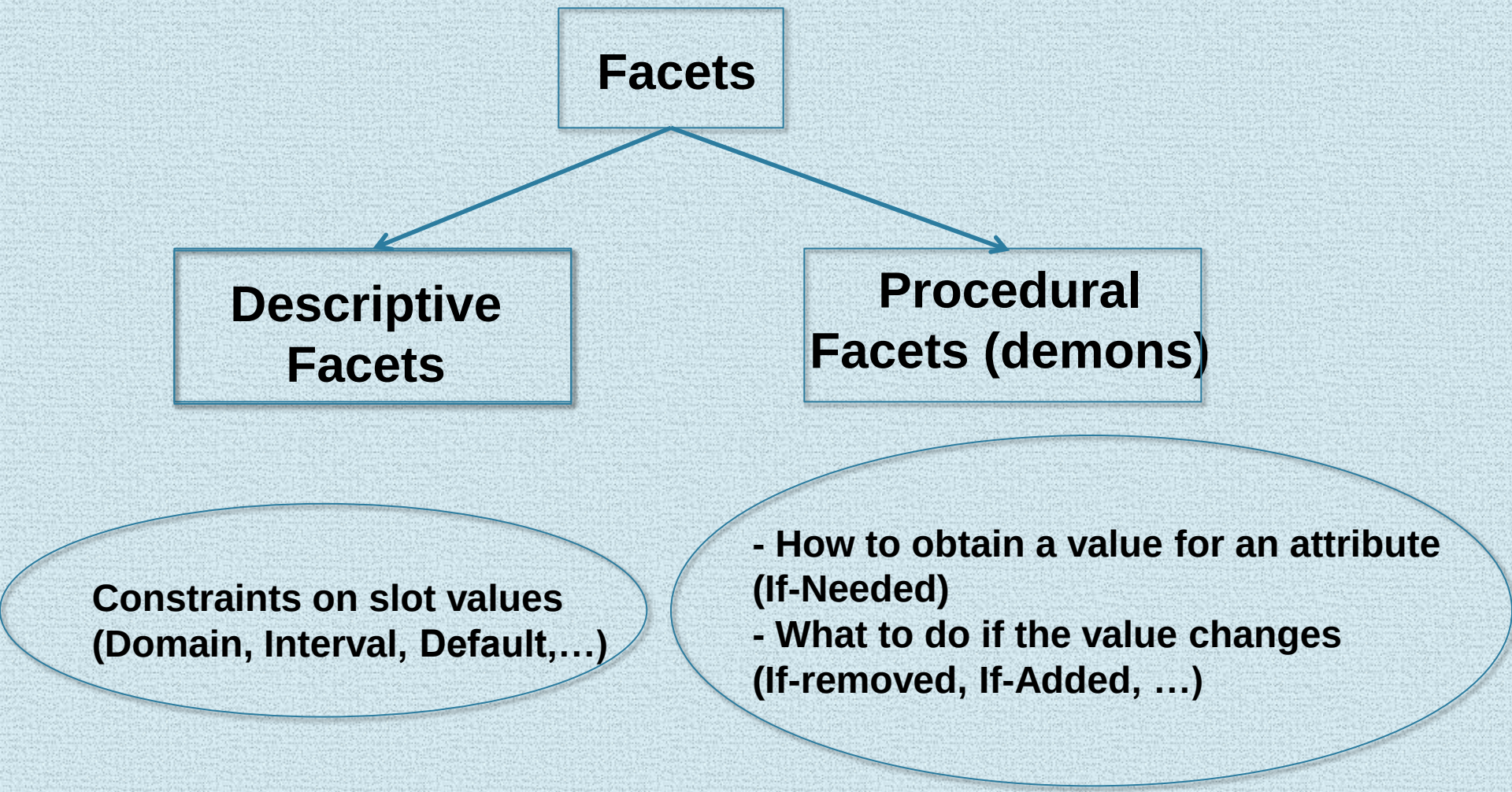
Polytech ’Marseille, FRANCE

Contents

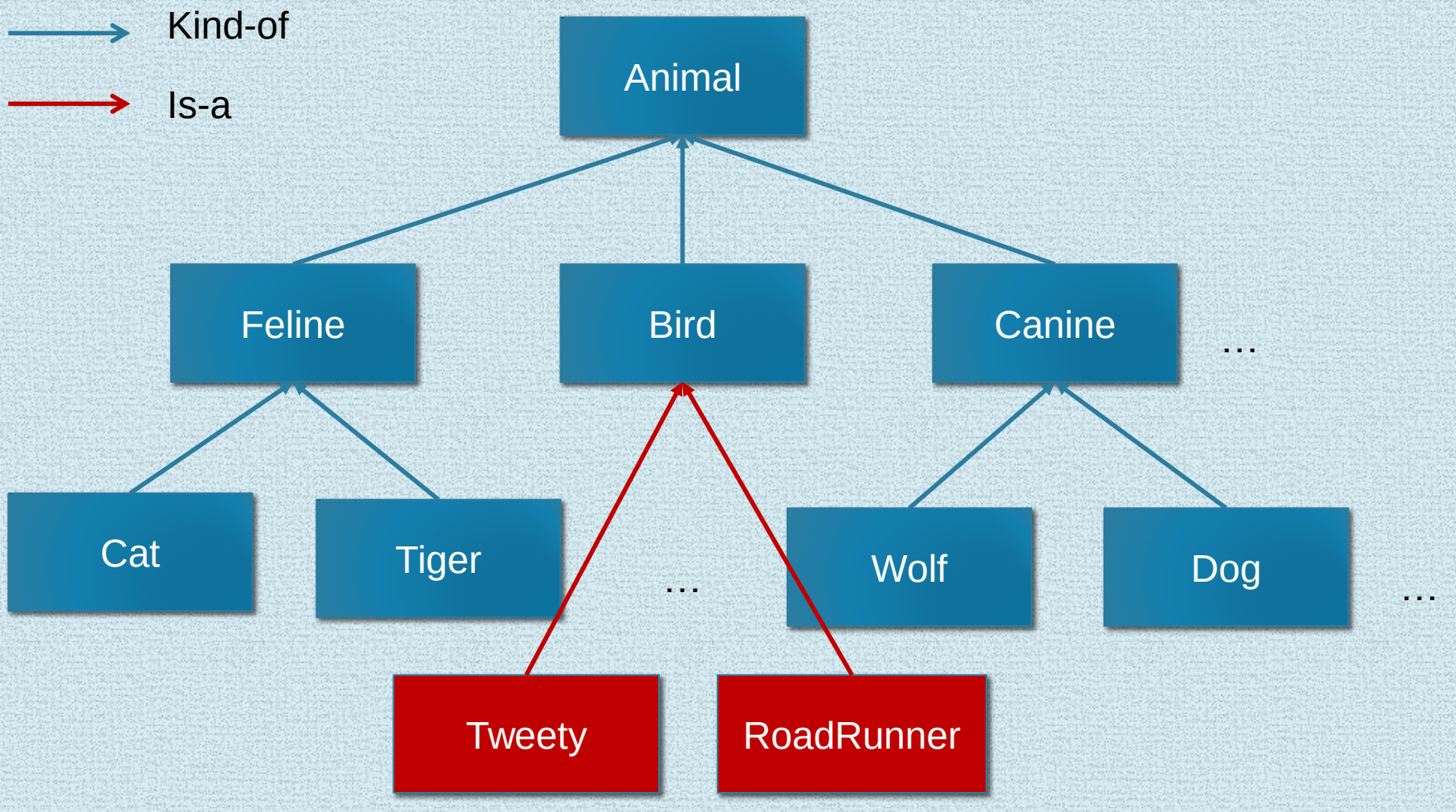
Frame-based representation (1)



Frame-based Representation (2)



Frame-based representation (3)



Hierarchy of inheritance - Instanciation

Frame-based representation (4)

Example :

Class Frame Animal

Kind-of

Value : {Animated-being}

BodyCover

Domain : {feathers, fur, smooth-coverage}

Locomotion

Domain : {walking, running, flying, crawling,…}

Color

Domain : {yellow, blue, multi-colored, …}

Age

Domain : Integer

Class Frame Bird

Kind-of

Value : {Animal}

BodyCover

Value : feathers

Locomotion

Default : flying

Color

Domain : {yellow, blue, multi-colored, …}

Age

Domain : integer

Singing

Domain : {yes, no}

Frame-based representation (5)

Example :

Instance Frame Tweety

Is-a

Value : {Bird}

BodyCover

Value : feathers

Locomotion

Value : flying

Color

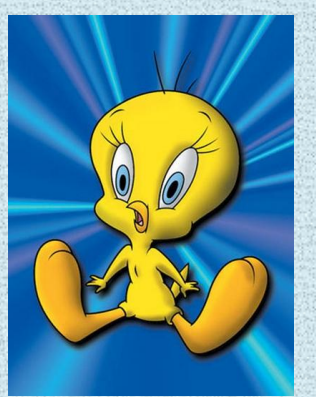
Value : yellow

Age

Value : 3

Singing

Value : yes



Instance Frame RoadRunner

Is-a

Value : {Bird}

BodyCover

Value : feathers

Locomotion

Value : running

Color

Value : grey-and-blue

Age

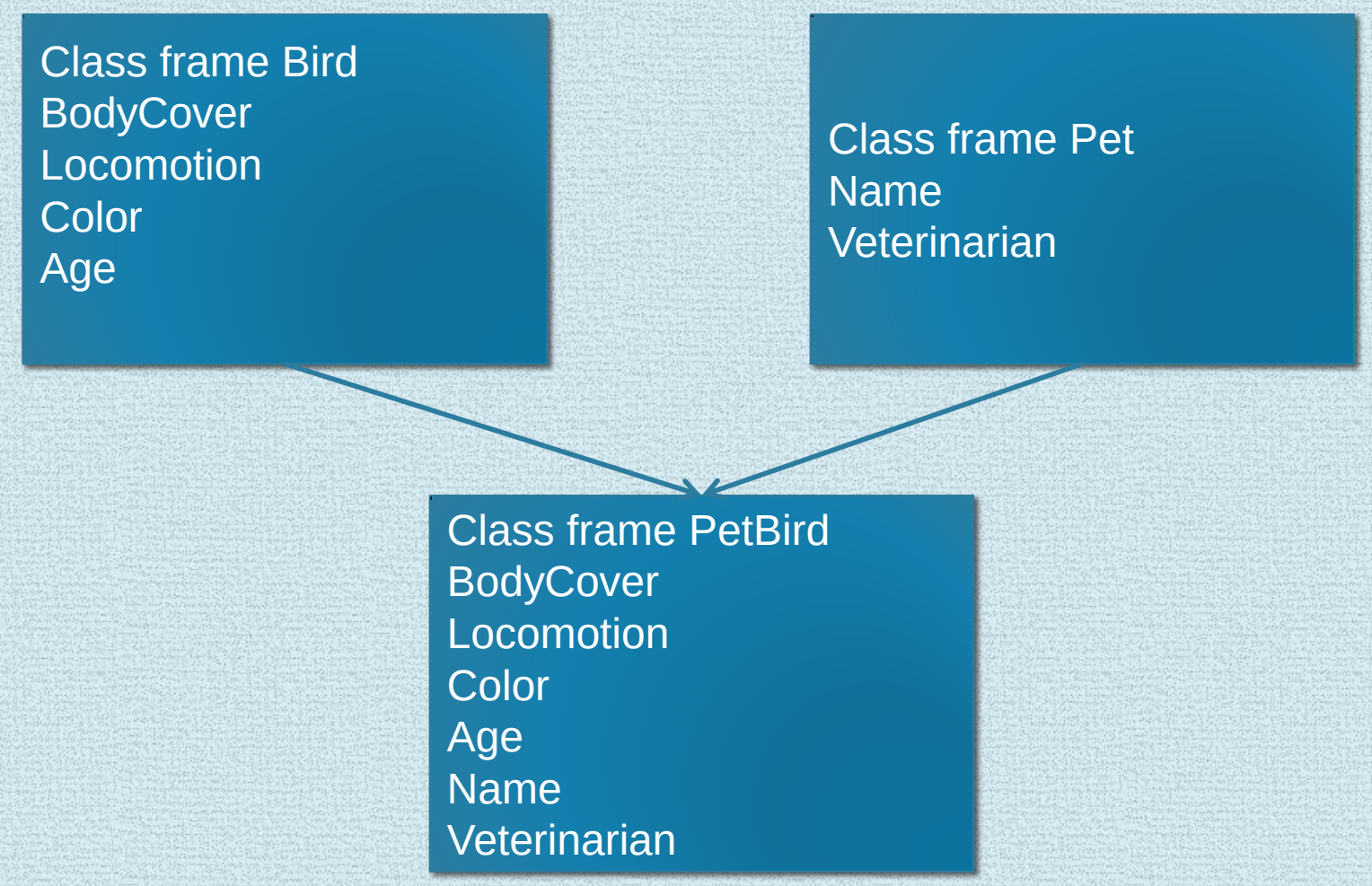
Value : 5

Singing

Value : no

Frame-based representation (6)

Multiple-inheritance



Modeling problem

Modeling a piece of knowledge in a Frame-Based representation

↓

It must be either :
- A concept or
- An instance of one or more concepts.

↓

An observation a liitle or incompletely known, whose membership concept(s) are not yet known cannot be stored in the framework of a frame-based representation.

Need to represent observations of little or incompletely known real entities and to have a method to build concepts from them and then to change their status to instances of these concepts, within a frame-based representation.

Frame-based Representation (8)

Notion of Goal of categorization

The veterinarian

The Mum

What are the relevant characteristics of an animal for them ?

The veterinarian	The Mum
Age	Age
Number of heart chambers	Price
Type of breathing	Beauty
Locomotion	Obedience
Type of vision	Kindness
…	…

When they see animals, they won't categorize them in the same way, different categories will be built.

What do we need ?

1) **A frame-based language that would allow the representation of observations of the real world without knowing to which concepts these observations are linked.**

=> the frame-based language OBJLOG+

2) **A method that generates concepts and instances linked to these concepts from observations of real entities and that takes into account the importance of the observations' properties according to different goals of categorization to generate multiple hierarchies, each one corresponding to a given perspective.**

=> CONFORT, a concept formation system that generates multiple hierarchies of class frames corresponding to different goals of categorization.

OBJLOG+ characteristics

Objlog+ : frame-based language built on top of Prolog, extensible and auto-referent. Its extensibility is due to the following characteristics :

1) All the basic elements are reified (auto-reference) : slots, facets, methods, messages, etc.

1) A new acceptation of the notion of frame that does not assume that a frame has a predefined semantics, being either a class frame or an instance frame.

2) A method has been defined in order to allow the creation of new facets the control structure of which is automatically managed by the system.

We will focus on the second feature in this context.

What's a frame in OBJLOG+

In classical frame-based languages, frames' semantics is implicit :

- If there's a **Kind-of** slot in the frame => it describes a concept.
- If there's a **Is-a** slot in the frame => it describes a concept instance.

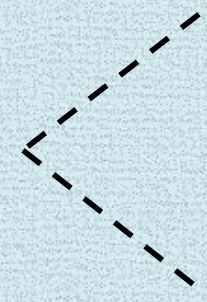
Frame in OBJLOG+ = **three-leveled data structure, slot/facet/value with no attached implicit semantics.**

Frame semantics is defined a posteriori and explicitly.

What's a frame in OBJLOG+ ? Categories of frame

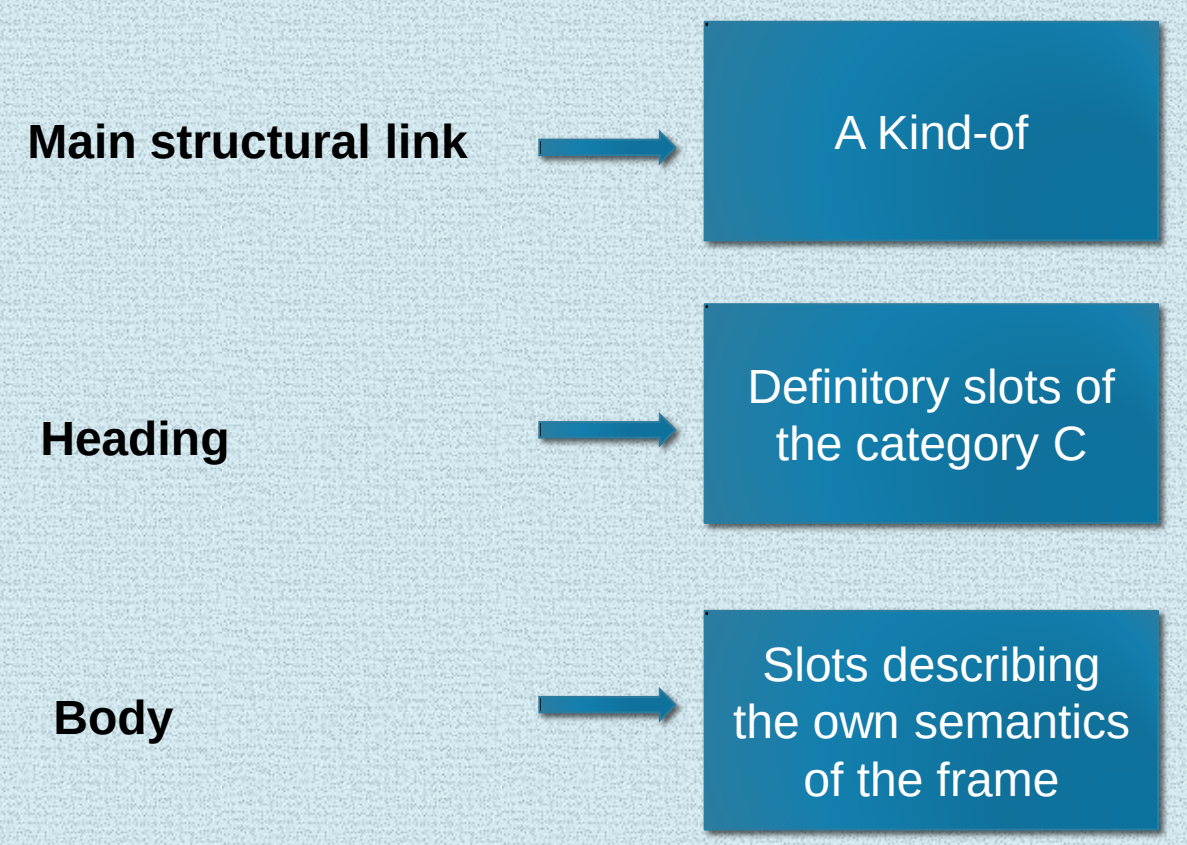
Frame defining a category C of frames with a common semantics

CategoryC
Kind-of
Value : FRAME
SlotC1…
SlotC2…
…
SlotCn
Global consistency
Value : GConsC
Local consistency
Value : LConsC
Methods
Value : Meth1C, Meth2C…



Definitory Slots

General description of a frame of category C



Frame organization in OBJLOG+

Notion of structural link

- Defined within a frame representing a category.
- Characteristic property :

Let L be a structural link :

F_1 **F_2**

Slot S **Slot L**

… **Value :** F_1

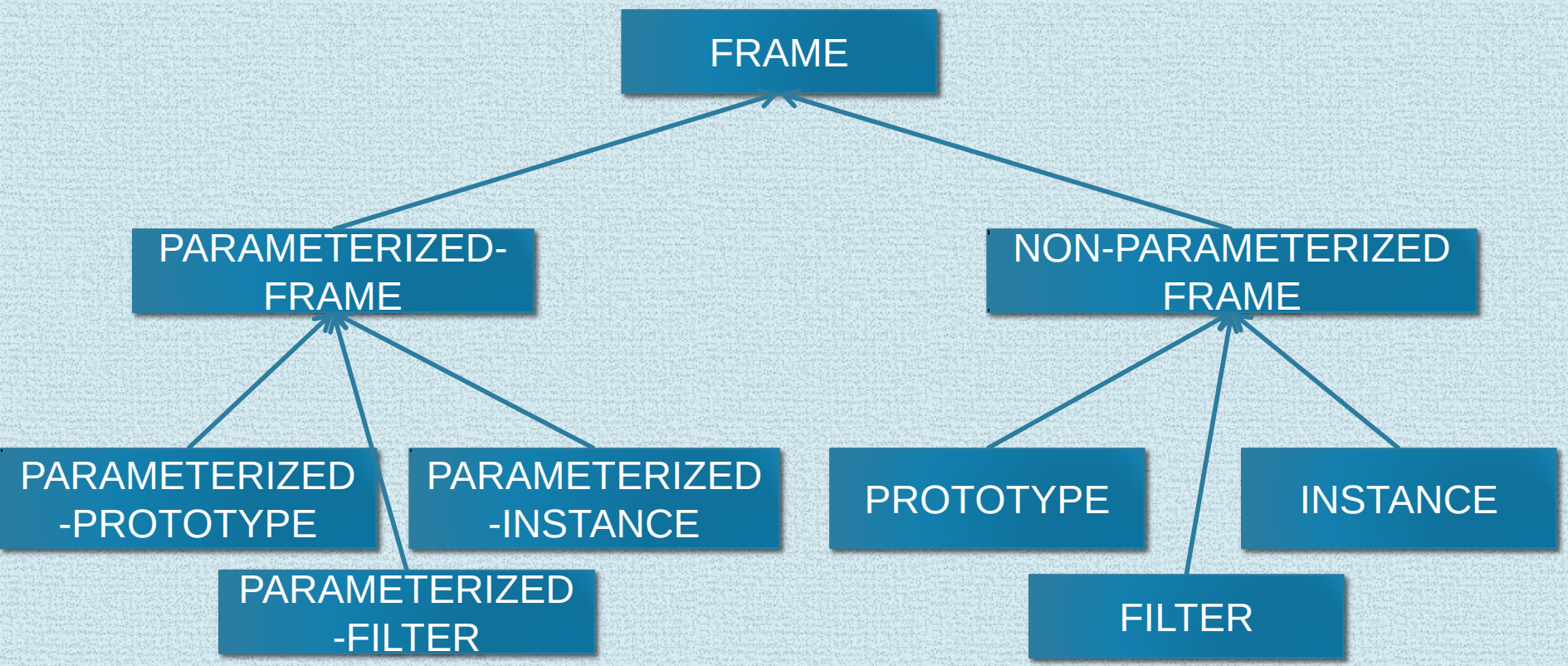
Slot S inheritable through structural link L => S is inherited in F2.

A slot can be :

- **Not inheritable**,
- **Inheritable through one or several structural links**.

Main structural link in Objlog+ is **Kind-of**, underlies the complete hierarchy of the frames of the language.

Core of OBJLOG+



Parameterized frames => new form of genericity .

Models for non-parameterized frames of the same category that differ from one another only concerning the values of some facets of their slots, the *parameters* of the parameterized frame.

Back to the problem : representing Observations

An observation :

- Represented by a frame in OBJLOG+ acceptation, without semantics,
- Sub-frame of the frame OBSERVATION
- The frame OBSERVATION has no definitory slots.

Global consistency : A frame representing an observation is directly attached to the frame OBSERVATION by means of the link Kind-of. The values of such a frame are local.

Local consistency : All the slots of an OBSERVATION frame are non inheritable.

Back to the problem : representing Observations

Basic method :

From a set of observations :

- Building hierarchies of concepts (probabilistic concepts in a first step, prototypes in a second step)
- Observations change their status to examples of the probabilistic concepts, then to instances of the generated prototypes.

=> That’s what is done by CONFORT

Main characteristics

CONFORT (CONcept Formation in Object RepresenTation)

- **Knowledge Acquisition tool** for helping an expert in his activity of elaborating and representing concepts of his domain from observations. The expert can interact with the system.

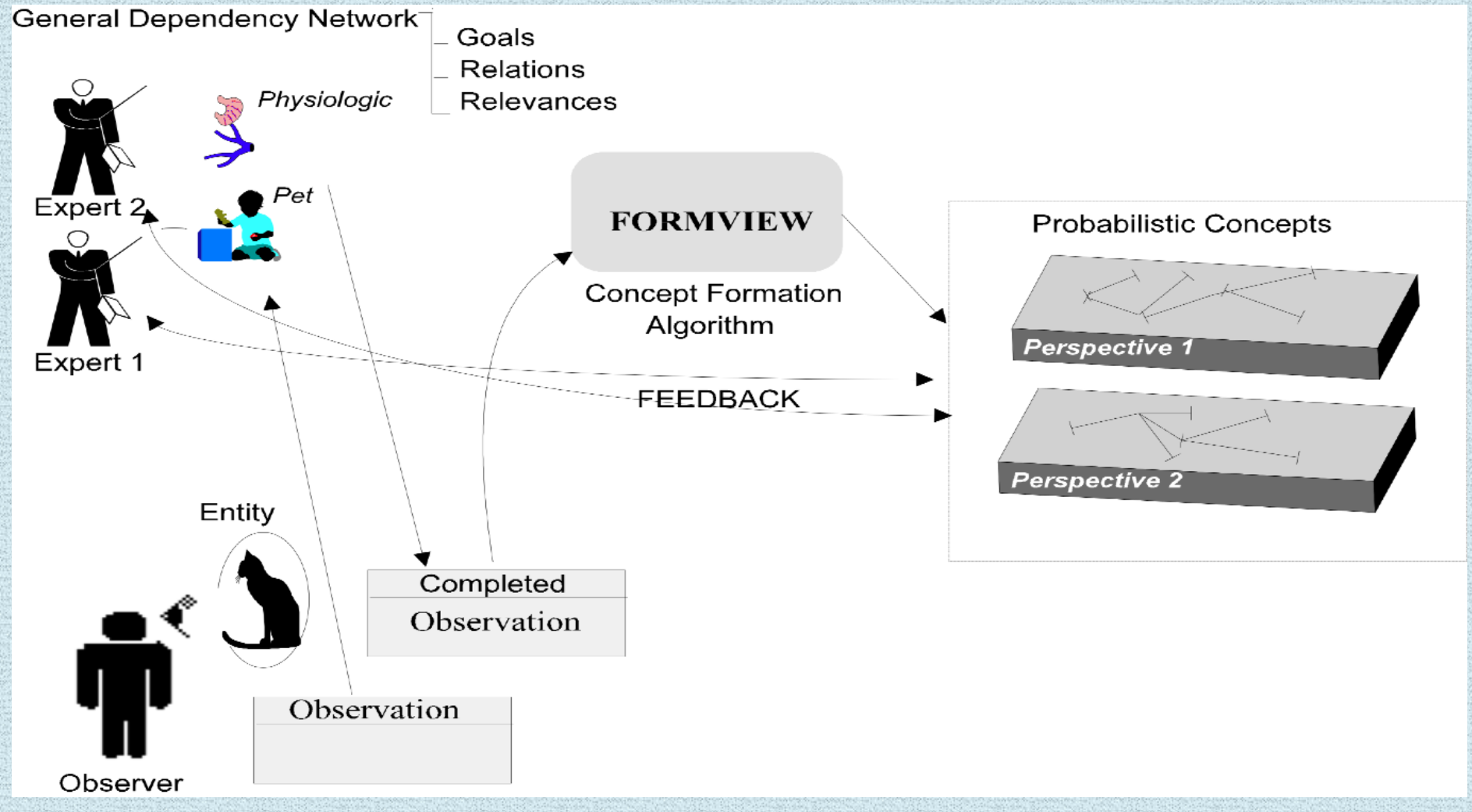
- Makes use of **machine learning and cognitive psychology ideas** concerning concept formation and categorization.

- According to cognitive psychological studies, it's **based on the assumption that categorization is a goal-driven process.**

=> Generation of **several probabilistic concept hierarchies**, each one representing and organizing concepts from observations **according to different perspectives** corresponding to different experts' categorization goals or opinions.

=> Generation of the **corresponding prototype hierarchies**.

CONFORT(2)


General Dependency Network
Goals
Relations
Relevances
Physiologic
Pet
Expert 2
Expert 1
FORMVIEW
Concept Formation Algorithm
Probabilistic Concepts
Perspective 1
Perspective 2
FEEDBACK
Entity
Completed
Observation
Observation
Observer

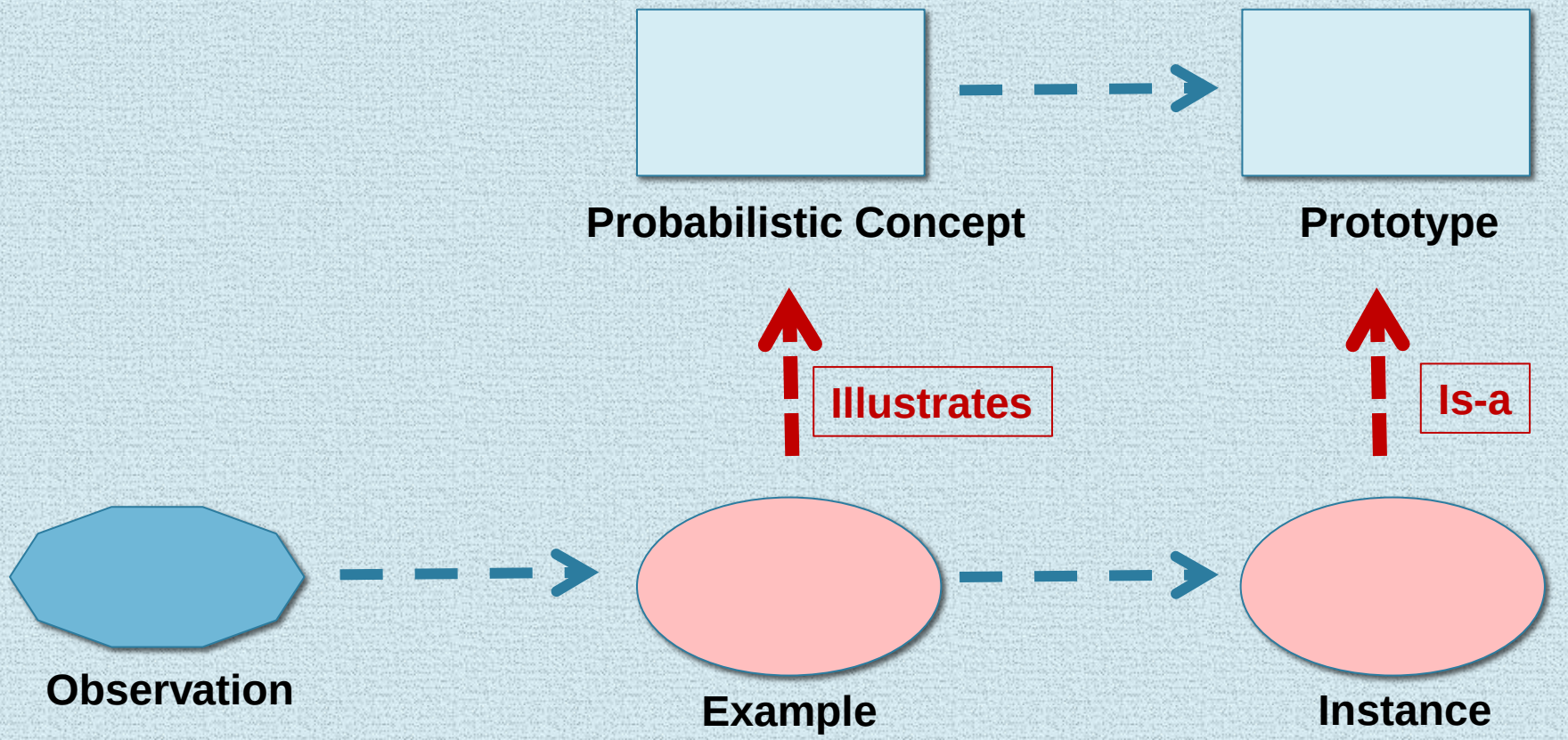
CONFORT (3)

Core of CONFORT : **FORMVIEW**, a learning algorithm of incremental concept formation.

We focus on :

- **FORMVIEW** that constructs **multiple hierarchies of probabilistic concepts** (probabilistic concept trees).
- The **generation of frame hierarchies** from probabilistic concept hierarchies.

CONFORT steps



What's a probabilistic concept C in CONFORT ?
(extension of the definition by Smith and Medin)

A conjunction of tuples defined by :
(At, v_{At}, PDvAt, PPvAt), where :

- At is an attribute from a set of attributes A,

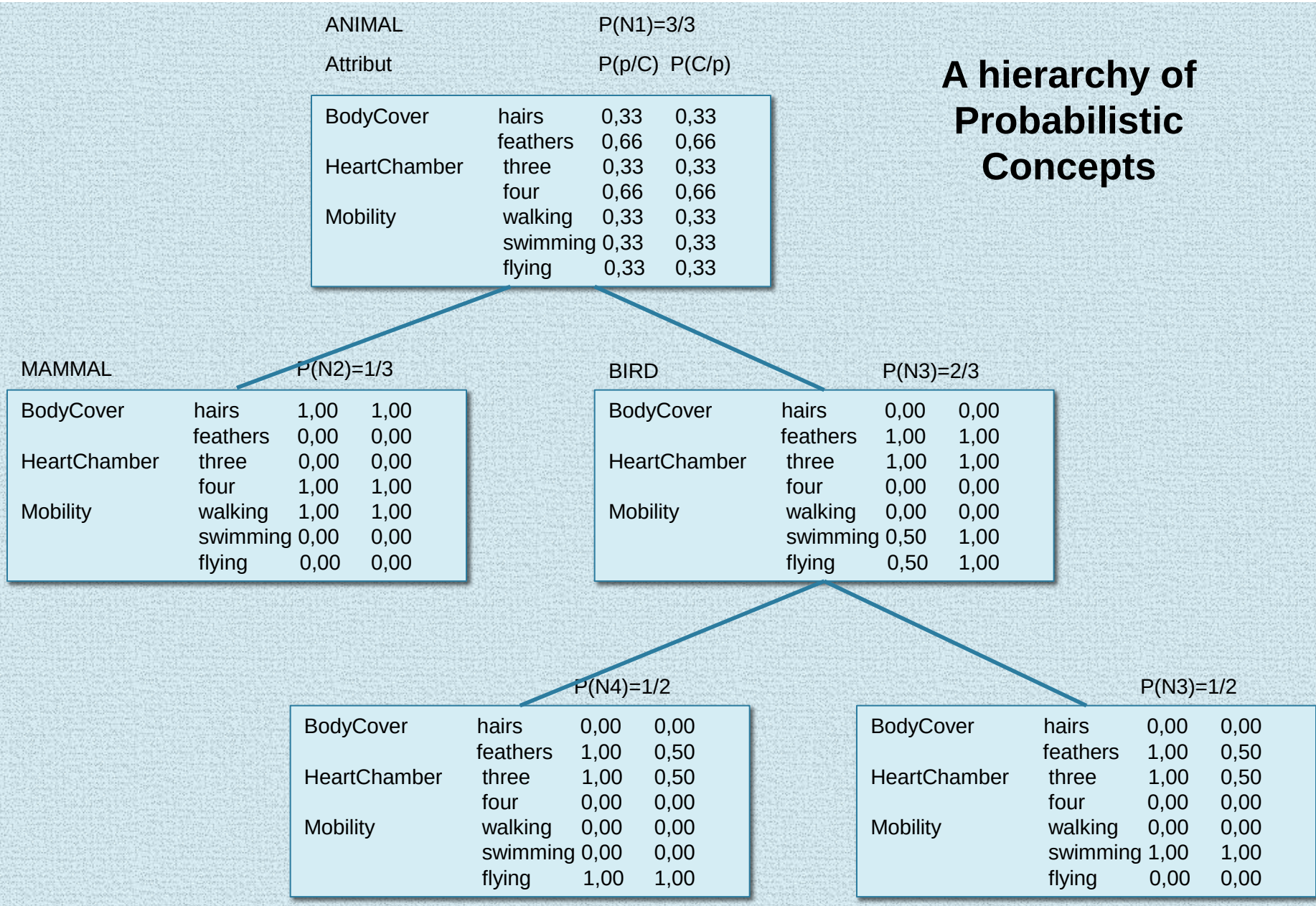
- v_{At} belongs to the set of values of the attribute At, (At)

- **PD_{vAt}** is the value of the conditional probability **$P(At=v_{At}|C)$** (***predictability***) for each value v_{At} from (At),

- **PP_{vAt}** is the value of the conditional probability **$P(C|At=v_{At})$** (***prediction power***) for each value v_{At} from (At).

CONFORT(6)

A hierarchy of Probabilistic Concepts

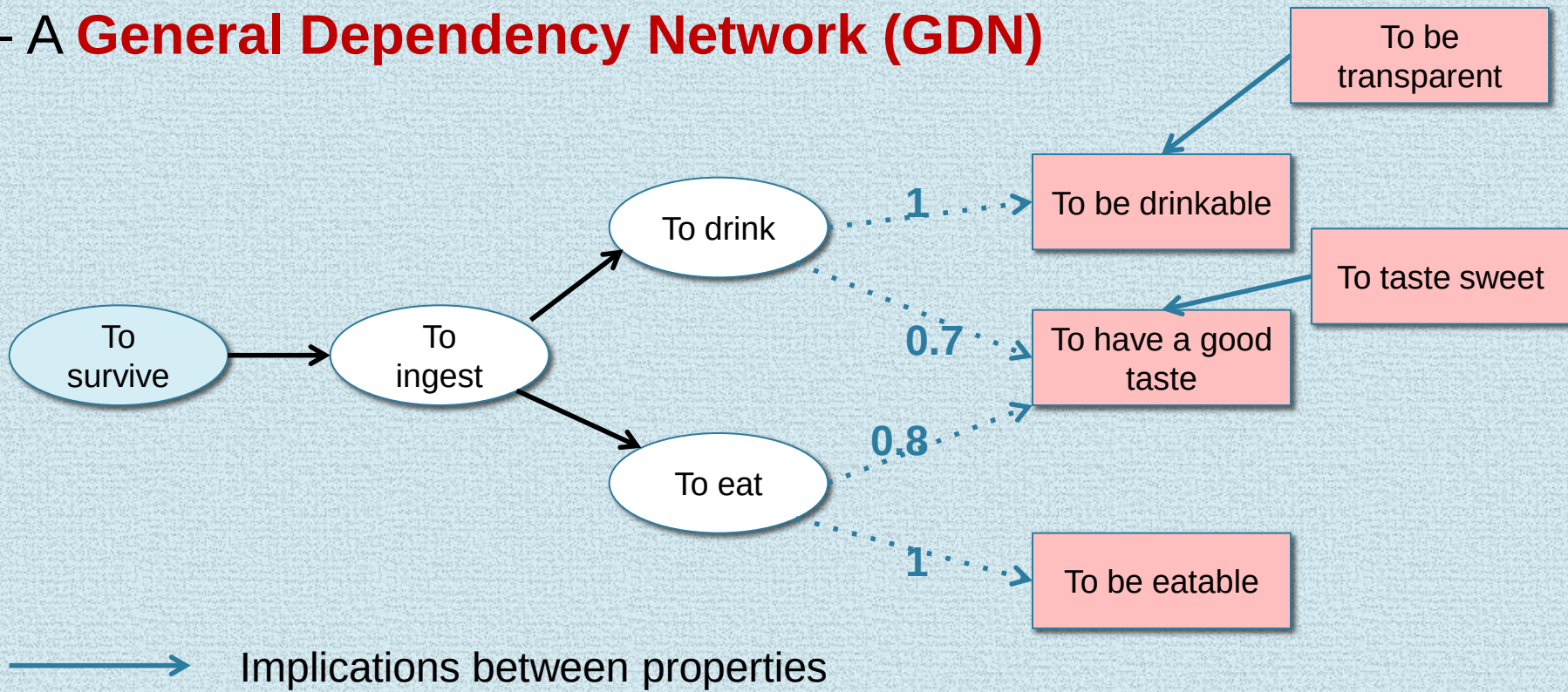


FORMVIEW INPUTS

- **Observations**

Described by a set of pairs (attribute, value).

- A **General Dependency Network (GDN)**



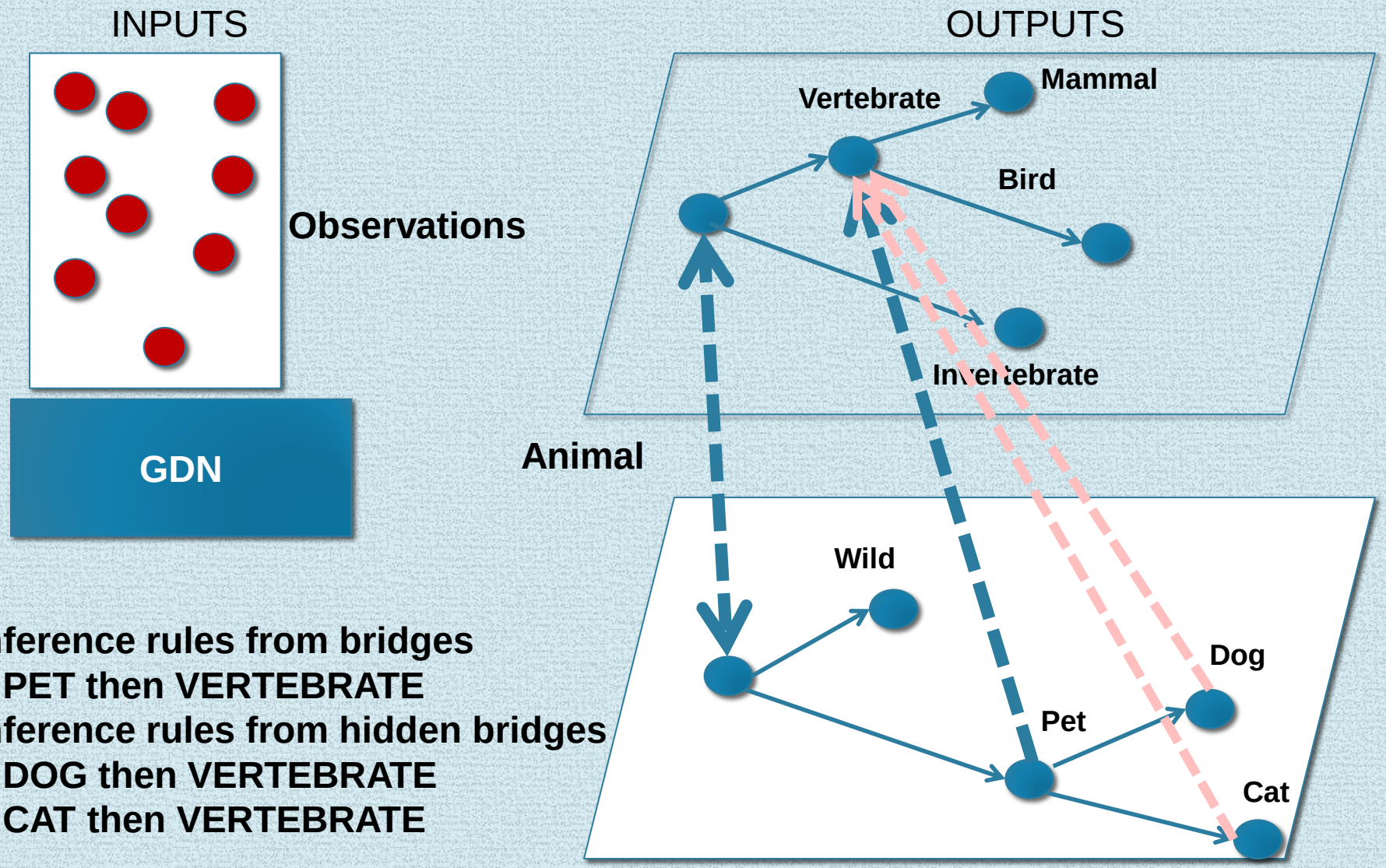
FORMVIEW OUTPUTS

- **Multiple hierarchies of probabilistic concepts** corresponding to different goals of categorization

- **Bridges : communication channels between hierarchies** representing different perspectives
  - If Ext(C1) = Ext(C2) then bridge(C1(p1), C2(p2)) =1
  - If Ext(C1) Ì Ext(C2) then bridge (C1(p1), C2(p2))=0
  - If Ext(C1) Ë Ext(C2) and Ext(C2) Ë Ext(C1) then bridge(C1(p1), C2(p2))=-1

The specialization relation allows FORMVIEW to establish **hidden bridges** between children of a bridge's source node and a bridge's target node.

PROBABILISTIC CONCEPTS HIERARCHIES IN FORMVIEW



Inference rules from bridges
If PET then VERTEBRATE
Inference rules from hidden bridges
If DOG then VERTEBRATE
If CAT then VERTEBRATE

FORMVIEW algorithm

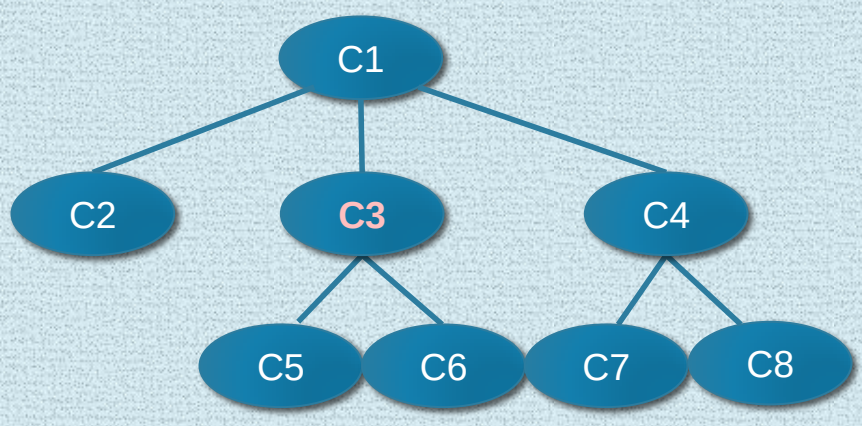
Inspired from classical incremental concept-formation algorithms that recognize regularities among a set of non-preclassified entities (observations) and induce a concept hierarchy that organizes these observations (for example COBWEB, Fisher).

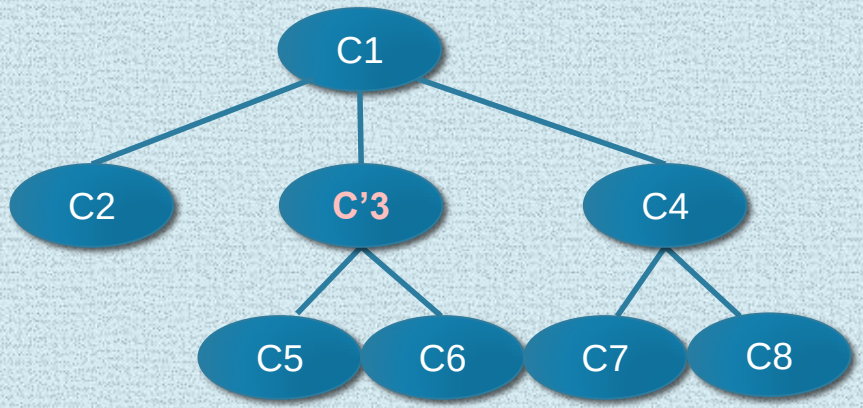
FORMVIEW uses 4 usual operators for building the probabilistic concept hierarchies it generates :

- **Incorporating an observation** into an existing node,
- **Creating of a new node** representing an observation,
- **Splitting a node**,
- **Merging two nodes**.

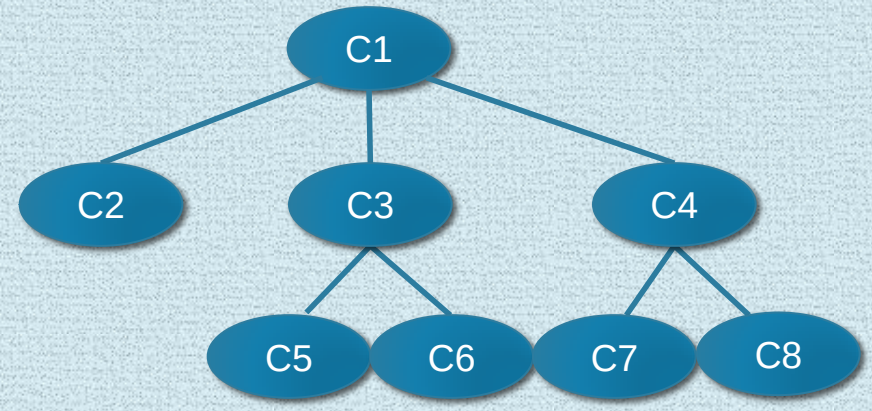
The choice of the operator to apply at each step is determined by means of a quality measure for concepts, the **category utility**.

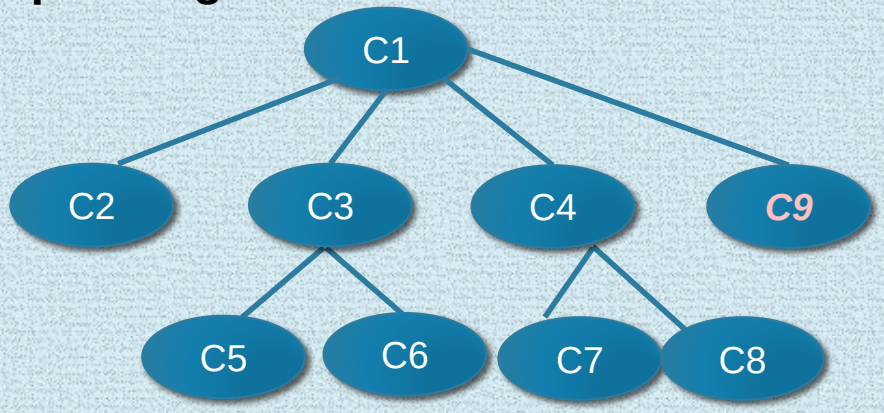
Operators for structuring a hierarchy





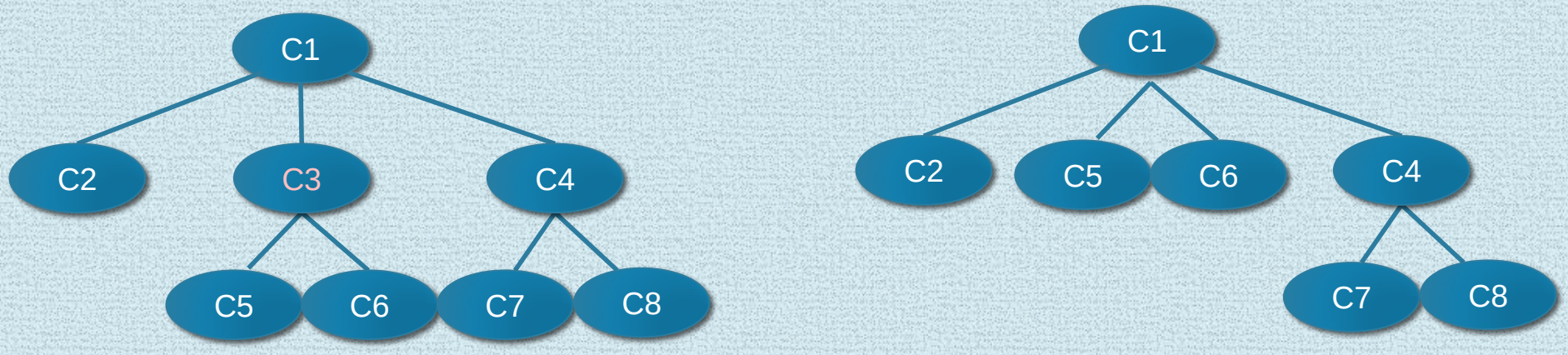
Incorporating



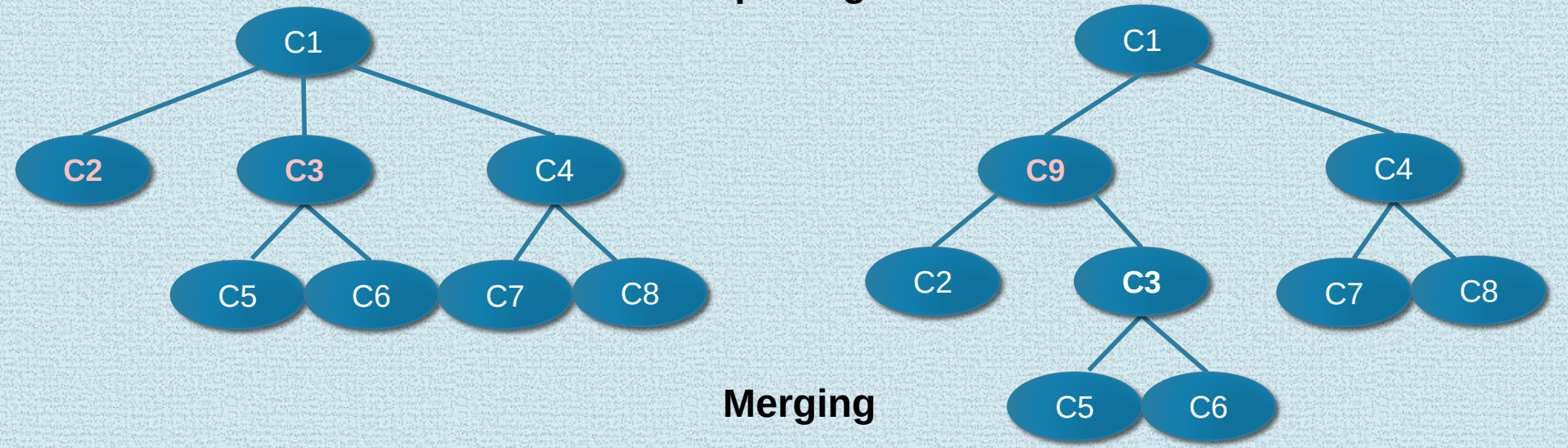


Creating

Operators for structuring the hierarchy (2)



Splitting



Merging

Classical concept formation algorithm (COBWEB)

FUNCTION COBWEB (Object, Root {of a classification tree})

Update counts of the Root

IF Root is a leaf

THEN Return the expanded leaf to accommodate the new object.

ELSE

1) For each direct sub-concept of Root,

calculate the utility of the sub-concept if O is incorporated into it.

Let C_{best1} and C_{best2} be the two concepts with the best utilities.

Let P_i be the partition obtained in incorporating O in C_{best1}.

- Create virtually a new sub-concept of Root including O. Let P_c be the resulting partition.
- Merge virtually C_{best1} and C_{best2}. Let P_m be the resulting partition.
- Split virtually C_{best1}. Let P_s be the resulting partition.

2) Calculate the utility of each partition and choose the one with the best utility and apply the corresponding operator.

IF the best partition is P_i, **THEN** call COBWEB(O, C_{best1})

IF it's P_m **THEN** call COBWEB(O, Merged Node)

IF it's P_s **THEN** call COBWEB(O, Root)

FORMVIEW Algorithm

1) From the first observation O_1 (or possibly several observations describing the same entity), calculate the complete observations for each perspective t, OC_{1t} from the GDN.
2) Create the roots $Root_t$ of each hierarchy corresponding to a perspective by means of the first completed observation, OC_{1t}.
3) For each perspective t :

   For each observation that follows O_1, O_k :

i) Calculate the complete observation for perspective t : OC_{kt}.

ii) **IF** OC_{kt} hasn't yet been categorized by means of a bridge, **THEN FORMVIEW-Incremental-Categorization ($Root_t$, OC_{kt})**.

FORMVIEW-Incremental-Categorization is analogous to COBWEB, except that the partitions **P_i, P_c, P_m and P_s include the concepts of the other perspectives that are linked to a concept of the initial partition by a bidirectional bridge.**

Moreover, **each time an observation is incorporated into a concept, the bridges from this concept are (re-)computed.**

Understanding the Category Utility : the Basic Level (Rosh)

What's this ?

Pr(Animal) < Pr(Spaniel) < Pr(Dog)

Probabilty of the answer

Is this a dog, an animal, a spaniel ?

T(Dog) < T(Animal) < T(Spaniel)

Time of response

GLUCK and CORTER's interpretation of basic level categories

« Basic level categories are the ones for which inferences made by human beings are the most numerous. »

The ***Category Utility*** (Gluck and Corter) :

- allows to discover the basic level within a category hierarchy.
- measures the capacity, for a given category, to predict the values of the attributes of the members of this category, its « **prediction power** ».
- can be described as a trade-off between the expected number of features that can be correctly predicted about a member of a category C, and the proportion of the environment P(C) to which those predictions apply.

GLUCK and CORTER's interpretation of basic level categories

Highly general category (e.g. animals) :

Few properties predicted (e.g. animate) + for a **large population**.

Highly specific categories (e.g. robins) :

Many properties predicted + **small population**

Basic level categories (e.g. birds) :

Maximises the trade-off between the expected number of accurate predictions and the scope of their application.

Category utility =

increase in the expected number of properties that can be correctly predicted given the knowledge of a category, $(P(pi|C)^2)$, over the expected number of correct predictions without such knowledge, $(P(pi)^2)$.

Formal expression of Category Utility

Let E be a set of entities defined by observations,H a hierarchy built on E, A the set of attributes describing the observations, PC a probabilistic concept described by A covering a category C

PC = {(p_i, $p(p_i/C)$, $P(C, p_i)$), 0<i<card(V(a)), a in A}

with $p_i=(a,v_i)$

The **utility of category C** is defined by (Gluck and Corter) :

$$UC(C) = P(C)\sum_{i=1}^{k}\left(P(p_i/C)^2 - P(p_i)^2\right)$$

By Bayes' formula :

$$UC(C) = \sum_{i=1}^{k} P(p_i)\left(P(C/p_i)P(p_i/C) - P(C)P(p_i)^2\right)$$

Formal expression of category utility

The factor $P(p_i)$ in the formula by Gluck and Corter is replaced in FORMVIEW by a factor that takes into account the semantic relevancy expressed for the property in the GDN.

$$UC(C) = \sum_{i=1}^{k} \mathrm{D}p_i \left(P(C / p_i) P(p_i / C) - P(C) P(p_i) \right)$$

$$\mathrm{D}(p_i) = relevance(p_i) + P(p_i)$$

Utility of a partition

In COBWEB (Fisher) :

Mean of the utilities of the categories of the partition.

In FORMVIEW :

Same computation, but the categories of other perspectives that are linked by means of bridges to categories of the partition are added.

The predictive power of a category in a hierarchy is measured:

- By means of the properties that can be predicted from this category in a hierarchy and
- By means of properties that can be predicted from categories of other perspectives, linked to the initial category by bridges.

FROM PROBABILISTIC CONCEPTS TO PROTOTYPES

Why generating a representation by prototypes ?

Probabilistic concepts : storage of the probabilities of appearance of valued properties and concepts.

Prototypes : Expression in a symbolic way and in intension of the semantics conveyed by the probabilistic concepts

=> more abstract and intelligible representation for the user.

The generation is made in two steps :

- Vertical dimension : definition of the hierarchical organization of the prototypes from the one of the probabilistic concepts.

- Horizontal dimension : definition of the composition of the prototypes, that is the properties that will constitute their description.

FROM PROBABILISTIC CONCEPTS TO PROTOTYPES

Goal : Searching for concepts without a large importance in terms of predictive power and ruling them out for the transformation into prototypes.

What's a concept with a good predictive power ?

When an observation is categorized into a concept, it's good if it allows to discover unknown properties of the observation.

Strategy :

- Hiding the value of a given property in the observation to be classified,
- after its categorization, comparing the hidden value with the most frequent value (the one with the highest predictability) of the attribute in the hosting category.

The frequency of good predictions for each property is stored and updated in the category, each time FORMVIEW incorporates an observation in it (recursive process).

Quality point for an attribute : the counter of the attribute in this concept is higher than the one in all its sub-concepts (historically the concept has permitted the highest number of predictions).

Quality concept = quality point for all its attributes.

Sub-concepts of a quality concept are not transformed into prototypes.

The horizontal dimension

Goal : defining the facets of the slots of the prototypes that correspond to the attributes of the remaining concepts. Some properties of the concepts may not appear through prototype slots due to the inheritance mechanism.

Facet definition

Domain : The set of values of the attribute in the concept.

⇒factorization and specialization

A method has been defined to generated domain for referential attributes.

Default : value with the highest predictability (P(a=v/C) superior to a threshold defined by the user (>0.5). Maximum one value.

Exception : value with the lowest predictability (P(a=v/C) inferior to a threshold defined by the user.

The horizontal dimension

New specific slots of the category Prototype :

- **Sufficient properties :** predictive power =1 (P(C/p) =1)

- **Necessary properties :** predictability =1 (P(p/C)=1)

- **Property correlation**

Important in cognitive psychology : correlations of properties play an important role when evaluating the typicality of an entity with regards to a category (Malt), allow to avoid compensations between elementary typicalities.

Example :

Small birds sing. Big birds do not sing.

Property (size, small) is more typical than (size, big)

Property (expression, singing) is more typical than (expression, cawing)

A small singing bird is more typical than a big bird cawing.

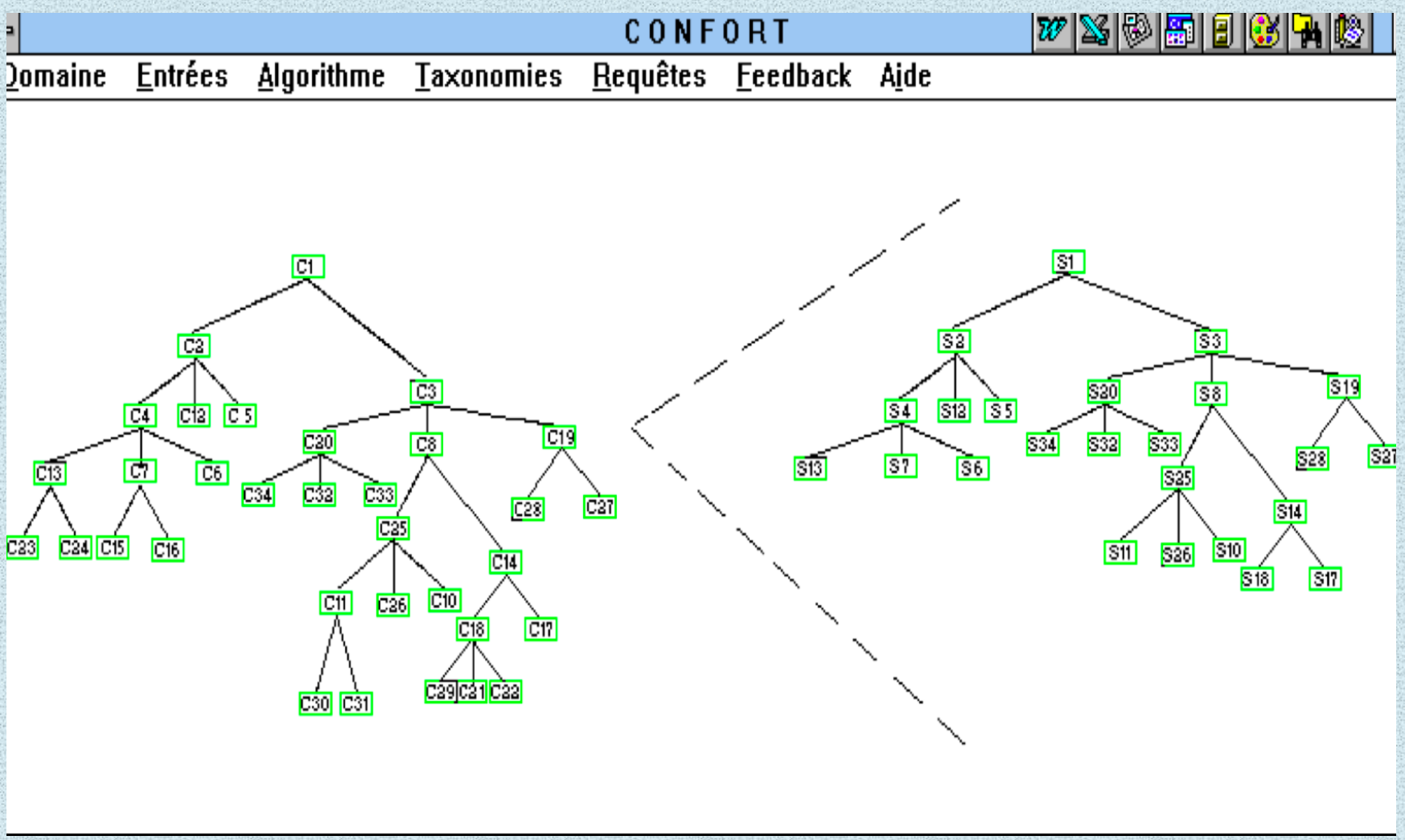
Without taking into account property correlation :

A big singing bird is more typical than a big bird cawing, which is wrong.

The taking into account of property correlation allows to correct global typicality.

Strategy : For each concept, cases of total correlation between two properties p and p' (P(p/p') and P(p'/p)=1)

From Probabilistic Concepts to Prototypes : example



From Probabilistic Concepts to Prototypes : example

C1
Kind-of
Value : {PROBABILISTIC-CONCEPT}
Correspondence
Value : S1
ProbConcept
Value : (20/20)
Habitat
ProbDomain : {(inner, 7/20), (outer,13/20)}
Food
ProbDomain : {(fresh, 13/20), (canned,7/20)}
Predator
ProbDomain : {(yes, 4/20), (no, 16/20)}
Venomous
ProbDomain : {(no, 1)}
Appearance
ProbDomain : {(normal, 16/20), (nice, 4/20)}
Origin
ProbDomain : {(intern, 15/20), (extern, 5/20)}
Owner
ProbDomain : {Owner1, 16/20), (Owner2, 4/20)}
Age
ProbDomain : {(young, 1)}
Intelligence
ProbDomain : {(high, 3/20), (average, 8/20), (low, 9/20)}
Price
ProbDomain : {(high, 8/20), (average, 8/20), (low, 4/20))

S1
Kind-of
Value : {PROTOTYPE}
Necessary
Value : {(venomous, no), (age, young)}
Sufficient
Value : {(venomous, no), (age, young)}
Habitat
Domain : (inner, outer)
Food
Domain : {fresh, canned}
Predator
Domain : {yes, no}
Venomous
Domain : {no}
Appearance
Domain : {(normal, nice)}
Origin
Domain : {(interne, externe)}
Owner
Domain : {Owner1,Owner2}
Age
Domain : {young}
Intelligence
Domain : {high, average, low}
Price
Domain : {high, average, low}

From Probabilistic Concepts to Prototypes : example

C3
Kind-of
  Value : C1
ConceptProb
  Value : 13/20
Correspondence
  Value : S3
Habitat
  ProbDomain : {(inner, 0),(outer,1)}
Predator
  ProbDomain : {(yes, 4/13), (no, 9/13)}
Venomous
  ProbDomain : {(no,1)}
Appearance
  ProbDomain : {(normal, 10/13),(nice, 3/13)}
Origin
  ProbDomain : {(intern, 9/13), (extern,4/13)}
Owner
  ProbDomain : {(Owner1, 10/13), (Owner2, 3/13)}
Food
  ProbDomain : {(fresh, 1)}
Age
  ProbDomain : {(young, 1)}
Intelligence
  ProbDomain : {(high, 3/13), (average, 5/13), (low, 5/13)}
Price
  ProbDomain : {(high, 7/13), (average, 4/13), (low, 2/13))

S3
Kind-of
  Value : P1
Necessary
  Value : {(habitat,outer), (food, fresh), (venomous, no), (age, young)}
Sufficient
  Value : {(habitat, outer), (food, fresh)}
Habitat
  Value : outer
Food
  Value : fresh
Appearance
  ProbDefault : (normal, 10/13)

C2

Kind-of
Value : S1
Kind-of-PC
Value : (C1,7/20)
Correspondence
Value : {S2}
Necessary
Value : {(Habitat, inner), (food, canned), (venomous,no), (age, young), (predator, yes)}
Sufficient
Value : {(Habitat, inner), (food, canned)}
Habitat
ProbDomain : {(inner,1),(outer, 0)}
Food
ProbDomain : {(fresh, 0), (canned,1)}
Predator
ProbDomain : {(yes,1),(no, 0)}
Venomous
ProbDomain : {(no,1)}
Appearence
ProbDomain : {(normal, 6/7), (nice, 1/7)}
Origin
ProbDomain : {(intern, 6/7), (extern, 1/7)}
Owner
ProbDomain : {(Owner1, 6/7), (Owner2, 1/7)}
Age
ProbDomain : {(young,1)}
Intelligence
ProbDomain : {(high, 0), (average, 3/7), (low, 4/7)}
Price
ProbDomain : {(high,1/7),(average, 4/7), (low, 2/7))

S2

Kind-of
Value : S1
Necessary
Value : {(habitat, inner), (food, canned),(venomous, no), (age, young), (predator, yes)}
Sufficient
Value : {(habitat, inner), (food, canned)}
Habitat
Value : inner
Food
Valuet : canned
Appearence
ProbDefault : (Normal, 6/7)
Origin
ProbDefault : (intern, 6/7)
Owner
ProbDefault : (Owner1, 6/7)
Intelligence
Domain : {(average, low)}

CONFORT modeling with OBJLOG+ (1)

As seen before, OBJLOG+ allows the creation of news facets.

Some new facets : **ProbDomain**, **Exception**, **ProbDefault**,…

New categories of frames :

- **Real-Entity**
- **Observation**
- **Objective** (to model the GDN)
- **ProbabilisticConcept**
- **Example**

Modifications in the category Prototype :

New slots : **Sufficient** and **Necessary**

PROBABILISTIC-CONCEPT
Kind-of
  Value : FRAME
Bridge
  Domain : {<PROBABILISTIC-CONCEPT, Real, Real, Real, Real>}
ConceptProb
  Domain : Real
  Cardinality : <1,1>
NbObservations
  Domain : Integer
  Cardinality : <1,1>
LstRealEntities
  Domain : {RealEntity}
  Cardinality : <1, infinite>
Perspective
  Domain : String
MergingUtility
  Domain : Real
  Cardinality : <1,1>
  If-needed : CalculateMergingUtility
SplittingUtility
  Domain : Real
  Cardinality : <1,1>
  If-needed : CalculateSplittingUtility
CreatingUtility
  Domain : Real
  Cardinality : <1,1>
  If-needed : CalculateCreatingUtility
UtilityTwoBestPlacing
  Domain : <PROBABILISTIC-CONCEPT, Real>
  Cardinality : <1,1>
  If-needed : CalculateTwoBestUtilitiesPlace
Global Consistency
  Value : ProgGlobalCoherencyPC
LocalConsistency
  Value : ProgLocalConsistencyPC
Methods
  Value : {Merging, Creating, Splitting, Placing, BridgeEstablishing}

ProgGlobalConsistency :

- In a Probabilistic Concept, it must exist at least one slot having the facet ProbDomain.
  (it describes more than one observation).
- The not definitory slots of a Probabilistic Concept are inheritable through the structural link Bridge.

Example :
ANIMAL
Kind-of
  Value : PC1
Bridge
  Value : <PC'3, 1, 0, 1, 0>
ConceptProb
  Value : 0.75
NbObservations
  Value : 3
LstRealEntities
  Value : {Ee1, Ee2, Ee3}
Perspective
  Value : Physiological
BodyCover
  ProbDomain : {(feathers, 0.20), (hairs, 0.80)}
HeartChamber
  ProbDomain : {(three, 0.33), (four, 0.66)}
Fertilization
  ProbDomain : {(external, 0.33), (Internal, 0.66)}

CONFORT'S ORIGINAL FEATURES

CONFORT is part of the family of systems doing **incremental concept formation that aim at grouping into categories descriptions of real entities**. (CLASSIT (Gennari), LABYRINTH (Thompson), BRIDGER (Reich), CFIX (Handa), OLOC, …) :

The originality of CONFORT lies in :

- its **use of a GDN** to allow the construction of **several hierarchies of concepts**, each one reflecting a **perspective representing a goal of categorization**.

- Its original algorithm that uses a utility measure that is calculated also by means of the **relevancy of the properties represented in the GDN**.

CONFORT'S ORIGINAL FEATURES

- The **use of bridges** between categories enrish the measure of utility.

- The last step of CONFORT, the **transformation of probabilistic concept hierarchies into prototypes hierarchies** allow to improve the representation of concepts :

  • It's much **more intelligible** because of the representation in intension. The prototypical nature of the probabilistic concepts is made explicit.

  • The prunning of the probabilistic concept hierarchies allows to keep only **meaningful concepts**.

Tests

CONFORT has been tested with the database of the description of the bridges in Pittsburgh stored at the Data Base repository of the University of California, Irvine.

Bridges are described by :

- **Specification properties** (including **aesthetic** properties),
- **Design properties** that the bridges have in relation to their specification properties.

The concepts generated embody both types of properties.

Two hierarchies are generated :

- One based on the bridge **specification properties** and **design properties**,
- The second on their **aesthetic properties**.

Tests

The interest of CONFORT lies in its use for the conception task.

Idea :

- **Building the two hierarchies by means of « complete » observations** describing bridges with specification properties and design properties.

- Formulating **a set of constraints for a bridge to be built**, expressed by specification properties that constitute **an observation to be classified**.

- **The concept** found to classify the observation **includes other properties, design ones, that can be inferred** and that the actual bridge to be built must have.

Tests

CONFORT proved to give very interesting results.

In particular, the aesthetic perspective, used through FORMVIEW bridges established between the functional and the aesthetic hierarchies enrished the process, aesthetic properties being viewed as specification properties too.

3 strategies :

1) A **usual classification that goes down to the leaf of the hierarchy** : not well adapted, because specific (e.g. numerical) characteristics can be inferred whereas no bridges are exactly similar.

2) A **case-based approach** : after classifying the observation into a concept, retrieving the cases used to build that concept for the user to study them and to get some inspiration.

1) **Using the prototype generated from the probabilistic concept** where the observation has been classified and inferring characteristics from it.

Sorry for talking so much and thanks a lot for your patient attention !

Any question?

Machine Learning Algorithms

CONFORT is part of the family of systems doing **incremental concept formation that aim at grouping into categories descriptions of real entities**.

Characteristics of the systems (CLASSIT (Gennari), LABYRINTH (Thompson), BRIDGER (Reich), CFIX (Handa), OLOC, …) :

- A **representation of the observations** by pairs (attribute, value) (some use a logical representation).

- A **representation of the concepts** by a conjunction of properties with probability distribution of each property within the category most of the time. is not inspired from the classical (or Aristotelician) view of concepts with CNS, but the typicality of the properties is « hidden » under measures of probability. In some systems, a statistical representation.

Machine Learning Algorithms

- The **organization of the generated concepts** is either a partition of concepts that may overlap or a hierarchy of concepts that do not overlap (except in OLOC).

- The **process of concept formation** is divided into two phases :

• **classification of the observations** to find the more appropriate concept to host them,

• **Learning process where the organization of the concepts is modified** if it's not satisfying enough by means of operators (creation, splitting, merging, etc…).

Machine Learning Algorithms

- The **measure of the quality** of the generated concepts is based on cognitive psychology considerations.

The grouping of observations into categories must allow to infer the highest possible number of properties of a new observation when it's classified into such a category.

The originality of CONFORT lies in :

- its **use of a GDN** to allow the construction of **several hierarchies of concepts**, each one reflecting a **perspective representing a goal of categorization**.

- Its original algorithm that uses a utility measure that is calculated also by means of the **relevancy of the properties represented in the GDN**.

Machine Learning Algorithms

- The **use of bridges** between categories enrish the measure of utility.

- The last step of CONFORT, the **transformation of probabilistic concept hierarchies into prototypes hierarchies** allow to improve the representation of concepts :

  • It's much **more intelligible** because of the representation in intension. The prototypical nature of the probabilistic concepts is made explicit.

  • The prunning of the probabilistic concept hierarchies allows to keep only **meaningful concepts**.

Concept formation via Formal Concept Analysis

One system of Concept Formation adopts the idea from FCA **of representing concepts by mutual closed sets of objects and attributes as well as the Galois lattice structure for concepts.**

The problem in FCA is the **number of all concepts** in a real-world context, in the worst case, that may be an **exponential function of the number of objects and attributes.**

OSHAM (Tu Bao) does not carry out an exhaustive search of the whole concept lattice.

Comparison to other approaches (6)

Concept formation via Formal Concept Analysis

The idea of OSHAM is **to generate only a part of the concept lattice corresponding to a concept hierarchy with a high utility score**.

OSHAM tends to a **tradeoff between the coverage and length of concept's intensions in order to guarantee forming sufficiently general and informative concepts**,

where the coverage $\phi(S)$ of an attribute subset S is defined by

$\phi(S)=card(\rho(S))/card(O)$

in a context (O, A, R) where O is a set of objects, A a set of attributes and R a binary relation between O and A, ρ being defined by :

$\rho(S)=\{o \in O/$ for all $a \in S, (o,a) \in R\}$

Starting from a set of objects, OSHAM **detects and organizes recursively concepts at different levels of generality in the concept hierarchy**. Each level of the hierarchy corresponds to a partition of the whole object set. Each concept is then clustered recursively into subconcepts with more special properties.

Comparison to other approaches (7)

Concept formation via Formal Concept Analysis

- Like COBWEB-like algorithms, including FORMVIEW, OSHAM form concepts with a high utility in such a way that all objects of each concept share the same set of attribute values with highest probabilities $P(p_i/C_k)$

- Unlike to most concept formation systems, **OSHAM is not sensitive to the order of the observation classification**.

- OSHAM works with **observations not having a fixed number of attributes.**

- OSHAM is a **non incremental learning method and it uses only attributes with symbolic values**.

Ontologies

An ontology is the explicit specification of a conceptualisation of a domain. (Gruber).

One must :

- identify and model the relevant concepts and terms.
- identify the relevant relations : subClassOf, isa, partOf, hasPart, closeTo, over, under, contain, connected, etc.
- define rules to combine concepts and relations partOf, for example.

Ontologies are generally built from the knowledge of experts and don't result from a concept formation process.

An ontology can be related to our work in the sense that CONFORT can build ontologies as the hierarchies of prototypes generated at the last step of CONFORT constitute an ontology.