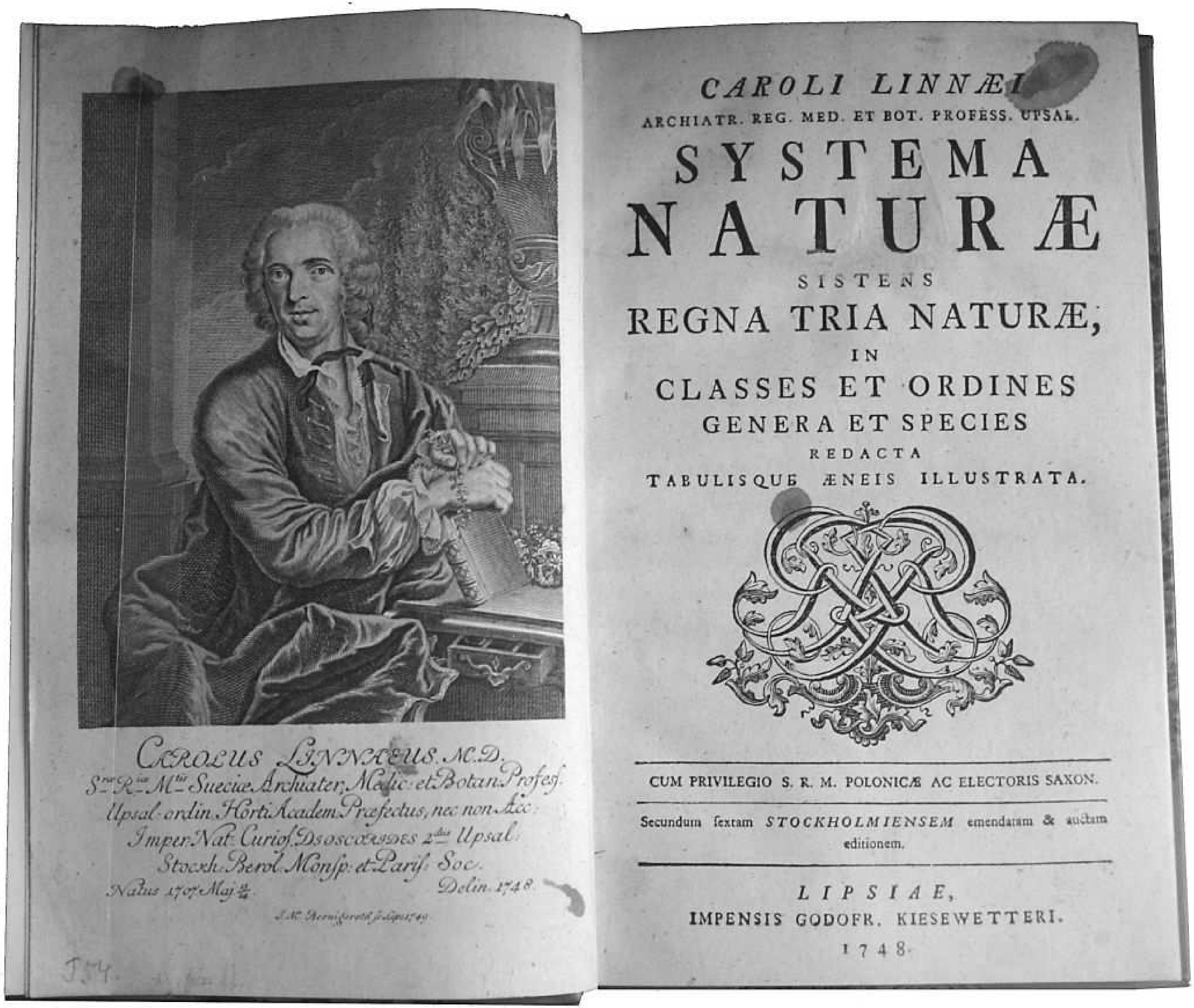
Description Logics and Frames

- Considerable interest in this topics
  - seen as starting point for representing huge quantities of knowledge
  - often used for representing terminologies in particular domains
  - any increase of level of automation will give rise to an increased significance in domain description

- Semantic web:
  OWL DL (Web Ontology Language/Description Logic): formal semantics (model theory) and support for reasoning

http://www.w3.org/TR/owl2-overview/

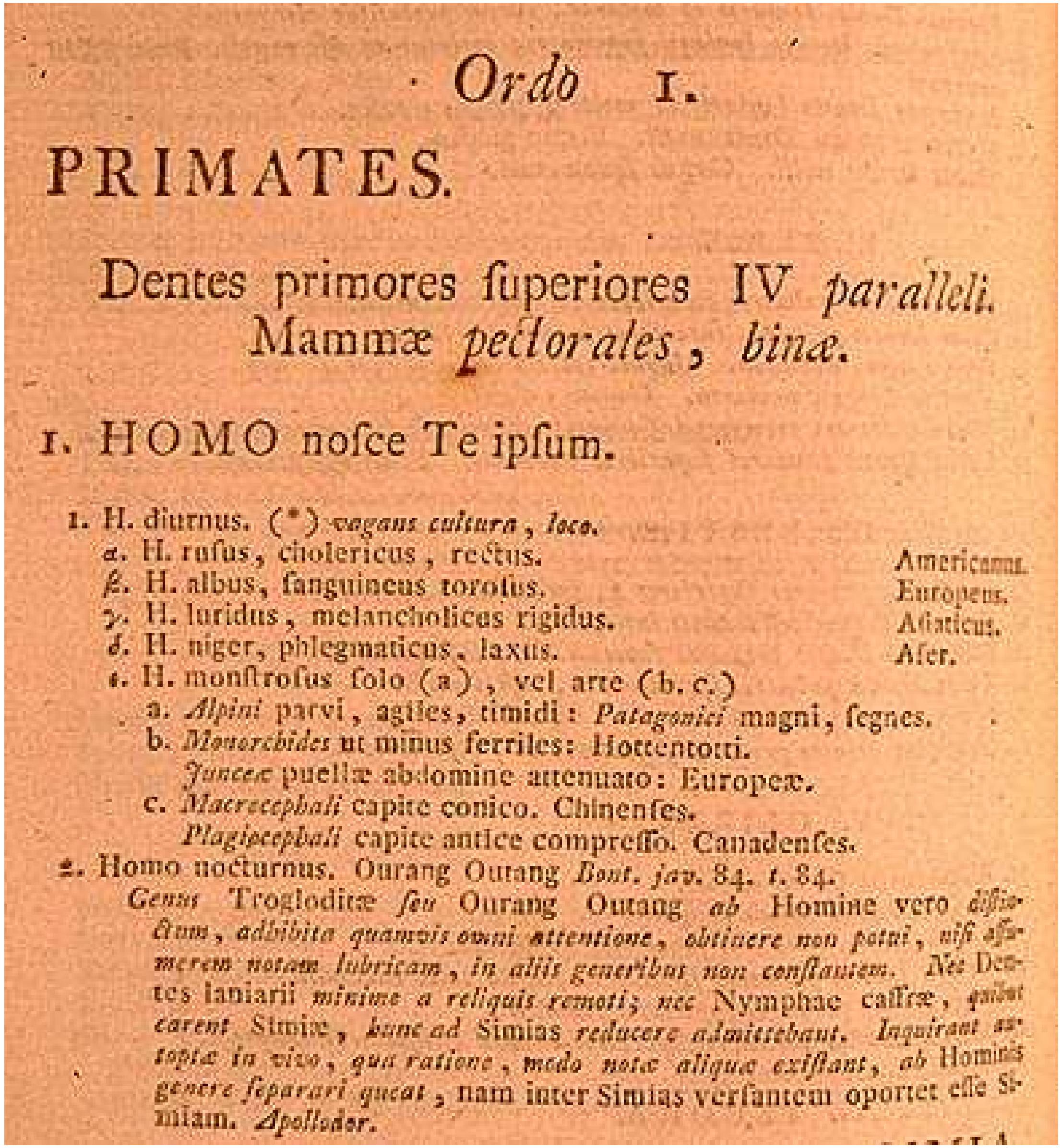
Basic ideas are old ...



Ontologies

- Ontology: science which investigates and explains the nature and essential properties and relations of all beings (Aristotle: Metaphysica)
- In AI, artifact that
  - makes use of a vocabulary,
  - with a set of rules about syntax and meaning,
  - purpose: computer-interpretable description of a domain
- Typical building blocks:
  - names (concepts)
  - relations and constraints
- Popular applications: in biomedical research (e.g. http://bioontology.org/)

Linnaeus



Ordo I.

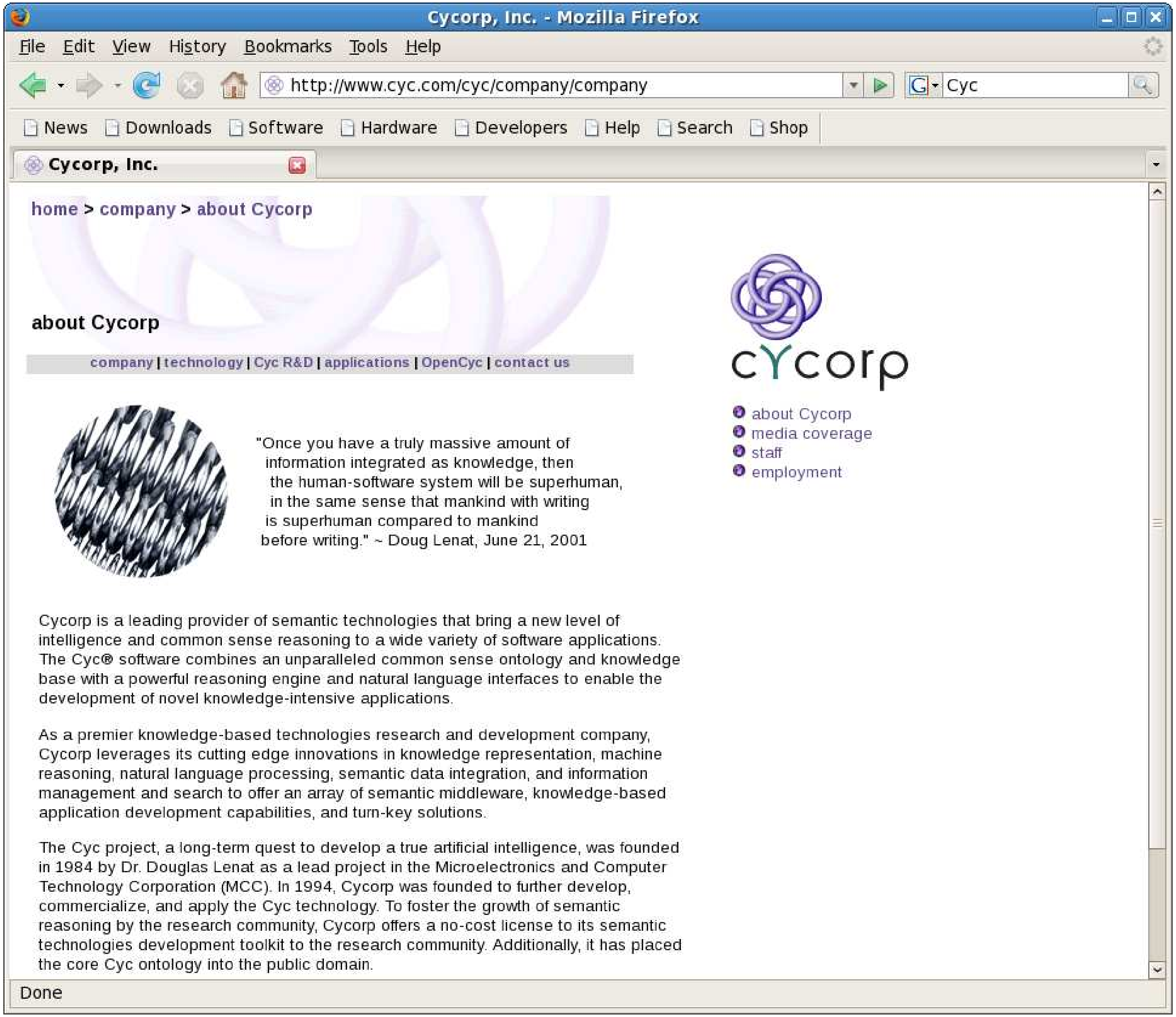
PRIMATES.

Dentes primores ſuperiores IV *paralleli.*
Mammæ *pectorales*, *binæ.*

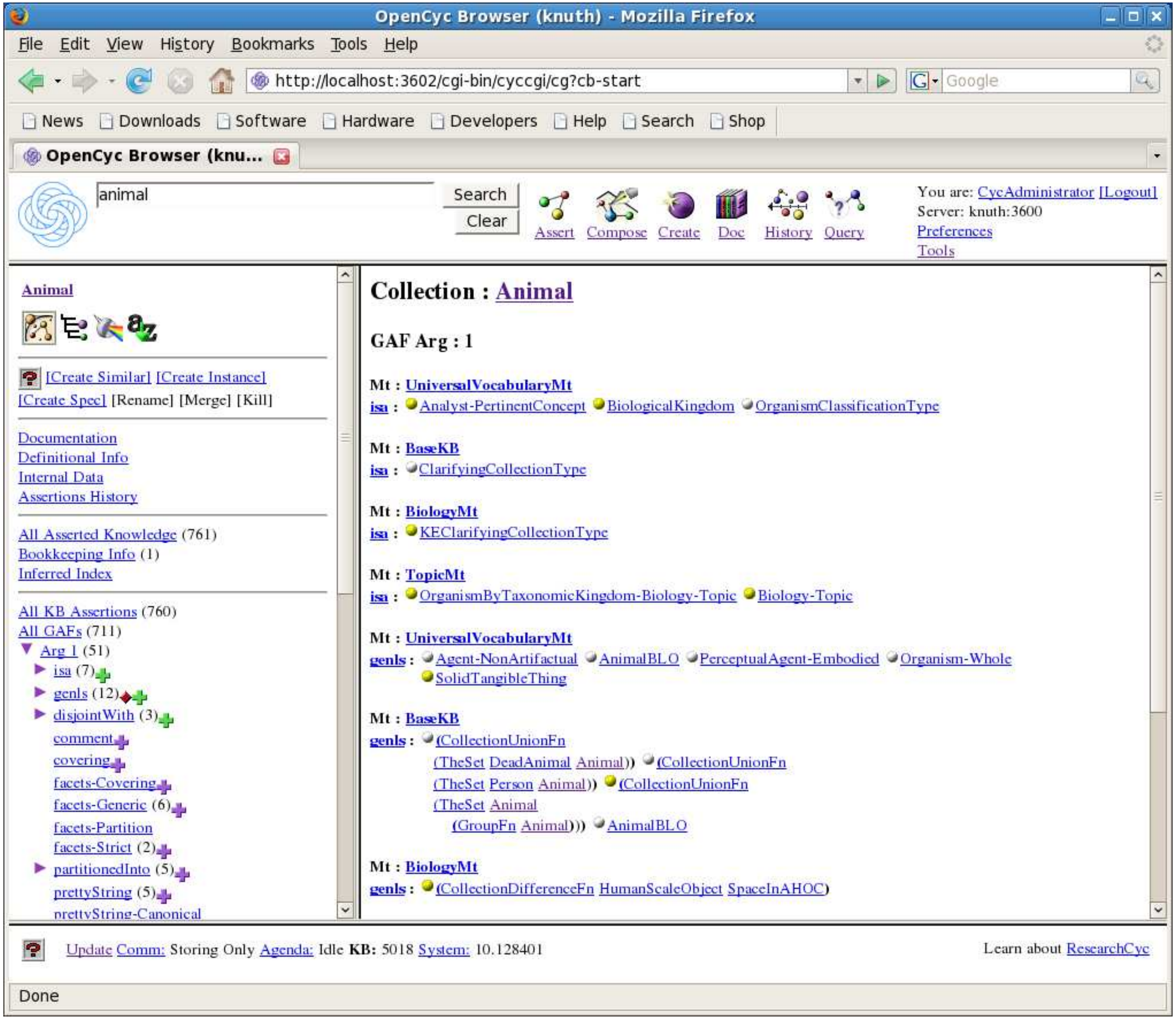
1. HOMO noſce Te ipſum.

1. H. diurnus. (*) *vagans cultura*, *loco.*
α. H. rufus, cholericus, rectus. Americanus.
β. H. albus, ſanguineus toroſus. Europeus.
γ. H. luridus, melancholices rigidus. Aſiaticus.
δ. H. niger, phlegmaticus, laxus. Afer.
ε. H. monſtroſus ſolo (a), vel arte (b. c.)
a. *Alpini* parvi, agiles, timidi: *Patagonici* magni, ſegnes.
b. *Monorchides* ut minus fertiles: Hottentotti.
Junceæ puellæ abdomine attenuato: Europeæ.
c. *Macrocephali* capite conico. Chinenſes.
Plagiocephali capite antice compreſſo. Canadenſes.
2. Homo nocturnus. Ourang Outang *Bont. jav.* 84. *t.* 84.
Genus Troglodytæ *ſeu* Ourang Outang *ab* Homine vero *diſtinctum*, *adhibita quamvis omni attentione*, *obtinere non potui*, *niſi aſſumerem notam lubricam*, *in aliis generibus non conſtantem.* *Nec* Dentes laniarii *minime a reliquis remoti*; *nec* Nymphae caſtræ, *quibus carent* Simiæ, *hunc ad* Simias *reducere admittebant.* *Inquirant autoptæ in vivo*, *qua ratione*, *modo notæ aliquæ exiſtant*, *ab* Hominis *genere ſeparari queat*, nam inter Simias verſantem oportet eſſe Simiam. *Apollodor.*

Cycorp


Cycorp, Inc. - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://www.cyc.com/cyc/company/company
Cyc
News Downloads Software Hardware Developers Help Search Shop
Cycorp, Inc.
home > company > about Cycorp
about Cycorp
company | technology | Cyc R&D | applications | OpenCyc | contact us
"Once you have a truly massive amount of information integrated as knowledge, then the human-software system will be superhuman, in the same sense that mankind with writing is superhuman compared to mankind before writing." ~ Doug Lenat, June 21, 2001
Cycorp is a leading provider of semantic technologies that bring a new level of intelligence and common sense reasoning to a wide variety of software applications. The Cyc® software combines an unparalleled common sense ontology and knowledge base with a powerful reasoning engine and natural language interfaces to enable the development of novel knowledge-intensive applications.
As a premier knowledge-based technologies research and development company, Cycorp leverages its cutting edge innovations in knowledge representation, machine reasoning, natural language processing, semantic data integration, and information management and search to offer an array of semantic middleware, knowledge-based application development capabilities, and turn-key solutions.
The Cyc project, a long-term quest to develop a true artificial intelligence, was founded in 1984 by Dr. Douglas Lenat as a lead project in the Microelectronics and Computer Technology Corporation (MCC). In 1994, Cycorp was founded to further develop, commercialize, and apply the Cyc technology. To foster the growth of semantic reasoning by the research community, Cycorp offers a no-cost license to its semantic technologies development toolkit to the research community. Additionally, it has placed the core Cyc ontology into the public domain.
cycorp
about Cycorp
media coverage
staff
employment
Done

OpenCyc



OpenCyc Browser (knuth) - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost:3602/cgi-bin/cyccgi/cg?cb-start
News Downloads Software Hardware Developers Help Search Shop
animal
Search
Clear
Assert Compose Create Doc History Query
You are: CycAdministrator [Logout]
Server: knuth:3600
Preferences
Tools
Animal
[Create Similar] [Create Instance]
[Create Spec] [Rename] [Merge] [Kill]
Documentation
Definitional Info
Internal Data
Assertions History
All Asserted Knowledge (761)
Bookkeeping Info (1)
Inferred Index
All KB Assertions (760)
All GAFs (711)
Arg 1 (51)
isa (7)
genls (12)
disjointWith (3)
comment
covering
facets-Covering
facets-Generic (6)
facets-Partition
facets-Strict (2)
partitionedInto (5)
prettyString (5)
prettyString-Canonical
Collection : Animal
GAF Arg : 1
Mt : UniversalVocabularyMt
isa : Analyst-PertinentConcept BiologicalKingdom OrganismClassificationType
Mt : BaseKB
isa : ClarifyingCollectionType
Mt : BiologyMt
isa : KEClarifyingCollectionType
Mt : TopicMt
isa : OrganismByTaxonomicKingdom-Biology-Topic Biology-Topic
Mt : UniversalVocabularyMt
genls : Agent-NonArtifactual AnimalBLO PerceptualAgent-Embodied Organism-Whole SolidTangibleThing
Mt : BaseKB
genls : (CollectionUnionFn (TheSet DeadAnimal Animal)) (CollectionUnionFn (TheSet Person Animal)) (CollectionUnionFn (TheSet Animal (GroupFn Animal))) AnimalBLO
Mt : BiologyMt
genls : (CollectionDifferenceFn HumanScaleObject SpaceInAHOC)
Update Comm: Storing Only Agenda: Idle KB: 5018 System: 10.128401
Learn about ResearchCyc
Done

Knowledge server

```
xterm

Start time: Wed Sep 30 23:27:42 CEST 2009
Lisp implementation: Cycorp Java SubL Runtime Environment
JVM: Sun Microsystems Inc. Java HotSpot(TM) Server VM 1.6.0_03 (1.6.0_03-b05)
Current KB: 5018
Patch Level: 10.128401
Running on: knuth
OS: Linux 2.6.23.17-88.fc7 (i386)
Working directory: /home/peterl/Research/Cyc/opencyc-2.0/server/cyc/run
Total memory allocated to VM: 1348MB.
Memory currently used: 541MB.
Memory currently available: 806MB.
Initializing HL backing store caches from units/5018/.
;;   At this point the cyc http server is running and you can access
;;   Cyc directly via the local web browser.
;;   http://localhost:3602/cgi-bin/cyccgi/cg?cb-start
;;   You can browse cyc via the Guest account or perform updates by
;;   logging on as CycAdminstrator.
CYC(1):
```

Representation of relations

1. Semantic nets:
   - syntactic: nodes and relations between nodes
   - represented as a graph
   - semantics of nodes and relationships
2. Frames:
   - (binary) relations represented as attributes
   - inheritance
   - subtyping
3. Description logic:
   - concepts
   - binary relationships
   - restrictions

Basics of description logics

Example DL:
$\mathcal{ALC} = \mathcal{A}$ttribute concept $\mathcal{L}$anguage with $\mathcal{C}$omplement

Basic ingredients:

- concepts
- roles
- Boolean operators

"A man is married to a doctor, and all of whose children are either doctors or professors"

$$\text{Human} \sqcap \neg\text{Female} \sqcap \exists\text{married.Doctor})$$
$$\sqcap(\forall\text{hasChild.}(\text{Doctor} \sqcup \text{Professor}))$$

Language elements

Concept descriptions:

- primitive concept C, e.g., Human, $\top$ (most general), $\bot$ (empty)
- primitive role r, e.g., hasChild
- conjunction $\sqcap$, e.g., SmartHuman $\sqcap$ Student
- disjunction $\sqcup$, e.g., Truck $\sqcup$ Car
- complement $\neg$, e.g., $\neg$Human
- value restriction $\forall r.C$, e.g., $\forall$hasChild.Doctor
- existential restriction $\exists r.C$, e.g., $\exists$happyChild.Parent

All understood in terms of (groups of) individuals and properties of individuals

General concept inclusion (GCI)

General concept inclusion (GCI) also called subsumption

For concepts C, D:

- $C \sqsubseteq D$, e.g., Professor $\sqsubseteq$ Person
- definition $C \equiv D$ is the same as $C \sqsubseteq D$ and $D \sqsubseteq C$ (Not $(C \sqsubseteq D) \sqcap (D \sqsubseteq C)$, why?)
- C in $C \equiv D$ is called a defined concept, whereas D consists of primitive concepts, e.g.,

$$\text{Father} \equiv \neg\text{Female} \sqcap \exists\text{hasChild.Human}$$

Concrete descriptions

Instances of concepts or roles, called assertions:

- $c : C$, means that c is an instance of concept C, e.g.,

$$\text{John} : \text{Person}$$

- $(b, d) : r$, means that the pair of individuals (b, d) is an instance of role r, e.g.,

$$(\text{John}, \text{Mary}) : \text{marriedTo}$$

Knowledge base

Knowledge Base = KB

Terminology = TBox

$\text{Father} \equiv \neg\text{Female} \sqcap \exists\text{hasChild.Human}$

$\text{Human} \sqsubseteq \text{Animal}$

Concrete assertions = ABox

$\text{John} : \text{Father}$

$(\text{John}, \text{Sheila}) : \text{hasChild}$

$\text{KB} = (\text{TBox}, \text{ABox})$

Knowledge base

$\text{KB} = (\text{TBox}, \text{ABox})$:

- TBox: contains general descriptions, definitions, subsumptions relationships (GCIs)

  $\text{Father} \equiv \neg\text{Female} \sqcap \exists\text{hasChild}.\text{Human}$
  $\text{Human} \sqsubseteq \text{Animal}$

- ABox: contains description of individuals (instances)

  $\text{John} : \text{Father}$
  $(\text{John}, \text{Sheila}) : \text{hasChild}$

- Sometimes $\text{ABox} = \emptyset$, then only interested in general principles: terminological reasoning

Meaning of description logic

In terms of set theory

Let $\mathcal{I} = (\Delta, \cdot)$ be an interpretation, then

- $\top^{\mathcal{I}} = \Delta$, and $\bot^{\mathcal{I}} = \varnothing$
- each concept $C^{\mathcal{I}} \subseteq \Delta$
- each role $r^{\mathcal{I}} \subseteq \Delta \times \Delta$
- $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$
- $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$
- $(\exists r.C)^{\mathcal{I}} = \{c \in \Delta \mid \exists d \in C^{\mathcal{I}}$ with $(c, d) \in r^{\mathcal{I}}\}$
- $(\forall r.C)^{\mathcal{I}} = \{c \in \Delta \mid \forall d \in \Delta,$ if $(c, d) \in r^{\mathcal{I}}$ then $d \in C^{\mathcal{I}}\}$

where $C^{\mathcal{I}}$ and $r^{\mathcal{I}}$ are interpretations of C and r as sets

Example

$\boxed{\text{Father} \equiv \neg\text{Female} \sqcap \exists\text{hasChild.Human}}$

Interpretation $\mathcal{I} = (\Delta, \cdot)$, with $\Delta = \{\text{John}, \text{Sheila}\}$

- $\text{Father}^{\mathcal{I}} = \{\text{John}\} \subseteq \Delta$
- $\text{Human}^{\mathcal{I}} = \{\text{John}, \text{Sheila}\}$
- $\text{hasChild}^{\mathcal{I}} = \{(\text{John}, \text{Sheila})\}$
- $(\exists\text{hasChild.Human})^{\mathcal{I}} = \{\text{John}\}$

Meaning of subsumption

Interpretation $\mathcal{I} = (\Delta, \cdot)$, then

$$(C \sqsubseteq D)^{\mathcal{I}} = C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$$

Example:

$$\text{Father} \sqsubseteq \text{Human}$$

$\Delta = \{\text{John}, \text{Sheila}\}$, then

$$\text{Father}^{\mathcal{I}} = \{\text{John}\}$$

and

$$\text{Human}^{\mathcal{I}} = \{\text{John}, \text{Sheila}\}$$

Relationship with predicate logic

Translation function τ_x (DL $\to$ first-order predicate logic) that introduces variable x:

- $\tau_x(C) = C(x)$
- $\tau_x(C \sqcap D) = (\tau_x(C) \wedge \tau_x(D))$
- $\tau_x(C \sqcup D) = (\tau_x(C) \vee \tau_x(D))$
- $\tau_x(\exists r.C) = \exists y\ r(x,y) \wedge \tau_y(C)$
- $\tau_x(\forall r.C) = \forall y\ r(x,y) \rightarrow \tau_y(C)$

Example: $\neg$Female $\sqcap$ $\exists$hasChild.Human:

$\tau_x(\neg\text{Female} \sqcap \exists\text{hasChild.Human})$
$= \tau_x(\neg\text{Female}) \wedge \tau_x(\exists\text{hasChild.Human}))$
$= \neg\text{Female}(x) \wedge \tau_x(\exists\text{hasChild.Human}))$
$= \neg\text{Female}(x) \wedge \exists y(\text{hasChild}(x, y) \wedge \text{Human})(y))$

Relationship with predicate logic

- GCIs $C \sqsubseteq D$ in TBox:

$$\bigwedge_{C \sqsubseteq D \in \text{TBox}} \forall x(\tau_x(C) \to \tau_x(D))$$

- Thus, $\sqsubseteq$ becomes logical implication

Example: Translate

$$\text{UnivTeacher} \sqsubseteq \text{Prof} \sqcup \neg\text{Undergraduate}$$

to predicate logic:
$\tau_x(\text{UnivTeacher} \sqsubseteq \text{Prof} \sqcup \neg\text{Undergraduate})$
$= \forall x(\text{UnivTeacher}(x) \to \tau_x(\text{Prof} \sqcup \neg\text{Undergraduate}))$
$= \forall x(\text{UnivTeacher}(x) \to (\text{Prof}(x) \vee \neg\text{Undergraduate}(x)))$

Relationship with predicate logic

- Translation of ABox:

$$\tau_x(\mathsf{ABox}) = \bigwedge_{c:C\in\mathsf{ABox}} \tau_c(C) \wedge \bigwedge_{(b,d):r\in\mathsf{ABox}} r(b,d)$$

- Thus, unit clauses of the form $C(c)$ and $r(b,d)$

Example:

$$\mathsf{ABox} = \{\mathsf{John} : \mathsf{Father}, (\mathsf{John}, \mathsf{Sheila}) : \mathsf{hasChild}\}$$

In first-order logic:

$$\mathsf{Father}(\mathsf{John}) \wedge \mathsf{hasChild}(\mathsf{John}, \mathsf{Sheila})$$

Reasoning

Possible procedure:

- Translate DL knowledge base to first-order logic
- Use a reasoning engine, e.g., resolution, for reasoning, then

  $$\tau(\mathsf{KB}) \vdash \phi$$

  with ϕ something like $\tau_x(\mathsf{Prof} \sqsubseteq \mathsf{Human})$ becomes:

  $$\tau(\mathsf{KB}) \land \neg\phi \vdash \bot$$

  (i.e., if KB is consistent and KB $\cup \neg\phi$ is inconsistent, then ϕ follows from KB)
- However, special purpose reasoning may be advantageous

Typical questions

Let KB $=$ (TBox, ABox), then:

- KB $\models C \sqsubseteq D$? (is C subsumed by D?)
- KB $\models c : C$? (is c an instance of C?)
- KB $\models (b, d) : r$? (is the pair (b, d) true for role r?)

Reasoning can be reduced to consistency checking:

- $(\text{TBox}, \text{ABox} \cup \{c : C \sqcap \neg D\}) \vdash \bot$
- $(\text{TBox}, \text{ABox} \cup \{c : \neg C\}) \vdash \bot$
- $(\text{TBox}, \text{ABox} \cup \{(b, d) : \neg r\} \vdash \bot$

Summary DL

- Description logics: restricted logical languages for the representation of conceptual information and terminologies
- Basis for all large-scale attempts for the development of knowledge bases, e.g. semantic web and biomedical ontologies
- Advantages:
  - Restricted syntax allows developing special purpose, efficient reasoning systems (e.g., Racer, Fact, Cyc)
  - $\mathcal{ALC}$ is decidable! (a 2 variable fragment of first-order logic)
- Disadvantage: many things cannot be represented in a DL

Frame formalism

Predecessor of description logics, still in use

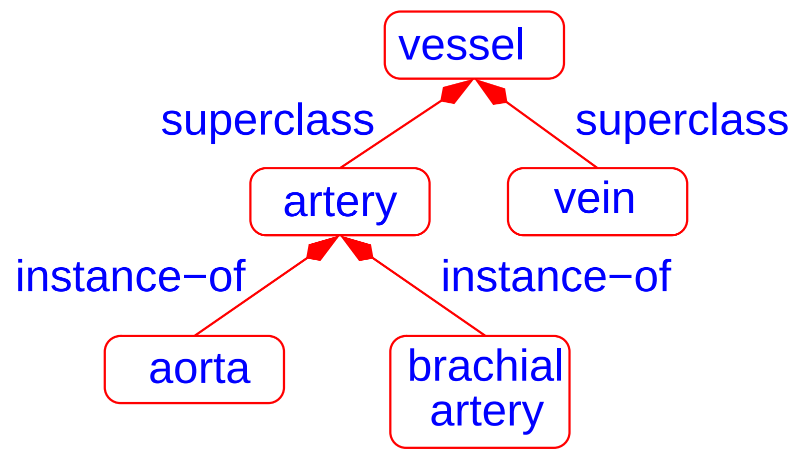
Concepts and properties of concepts

Example (human anatomy): “An artery is a vessel. An artery transports blood from the heart to the tissues, and is characterised by a thick wall and much muscular tissue.”

Frame taxonomy: hierarchical organisation

- nodes (vertices): classes and instances
- arcs (arrows): superclasses and instance-of relationships

Example



Syntax

Basic elements:

- subclass relationship between frames
- attributes or slots
- values or fillers

⇒ attribute-value pair or slot-filler combination

Example: (structure, tube) is an attribute-value pair

class vessel **is**
  superclass nil;
  structure = tube;
  contains = blood
end

class artery **is**
  superclass vessel;
  wall = muscular
end

instance aorta **is**
  instance-of artery;
  diameter = 3cm
end

Meaning of frames

Semantics in terms of predicate logic:

class C **is**	$\forall x(C(x) \rightarrow S(x))$
superclass S;	
$a_1 = b_1$;	$\forall x(C(x) \rightarrow a_1(x) = b_1)$
$\vdots$	$\vdots$
$a_n = b_n$	$\forall x(C(x) \rightarrow a_n(x) = b_n)$
end	
instance i **is**	$C(i)$
instance-of C;	
$d_1 = e_1$;	$d_1(i) = e_1$
$\vdots$	$\vdots$
$d_m = e_m$	$d_m(i) = e_m$
end	

Relationship with description logic

	TBox:
class C **is**	$C \sqsubseteq S$
superclass S;	
$a_1 = b_1$;	$C \sqsubseteq \exists a_1.\{b_1\}$
$\vdots$	$\vdots$
$a_n = b_n$	$C \sqsubseteq \exists a_n.\{b_n\}$
end	

($\{i\}$ is a concept if i is an individual)

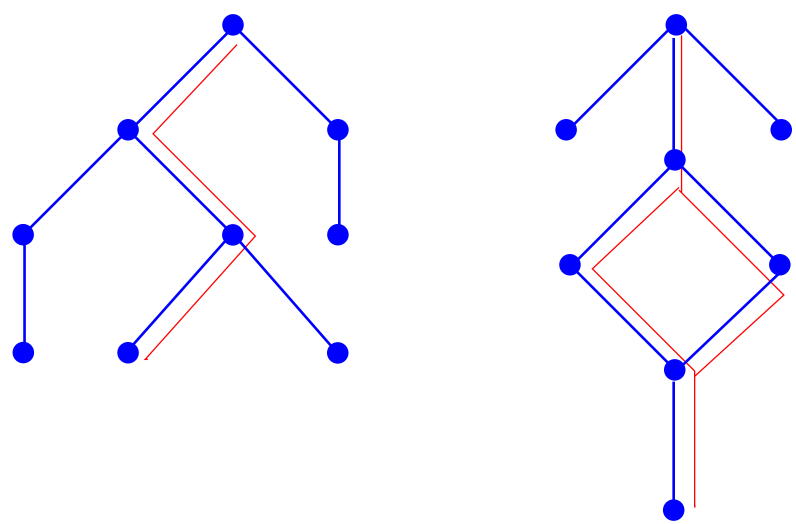
	ABox:
instance i **is**	$i : C$
instance-of C;	
$d_1 = e_1$;	$(i, e_1) : d_1$
$\vdots$	$\vdots$
$d_m = e_m$	$(i, e_m) : d_m$
end	

Reasoning

- Basic reasoning method is called inheritance, sometimes example of non-monotonic reasoning
- Frame inherits attribute-value pairs from its generalisations

Two types of inheritance:

1. single: tree-shaped taxonomy
2. multiple: general graph-shaped taxonomy (loops)



Single inheritance

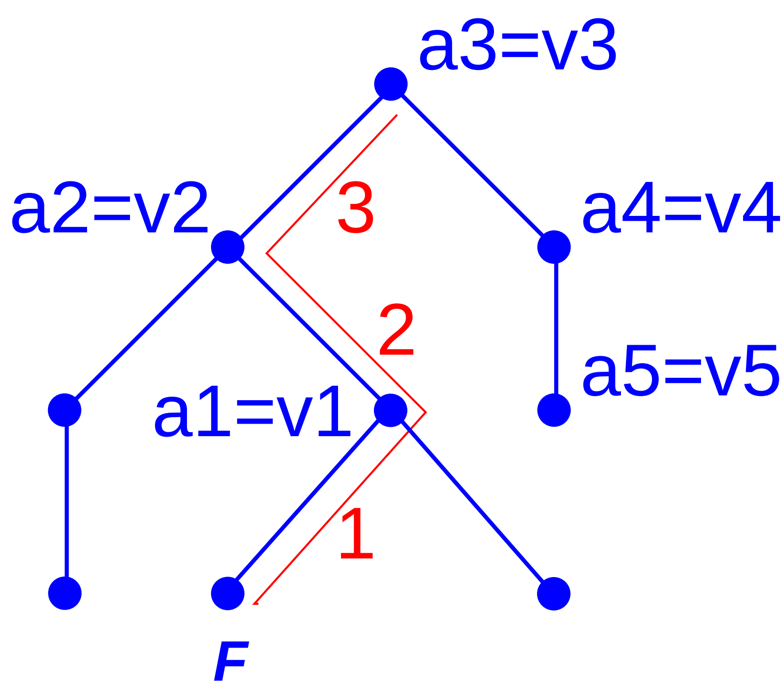
Requirement: unique attribute names

function Inherit(frame, a-v-pairs)

    if frame = **nil**
    then return(a-v-pairs)
    fi;
    a-v-pairs $\leftarrow$ a-v-pairs $\cup$
            AttributePart(frame);
    return(Inherit(superframe(frame),
            a-v-pairs))

end

Example



F inherits $a_1 = v1, a_2 = v_2$ en $a_3 = v_3$

Exceptions

Non-unique attribute names $\Rightarrow$ exceptions / inconsistency

Example

class artery **is**
  instance-of artery;
  superclass vessel;
  blood = oxygenrich
end

instance pulmonary-a **is**
  instance-of artery;
  blood = oxygenpoor
end

$\forall x(\text{artery}(x) \rightarrow \text{vessel}(x))$
$\forall x(\text{artery}(x) \rightarrow \text{blood}(x) = \text{oxygenrich})$
artery(pulmonary-a)
blood(pulmonary-a) = oxygenpoor

Logically inconsistent!

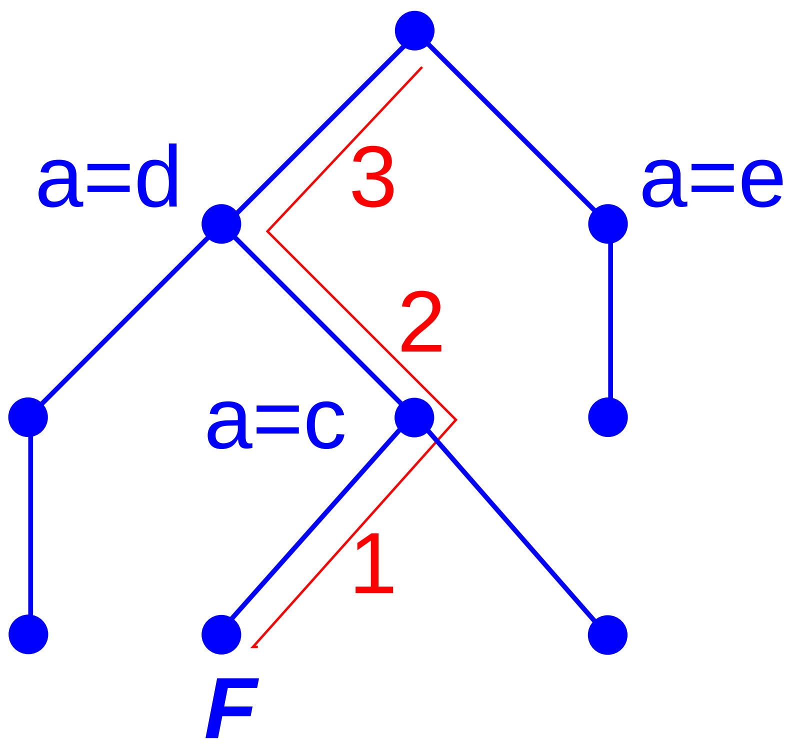
Solution: local 'overwriting'

Single inheritance with exceptions

Search the taxonomy until a value is found:

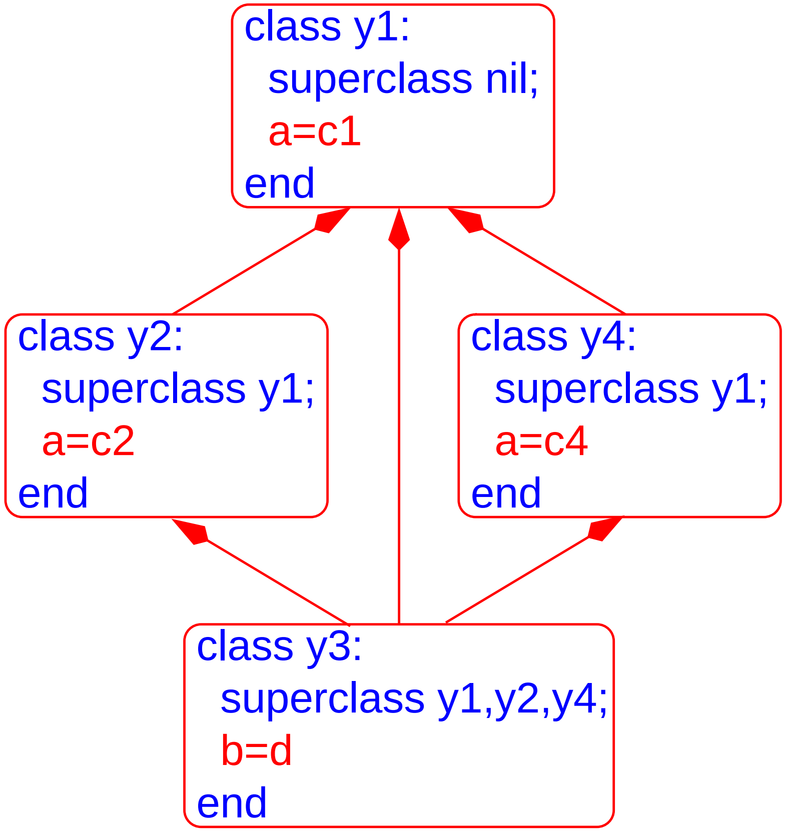
function Inherit(frame, a-v-pairs)
    if frame = **nil**
    then return(a-v-pairs)
    fi;
    pairs $\leftarrow$ AttributePart(frame);
    a-v-pairs $\leftarrow$ a-v-pairs $\cup$
        NewAttributes(pairs, a-v-pairs);
    return(Inherit(superframe(frame), a-v-pairs))
end

Example



F inherits $a = c$

Multiple inheritance with exceptions



Questions:

1. Taxonomy consistent?
2. If yes, what are the inherited values for attribute a vof y_3?

Requirement: solution must work for both: graph and tree shaped taxonomies!

Monotonic reasoning

- Inheritance with exceptions example of non-monotonic reasoning
- Monotonic reasoning:
  - knowledge base KB
  - add knowledge to KB and obtain new knowledge base KB$'$
  - if KB $\vdash$ Results and KB$'$ $\vdash$ Results$'$ then Results $\subseteq$ Results$'$
  - more knowledge yields more results
- Example:

  KB $= \{P \rightarrow Q\}$ KB$' = \{P \rightarrow Q, P\}$

  Results $=$ KB Results$' =$ KB$' \cup \{Q\} =$ KB $\cup \{P, Q\}$

Non-monotonic reasoning

- Non-monotonic reasoning is more close related to human reasoning
  - knowledge base KB
  - add knowledge to KB and obtain new knowledge base KB$'$
  - if KB $\vdash$ Results and KB$'$ $\vdash$ Results$'$ then Results $\subseteq$ Results$'$ does not hold in general
  - more knowledge does not necessarily yield more results
- Example ($\mid\!\sim$ is non-monotonic):

$$\{\mathit{artery}(\mathit{left\text{-}pulmonary\text{-}artery}),$$
$$\mathit{blood}(\mathit{left\text{-}pulmonary\text{-}artery}) = \mathit{oxygen\text{-}poor},$$
$$\forall x(\mathit{artery}(x) \rightarrow \mathit{blood}(x) = \mathit{oxygen\text{-}rich})\}$$
$$\mid\!\sim \mathit{blood}(\mathit{left\text{-}pulmonary\text{-}artery}) = \mathit{oxygen\text{-}poor}$$