How to do (certain) things with frames

Sebastian Löbner & Ralf Naumann

SFB 991 colloquium 27.11.2014

1 Formal frame definitions

1.1 Topology

Frames are a network of „nodes“ and links between them. Depending on the type of concept which a frame represents, this network has certain topological properties: What is linked to what? Is the network as a whole connected? Are loops admitted? etc.

- For all types of frames, the network is connected. A frame network forms one connected whole.
- The links are directed.
- There may be a source node from which every other node can be reached via a series of one or more links.

Later, links will correspond to attributes. They link the arguments of attributes to their values.

1.1 Topology

Later in frames:
I = set of nodes, V = attribute links = argument-value pairs, r = central node

Definition 1 Access relations

$V \subseteq I^2$ is a non-empty finite set.
The **immediate access relation → on I** is V itself, i.e.

a. i→j iff ⟨i, j⟩ ∈ V for all i, j ∈ I.

The access relation ⇒ on I is the transitive closure of V:

b. For all i, j ∈ I, if i → j, then i ⇒ j.
For all i, j, k ∈ I: if i ⇒ j and j ⇒ k, then i ⇒k.

Definition 2 Networks with a source

$V \subseteq I^2$ is a nonempty finite set. $I_V = \{ i \in I : \exists \langle x, y\rangle \in V \text{ such that } i=x \text{ or } i=y\}$.
V is **unidirectionally connected, with source r** iff

a. $r \in I_V$

b. For all $i \in I_V$, $i \neq r$, $r \Rightarrow i$.

1.2 Abstract frame structures

An abstract frame structure for sortal concepts is based on a unidirectionally connected network with a source node r. Nodes are marked by variables (I), links are labeled with 'attribute labels' (A), nodes may carry a class label (C). A partial function *l* links nodes and attributes to their value nodes. A partial function *v* labels nodes with class labels.

Example
Frame structure BEM (a brown-eyed male person)

I: r, x, y, z

A: EYES, COLOR, SEX

C: person, brown, male

l: *l*(r, EYES) = x   *l*(x, COLOR) = y   *l*(r, SEX) = z

v: *v*(r) = person   *v*(y) = brown   *v*(z) = male

1.2 Abstract frame structures

Definition 3. Sortal frame structure

A **sortal frame structure** $\mathfrak{f}$ is a 6-tupel $\langle r, I, A, C, l, v\rangle$ such that

a. I is a finite set of individual lables, 'r' ∈ I.

b. A is a finite set of attribute labels.

c. C is a finite set of class labels

d. l is a partial function l: I×A⟶I. The set $\{\langle i, j\rangle \subseteq I^2 : \exists a{\in}A \;\; l(i, a){=}j\}$ is unidirectionally connected, with source r.

e. v is a partial function v: I⟶C

The function v optionally assigns class labels to nodes. They symbolize a class specification of the potential referent of the node.

- A frame structure is a **symbolic structure**, a network of 'nodes' and directed links between them. All 'nodes', 'attributes' , and 'classes' (= types) are just labels. They do not denote anything as long as they are not assigned denotations.

1.3 Relating frame structures to graphs

Example

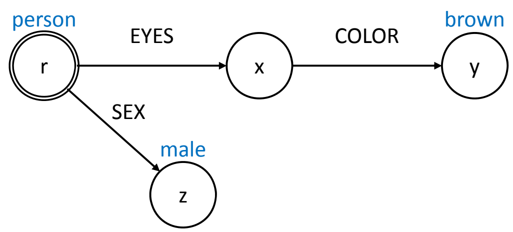
I: r, x, y, z

A: EYES, COLOR, SEX

C: person, brown, male

l: l(r, EYES) = x l(x, COLOR) = y l(r, SEX) = z

v: v(r) = person v(y) = brown v(z) = male



1.3 Relating frame structures to graphs

Definition 4. Associated frame graph

For a frame structure $\mathfrak{f} = \langle r, I, A, C, l, v\rangle$, the **associated graph Gr($\mathfrak{f}$)** is a directed labeled graph with the set of nodes N and arcs V, such that

a. There is a bijection n: I⟶ N; n(i) is labeled with 'i'. N = n[I].

b. There is exactly one arc in V from n(i) to n(j), labeled with 'a' iff l(i, a) = j.

c. Tor every i ∈ I, if there is c ∈ C such that C(i)=c, then n(i) carries the specification 'c'.

Remark

A frame graph can be regarded as a complex two-dimensional *expression* . If such an expression is interpreted, e. g. by relating the labels to an ontology, it constitutes a *representation*.

1.4 Ontologies

An ontology $\mathfrak{O}$ is a highly complex system of individuals and attributes that assign values to these individuals.

Individuals are those entities which can be arguments or values of an attribute.

Attributes are partial functions that map individuals on individuals. Attributes may be one-place or multiplace. They must not be empty, i.e. they must assign a value to at least some individuals or tuples of individuals.
The set of attributes is closed under functional composition. For example, if HAIR and COLOR are attributes in $\mathfrak{O}$, then COLOR_OF_HAIR is too. The set of attributes is also closed under inversion of injective attributes. For example, if HEAD is an attribute, then its inversion HEAD^{-1}(for the head owner) is also an attribute in $\mathfrak{O}$.

1.4 Ontologies

Sorts The universe is partitioned into sorts, such as real world physical objects, numbers, temperatures, truth-values. Every individual is of exactly one sort. The sorts derive from the fact that for every attribute the (i-th) domain is restricted to individuals of the same sort; also, every attribute returns values of one sort only. Some sorts may be ordered, e.g. the sorts corresponding to the values of attributes such as SIZE, PRICE, AGE, or the sort TIME.

Classes The classes in $\mathfrak{O}$ form a sorted type hierarchy. There are no classes that contain elements of different sorts. For every individual x, there is the atomic class {x}. The set of classes is closed under intersection, and closed under the image and pre-image operation on attributes. For example, the class hair forms a subclass of the domain of the attribute COLOR; its image is the class hair color. Conversely, the pre-image of the COLOR attribute of the class {Red} forms the class red object within the domain class of COLOR.

1.4 Ontologies

Ontological question (1)

Are there individuals in the natural ontology that can only be values, but not arguments of attributes?

Are there terminal nodes in frames that cannot expanded?

Candidates: Values of property attributes such as COLOR LENGTH SHAPE

Definitions	f: A ⟶B is a partial function, A'⊆A, B'⊆B				
image	f[A']	$=_{df}$	{ y∈B : ∃x∈A' f(x) = y }		
preimage	f^{-1}[B']	$=_{df}$	{ x∈A : ∃y∈B' f(x) = y }		
domain	dom(f)	$=_{df}$	f^{-1}[B]	=	{ x∈A : ∃y∈B f(x) = y }
codomain	cod(f)	$=_{df}$	f[A]	=	{ y∈B : ∃x∈A f(x) = y }

1.4 Ontologies

Definition 5. [First-order] ontology

A first-order **ontology** $\mathfrak{O}$ is a quadrupel ⟨U, $\mathcal{A}$, $\mathcal{S}$, $\mathcal{C}$ ⟩ such that

a. U is the universe, a non-empty set of individuals.

b. $\mathcal{A}$, the set of attributes, is a set of non-empty partial functions a: $U^n \longrightarrow U$

c. $U = \bigcup_{a\in\mathcal{A}} dom(a) \cup \bigcup_{a\in\mathcal{A}} cod(a)$

d. $\mathcal{A}$ is closed under functional composition.

e. If a ∈ $\mathcal{A}$ is injective , then there is a partial function a'∈ $\mathcal{A}$, such that for every x, y ∈ U, a'(y) = x iff a(x) = y. (*Closure under inversion*).

f. There is a partition $\mathcal{S}$ of U into sorts such that for every a ∈ $\mathcal{A}$, a: $U^n \longrightarrow U$ there are sorts s, s_1, …, s_n ∈ $\mathcal{S}$ such that for all i = 1, …, n $dom_i(a) \subseteq s_i$ and cod(a) ⊆ s.

g. $\mathcal{C} \subset \wp(U)$. For every c ∈ $\mathcal{C}$, there is an s ∈ $\mathcal{S}$ with c ⊆ s.

h. For every x ∈ U, {x} ∈ $\mathcal{C}$. {{x}| x ∈ U} is the set of atoms in $\mathcal{C}$.

i. If a ∈ $\mathcal{A}$, c ⊆ dom(a), then a[c] ∈ $\mathcal{C}$
if a ∈ $\mathcal{A}$, c ⊆ cod(a), then a^{-1}[c] ∈ $\mathcal{C}$

j. For every c, c' ∈ $\mathcal{C}$, c ∩ c' ∈ $\mathcal{C}$.

1.5 Relating frames to ontologies

We can assign meaning to a frame structure – turn it into a *representation* - by relating it to an ontology, basically by using terms for individuals, attributes, and classes in the ontology as labels in the frame structure.

Definition 6. Ontologically specified frame

An ontologically specified frame (OSF) on the ontology $\mathfrak{O}$ = ⟨U, $\mathcal{A}$, $\mathcal{S}$, $\mathcal{C}$ ⟩ is a frame structure $\mathfrak{f}$ = ⟨r, I, A, C, l, v⟩ such that

a. I is a non-empty subclass of individual term (variable of constant)

b. A is a set of labels that denote an attribute in $\mathcal{A}$

c. C is a set of labels that denote a class in $\mathcal{C}$

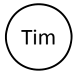
Remark OSFs on $\mathfrak{O}$ are **representations** of classes or individuals in $\mathfrak{O}$.

1.6 Trivial (but useful) frames

Two types of trivial frames

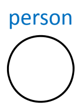
For any individual term t , $\mathfrak{f}_t = \langle t, \{t\}, \emptyset, \emptyset, \emptyset, \emptyset \rangle$

$\mathfrak{f}_t$ is the frame that consists only of one node, labeled with t.



For any class term cl, $\mathfrak{f}_{cl} = \langle x, \{x\}, \emptyset, \langle x, cl \rangle, \emptyset, \emptyset \rangle$

$\mathfrak{f}_{cl}$ is a frame that consists of only one node, labeled with a variable and the class label cl. The variable is to be chosen in context as to avoid variable collision.



1.7 Metalanguages

The metalanguage of an ontology $\mathfrak{D}$

We may just ordinary language or a first-order predicate logic as a metalanguage for a given ontology $\mathfrak{D}$. In both cases, the lexicon of the metalanguage includes the following sets of expressions:

IndVar Individual variables ranging over the elements of the universe U

IndCon Individual constants: designations of particular individuals in U

IndTerm The union of IndVar and IndCon

AttrCon Designations of particular attributes in $\mathcal{A}$

ClassCon Designations of particular classes in $\mathcal{C}$

1.7 Metalanguages

Definition 7

For a frame structure $\mathfrak{f} = \langle r, I, A, C, l, v \rangle$, the **associated PL1 language PL1($\mathfrak{f}$)** is a standard PL-1 language such that

1. a. All elements of I are in IndVar or IndCon.
   b. All elements of A are in AttrCon.
   c. All elements of C are in ClassCon.
2. a. For all n-place function constants a and individual terms $t_1, \ldots, t_n$, '$a(t_1, \ldots, t_n)$' is an individual term.
   b. For all n-place function constants a and individual terms $t_1, \ldots, t_n$, t '$a(t_1, \ldots, t_n) = t$' is a formula of PL1($\mathfrak{f}$).
   c. For all set constants c and all individual terms t, '$t \in c$' is a formula of PL1($\mathfrak{f}$).
3. a. If ϕ and ψ are formulae, so is '$\phi \wedge \psi$'
   b. If ϕ is a formula, and x a variable, $\exists x(\phi)$ is a formula.

1.7 Metalanguages

Definition 8

Let $\mathfrak{f} = \langle r, I, A, C, l, v \rangle$ be a frame on the ontology $\langle U, \mathcal{A}, \mathcal{S}, \mathcal{C} \rangle$, let PL1($\mathfrak{f}$) be the associated PL1 language. A **canonical model** of PL1($\mathfrak{f}$) is a pair $\langle U, [\![\ ]\!] \rangle$ such that:

1. a. For every individual variable $i \in I$, $[\![i]\!] \in U$;
   for every individual constant $t \in I$, $[\![t]\!] = t$
   b. For every $a \in A$, $[\![a]\!] = a$
   c. For every $c \in C$, $[\![c]\!] = c$
2. For all n-place function constants a and individual terms $t_1, \ldots, t_n$,
   $[\![a(t_1, \ldots, t_n)]\!] = a([\![t_1]\!], \ldots, [\![t_n]\!])$
3. a. For all n-place function constants a and individual terms $t_1, \ldots, t_n$, t
   $[\![a(t_1, \ldots, t_n) = t]\!]$ = true iff $a([\![t_1]\!], \ldots, [\![t_n]\!]) = [\![t]\!]$
   b. For every set constants c and every individual term t,
   $[\![t \in c]\!]$ = true iff $[\![t]\!] \in c$
4. a. For all formulae ϕ and ψ,
   $[\![\phi \wedge \psi]\!]$ = true iff $[\![\phi]\!] = [\![\psi]\!]$ = true
   b. For every formula ϕ, and variable x,
   $[\![\exists x(\phi)]\!]$ = true iff there is an individual $u \in U$ such that $[\![\phi]\!]$ = true for $[\![x]\!] = u$.

1.8 Translating frames into [formal] language

Definition 9. Canonical satisfaction formula for a frame $\mathfrak{f}$

For a frame structure F = ⟨r, I, A, C, *l*, *v*⟩ there is a canonical satisfaction formula SatFor($\mathfrak{f}$).

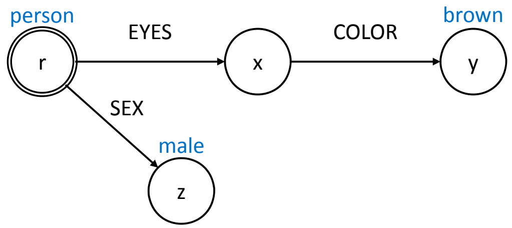
a. If $\mathfrak{f}$ is a trivial frame $\mathfrak{f}_t$, the formula is 't=t'

b. If *l* or v is not empty, SatFor($\mathfrak{f}$) is the conjunction of all the statements

(1)	'r∈c'	if *v*(r) = c
(2)	'a(i)=j ∧ j∈c'	if there are i, j ∈ I such that *l*(i, a) = j and *v*(j) = c
(3)	'a(i)=j'	if there are i, j ∈ I such that *l*(i, a) = j and *v* is not defined for j.

1.8 Translating frames into [formal] language

Example: $\mathfrak{f}$ = BEM



SatFor(BEM) = r ∈ person
∧ EYES(r) = x
∧ COLOR(x) = y ∧ y ∈ brown
∧ SEX(r) = z ∧ z = ♂

Equivalently r ∈ person ∧ COLOR(EYES(r))∈ brown ∧ SEX(r) = ♂

1.9 Satisfaction

Definition

Let ϕ be an open PL1 formula with $\{v_1, \dots, v_n\}$ the set of variables that occur free in ϕ, with v ∈ $\{v_1, \dots, v_n\}$. Then

a. '∃ ϕ' is the **existential closure of** ϕ

b. '$\exists_{\neq v}$ ϕ' is the existential closure of ϕ excepting v: $\exists v_1 \dots$ ~~∃v~~ $\dots \exists v_n$ ϕ

Definition 9 . Satisfaction condition for a node i in a Frame $\mathfrak{f}$

SatCon ($\mathfrak{f}$, i) $=_{df}$ '$\exists_{\neq i}$ SatFor($\mathfrak{f}$)'

Example: SatCon(BEM, r)
∃x∃y∃z (r ∈ person ∧ EYES(r) = x ∧ COLOR(x) = y ∧ y ∈ brown ∧ SEX(r) = z ∧ z = ♂)

1.9 Satisfaction

Definition 10 . Satisfaction class for a node i in a Frame $\mathfrak{f}$

a. If i is an individual constant,
SatCls($\mathfrak{f}$,i) $=_{df}$ {i} if SatCon ($\mathfrak{f}$, i) is true, SatCl($\mathfrak{f}$, i) $=_{df}$ Ø else

b. If i is an individual variable,
SatCls($\mathfrak{f}$, i) $=_{df}$ { i : SatCon($\mathfrak{f}$, i)}

Example:
SatCl(BEM, r) = { r : ∃x∃y∃z (r ∈ person ∧ EYES(r) = x ∧ COLOR(x) = y ∧ y ∈ brown
∧ SEX(r) = z ∧ z = ♂)

1.9 Satisfaction

Remark

For the trivial frame $\mathfrak{f}_t$,

SatCon($\mathfrak{f}_t$) = 't = t'. (Necessarily true)

If t is an individual constant, SatCls($\mathfrak{f}_t$) = { t }.

If t is an individual variable, SatCls($\mathfrak{f}_t$) = { t : t=t } = U.

For the trivial frame $\mathfrak{f}_{cl}$,

SatCon($\mathfrak{f}_{cl}$) = 'x ∈ cl'

SatCls($\mathfrak{f}_{cl}$) = { x : x ∈ cl } = cl.

1.10 Subsumption

Definition 11. Subsumption

Let $\mathfrak{f}_1 = \langle r_1, I_1, A_1, C_1, l_1, v_1 \rangle$ and $\mathfrak{f}_2 = \langle r_2, I_2, A_2, C_2, l_2, v_2 \rangle$ be two frames on the same ontology. Let i_1 be a node in $\mathfrak{f}_1$, and i_2 a node in $\mathfrak{f}_2$.

$\mathfrak{f}_1$ with respect to i_1 subsumes $\mathfrak{f}_2$ with respect to i_2,

$\mathfrak{f}_1(i_1) \sqsubseteq \mathfrak{f}_2(i_2)$ iff SatCls($\mathfrak{f}_1$, i_1) ⊆ SatCls($\mathfrak{f}_1$, i_1)

Equivalently iff SatFor($\mathfrak{f}_1$, i_1) ⇒ SatFor($\mathfrak{f}_1$, i_1)

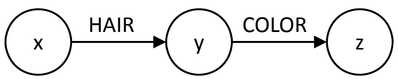
Remark

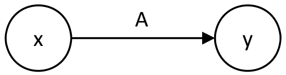
This is a semantic notion of subsumption.

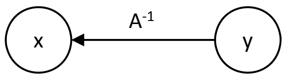
1.10 Subsumption

Remark

It takes a semantic notion of subsumption to capture the equivalence of pairs of frames such as the following:







1.11 Anchoring frames

Remark

If a node is labeled with an individual constant, it denotes a particular individual in the ontology. Such nodes are **fixed points** of the frame.

If i is a particular individual in U, and 'x' an individual variable, the following are equivalent:

Remark

If the referential node is a fixed point, the frame is an **instantiated frame.**

1.11 Anchoring frames

Definition 12
A frame is **uniquely anchored** iff there is exactly one way of assigning individual values to all its nodes labeled with variables. (If a frames is uniquely anchored, its satisfaction class contains exactly 1 element).

Observation
Not all fixed points lead to a unique anchoring of the frame.

Ontological question (2)
Given an ontological frame structure 𝔣 for which all indices are variables – for which (types of) nodes does replacement of the node label by an individual constant lead to unique anchoring?

Partial answer (1): If 𝔣 is a sortal frame and the referential node is labeled with an individual constant, the whole frame gets anchored, if it is satisfiable.

1.11 Anchoring frames

Partial answer (1)
A sortal frame is uniquely anchored if the referential node is a fixed point.

This is not a necessary condition.

Partial answer (2)
A sortal frame is uniquely anchored if the referential node is the value of an attribute (in the ontology) of a fixed point.

2. First-order comparators

Observation (1)
Sensory/cognitive systems all, even the most primitive ones, have the ability to register/recognize changes and differences in their environment. Comparators in these systems compare an input to a previous internal state in order to register changes; or they compare sensory input to stored contents. For example, if we categorize a perceptual input as belonging to/originating of something of a category 'cat', we have applied a comparator to stored category representations (concepts) and the object as presented by the perceptual system.

Observation (2)
A common form of denial in Japanese is to say “Chigau.” ('It's different')

2.1 First-order comparators

First-order comparators are two-place attributes. They assign a comparison outcome value to two individuals. Due to the definition of ontologies, the two arguments are necessarily of the same sort. Possible comparators depend on the sort they are applied to, in particular on the existence of orderings for the sort.

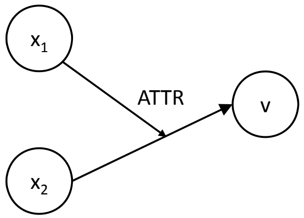
Definition 13: First-order comparator
For an ontological signature $\langle U, \mathcal{A}, \mathcal{S}, \mathcal{C} \rangle$, $s \in \mathcal{S}$, $\{true, false\} \in \mathcal{S}$.

a. For any sort s, there is the simple comparator COMP_s that returns the values $=$ and $\neq$.
b. if there is a linear ordering $<$ on s, $\text{COMP}_{s<}$ returns the values $<, =, >$
c. if there is a mereological ordering $\sqsubset$ on s, $\text{COMP}_{s\sqsubset}$ returns the values $\sqsubset, =, \sqsupset$

Remark
In the associated logic, we may write, e.g., 'x = y' instead of '$\text{COMP}_s(x, y)$ = '='.

2.1 First-order comparators

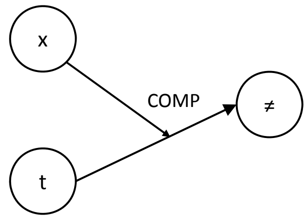
Graph notation for two-place attributes:
x_1 = 1st argument, x_2 = 2nd argument, v = value.



2.2 Frame-internal negation (1)

We can use the first-order comparator to model the negation of identity predications, predications meaning 'x ≠ t'.

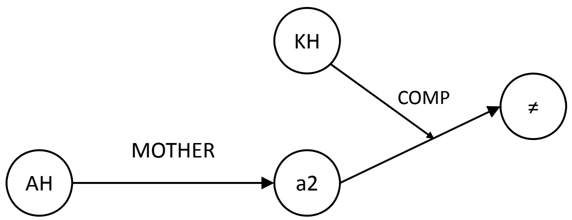
General pattern:



SatFor: COMP(x, t) = '≠' iff x ≠ t

2.2 Frame-internal negation (1)

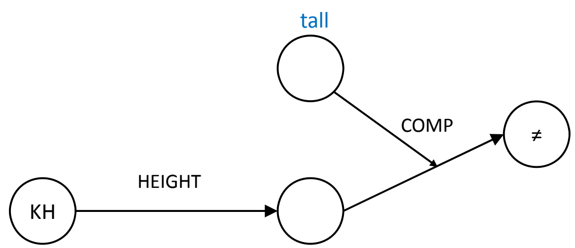
(2) *Katharine Hepburn is not Audrey Hepburn's mother.*



2.2 Frame-internal negation (1)

We can not use the first-order comparator to model the negation of predications meaning 'x ∉ c'

(3) *Katharine Hepburn is not tall.*



Meaning: There is a tall height that K. H. is not.

3. Tensed ontologies

We admit a sort **time** in the universe of the ontology.
Times can be thought of as intervals on the time axis.

Definition 14. Orderings of times

a. There is a partial linear ordering '<' = 'earlier', the chronological ordering, which gives rise to the following relations:
db (detached, before), ib (immediately before),
= , ia (immediately after), da (detached after).
Corresponding comparator: $\text{COMP}_{t,<}$

b. There is a partial mereological ordering of subintervalship with the alternative values ⊏, =, ⊐.
Corresponding comparator: $\text{COMP}_{t,\sqsubset}$

3.1 Tensed ontologies

Definition 15. Tensed ontology
⟨U, $\mathcal{A}$, $\mathcal{S}$, $\mathcal{C}$ ⟩ is a tensed ontology if

a. there is a sort Time in $\mathcal{S}$, such that Time is a set of times (i.e. time intervals); there is a mereological and a chronological partial ordering on Time.

b. there are two-place attributes a_t in $\mathcal{A}$ that take times as their first argument:
a_t: Time × U ⟶ U

c. **Homogeneity**:
if a is an attribute that takes times as arguments and assigns a value y to time t, then a assigns the same value y to all times t' with t' ⊑ t.

Remark
Analogous homogeneity requirements are in order for other domains.

3.1 Tensed ontologies

Remark
In a realistic ontology, most attributes will be tensed. There are, however, attributes whose values are not time-dependent, e.g. attributes specifying the origin of an object (producer, parents) or attributes specifying once-only events (birth, death).

Ontological question (3)
Which (types of) attributes in the natural ontology take values that change in time?

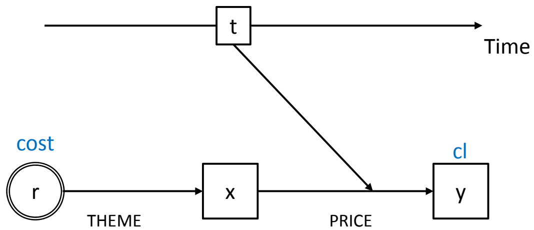
3.1 Tensed ontologies

Definition 15. Tensed ontology (ctd.)

d. There is a sort E of events/eventualities in $\mathcal{S}$.
There are three attributes defined on the sort of events, assigning times to events:
τ: E ⟶ Time returns the time that the event occupies

3.3 States: Stative dimensional verbs

"at time t, x costs cl"

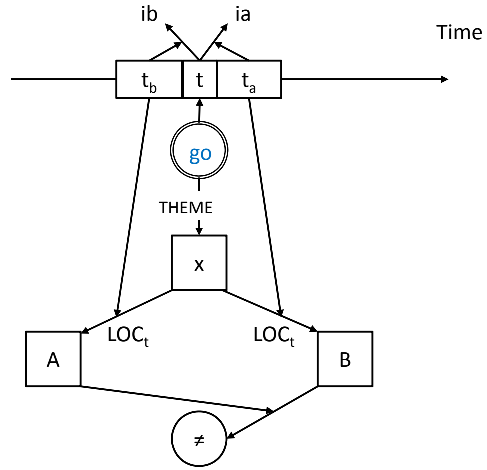


SatFor(cost): PRICE(t, THEME(r)) ∈ cl

3.3 Simple changes

"go x from A to B"

SatFor:
$\tau(e) = t$
t_b ib t
t_a ia t
THEME(e) = x
LOC(t_b, x) = A
LOC(t_a, x) = B
A ≠ B

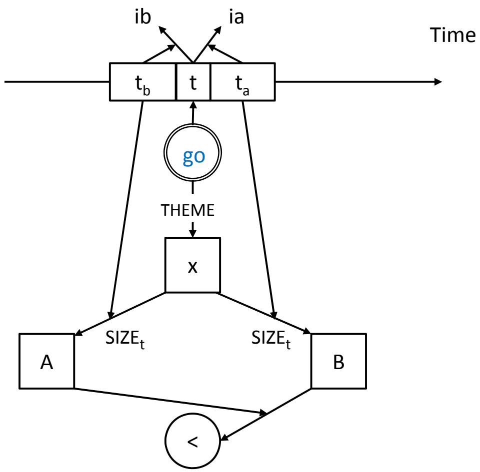


3.3 Degree achievements

"x grow"

Degree achievements in general:

SatFor:
$\tau(e) = t$
t_b ib t
t_a ia t
THEME(e) = x
SIZE(t_b, x) = y
SIZE(t_a, x) = z
y < z



4. Second-order comparator ©

4.1 The basic definition

The second-order comparator compares the conditions in two frames on two nodes. It returns the truth value 'true' iff the information of the first frame about the first node subsumes he information of the second frame about the second node.

Definition 16 : The second-order 'big' comparator

Let $\mathfrak{f}_1$and $\mathfrak{f}_2$ be two frames on the same ontology.
Let $i_1/\mathfrak{f}_1$ be a node in $\mathfrak{f}_1$, $i_2/\mathfrak{f}_2$ be a node in $\mathfrak{f}_2$.

©($i_1/\mathfrak{f}_1$, $i_2/\mathfrak{f}_2$)	$=_{df}$	true	if $\mathfrak{f}_1(i_1) \sqsubseteq \mathfrak{f}_2(i_2)$
		false	else
©($i_1/\mathfrak{f}_1$, $i_2/\mathfrak{f}_2$)	$=_{df}$	true	if SatCls($\mathfrak{f}_1$, i_1) ⊆ SatCls($\mathfrak{f}_2$, i_2)
		false	else

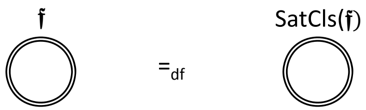
4.1 The basic definition

Simplified notations for comparisons with trivial frames

©(x, i/$\mathfrak{f}$) $=_{df}$ ©(x/$\mathfrak{f}_x$, i/$\mathfrak{f}$)

©(i/$\mathfrak{f}$, cl) $=_{df}$ ©(i/$\mathfrak{f}$, y/$\mathfrak{f}_{cl}$) for some appropriate variable 'y'

©(x, cl) $=_{df}$ ©(x/$\mathfrak{f}_x$, y/$\mathfrak{f}_{cl}$) for some appropriate variable 'y'



4.2 World frames

Definition 17 : World frames
A world frame is a giant frame on a given ontology that contains all known/assumed information on the “world” as it can be captured in the notions of the ontology.

Remark
It is assumed that the world knowledge (shared, or individual) can be represented in a giant connected frame.

4.3 Truth and falsity

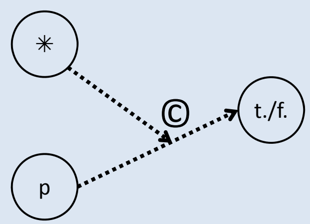
- Truth and falsity with respect to a given world frame can be modeled *within* frame theory by applying the Comparator to an appropriate node ✳ in the world frame and the central node of a proposition.

Crucial question
How are we to determine the relevant node of comparison, the anchor node, in the world frame used? This requires a theory of linking utterances and their reference to the world, including a time. Observing that *utterances are events in the world frame* will be the point of departure for anchoring.

4.3 Truth and falsity

Definition 18. Truth and falsity in the world
Let $\mathfrak{f}_p$ be a frame ⟨r, I, A, C, l, v⟩ on the ontology ⟨U, $\mathcal{S}$, $\mathcal{C}$, $\mathcal{A}$ ⟩ that represents a proposition. Let $\mathfrak{W}$ be a world frame on the same ontology, ✳ a node in $\mathfrak{W}$.

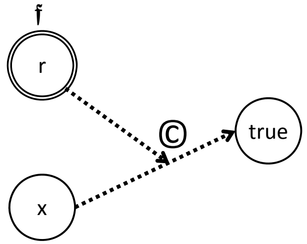
p is true [false] related to the node ✳ in $\mathfrak{W}$ iff ©(✳/$\mathfrak{W}$, p/$\mathfrak{f}_p$) = true [false].



4.4 Frame satisfaction and subsumption

An individual x in the ontology **satisfies** a frame $\mathfrak{f} = \langle r, I, A, C, l, v\rangle$ iff it compares positively with it .

General pattern

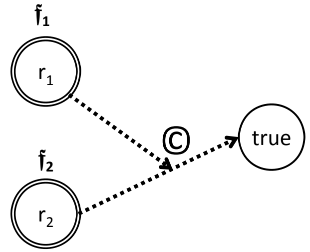


©(x, r/$\mathfrak{f}$) = true

4.4 Frame satisfaction and subsumption

$\mathfrak{f}_1 = \langle r_1, I_1, A_1, C_1, l_1, v_1\rangle$ subsumes $\mathfrak{f}_2 = \langle r_2, I_2, A_2, C_2, l_2, v_2\rangle$ iff it compares positively with it

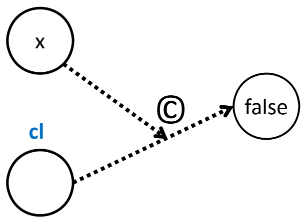
General pattern



4.5 Frame-internal negation (2)

Negation of a predication 'x ∈ cl'

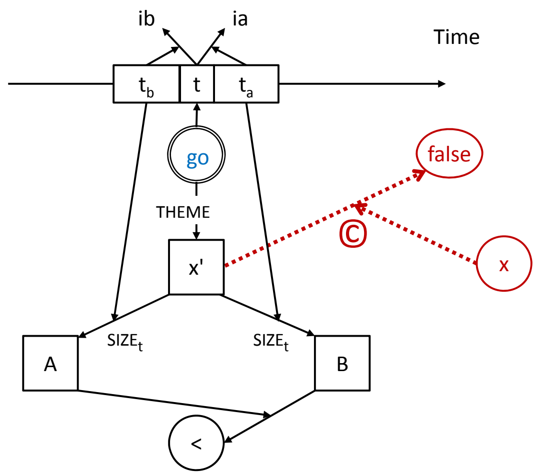
General pattern: ©(x, cl) = false



SatFor: $\mathfrak{f}_x \not\sqsubseteq \mathfrak{f}_{cl}$, iff $x \notin cl$

4.5 Frame-internal negation (2)

"x not grow"



5. Ontologies with space

We admit a sort **space** in the universe of the ontology.
Times can be thought of as regions in 3-dimensional space.

Definition 19. Orderings of spaces
There is a partial topological ordering with the alternative values $\subset$, =, $\supset$.
Corresponding comparator: $\text{COMP}_{\text{space},\subset}$

Space related attributes:

SPACE: U ⟶ space	SPACE(x) is the region in space occupied by x
ON: U ⟶ space	ON(x) is the region in space immediately above SPACE(x)
IN: U ⟶ space	IN(x) is the interior of SPACE(x)
etc.	

5. Ontologies with space

Example:

cup x is on table y