

IL MODELLO A OGGETTI IN IA

Premessa:

- Concetto di *oggetto* riconducibile a settori dell'informatica sviluppatasi storicamente in modo alquanto disgiunto e indipendente:
 - linguaggi di programmazione
 - basi di dati
 - intelligenza artificiale
- Necessità di una migliore **integrazione** tra sottosistemi diversi (basi di dati, pacchetti applicativi, reti di comunicazione, supervisor intelligenti)
- Modello ad oggetti come fattore di **aggregazione** sia a livello concettuale sia realizzativo
- Come perseguire questa integrazione è ancora problema aperto
- Esiste un insieme di **concetti comuni** per dare una caratterizzazione largamente accettabile del paradigma ad oggetti?
- Importanza dei formalismi per Semantic Web

CONCETTO DI OGGETTO

- **Oggetto** = "Knowledge grain"

per rappresentare **concetti** del mondo reale

*A knowledge representation framework should not merely prescribe how small bits of information ought be represented, but it should also detail how the totality of information ought to be **structured** and organized so that information may be **retrieved** and relevant **inferences** may be drawn **efficiently** (Shastri, 1989)*

OGGETTI IN IA

- *Strutturazione*
- *Recupero*
- *Inferenza*
 - *Classificazione*
 - *Ereditarietà*
 - *Riconoscimento*
- *Efficienza*

RAPPRESENTAZIONE A OGGETTI VS. RAPPRESENTAZIONE RELAZIONALE

Dipendenza dal modo con cui si rappresentano i dati

- **Rappresentazione relazionale: dati raggruppati concettualmente attraverso le loro proprietà**
 - `person (john)`
 - `job (john, employee)`
 - `project (john, cnr1234)`
- **Rappresentazione ad oggetti: i dati sono concettualmente tutti raggruppati all'interno dello stesso oggetto**

```
object    john: is-a person
          job  -> employee
          project -> cnr1234
```

RAPPRESENTAZIONE A OGGETTI VS. RAPPRESENTAZIONE RELAZIONALE

- L'utente può accedere a *tutte* le informazioni di un oggetto mediante l'identificatore unico (logico o fisico) dell'oggetto stesso
- Si possono individuare altre proprietà (opzionali).
- Anche se storicamente, i linguaggi ad oggetti adottano uno stile di programmazione procedurale, i *paradimi di rappresentazione* (dichiarativo, procedurale) e *programmazione* (funzionale, imperativo, dichiarativo) possono essere concepiti come *ortogonali* rispetto al concetto di sistema "*object-oriented*"

APPROCCI

- **Approccio Logico o Dichiarativo:**
 - Usato nelle reti semantiche, nella logica terminologica o descrittiva (famiglia KL-ONE).
 - Un oggetto è una congiunzione di **proprietà** e può essere rappresentato come clausole della logica
- **Approccio Procedurale:**
 - Usato nei sistemi di rappresentazione della conoscenza basati su *frames* e nei linguaggi di programmazione ad oggetti
 - Un oggetto è una struttura dati con uno **stato** e un **comportamento** (behaviour)
 - Le proprietà di un oggetto possono essere attributi, relazioni e procedure (slots, subslots, tipi, cardinalità, vincoli)
 - Conoscenza procedurale espressa mediante demoni (attaccati agli attributi) e metodi (attaccati agli oggetti)

APPROCCI Dichiarativi vs Procedurali

- Reti semantiche
- Linguaggi terminologici (logiche descrittive)

- Frames

APPROCCI Dichiarativi

- Reti semantiche
- Logiche terminologiche (logiche descrittive)
- Frames

RETI SEMANTICHE

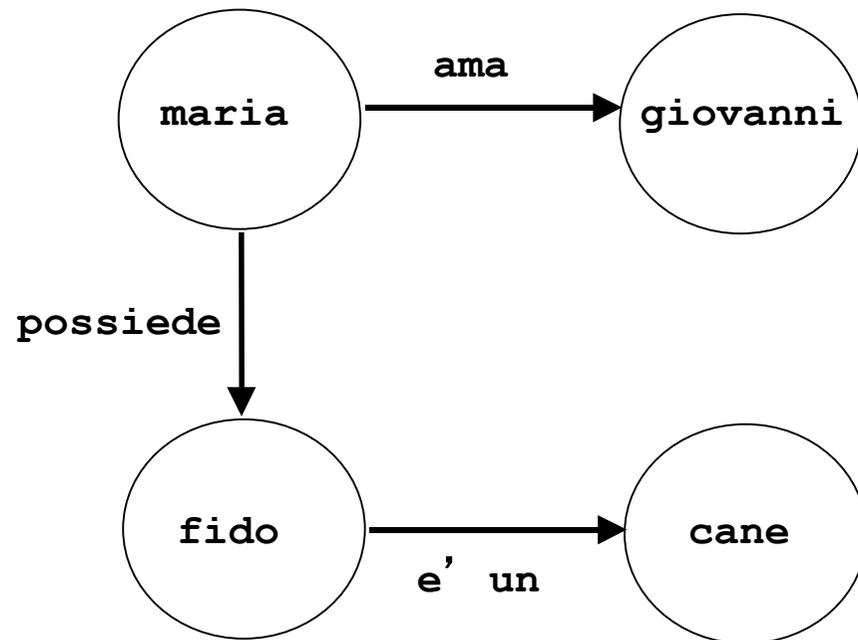
(Brachman, 1979)

- Non sono di chiara origine e definizione;
- Particolarmente utili per il linguaggio naturale, descrizione e comprensione di forme, elaborazione di modelli psico-cognitivi.
- Oggetto o **concetto** rappresentato da un nodo
- Relazioni gerarchiche (*is-a*, *instance-of*)

- Valori di proprietà rappresentati come nodi legati attraverso archi etichettati

- **CONOSCENZA: OGGETTI E RELAZIONI BINARIE**
 - OGGETTI = NODI di un grafo diretto etichettati;
 - RELAZIONI = ARCHI etichettati con una direzione per evitare ambiguità

ESEMPIO



CARATTERISTICHE

- La duplicazione di nodi che rappresentano lo stesso concetto viene sempre evitata.
- NON RIDONDANZA:
 - NODI: singola locazione di memoria
 - ARCHI: strade concettuali, ma anche fisiche (puntatori), per raggiungere in modo efficiente i concetti interessati.
- **Rappresentazione = Conoscenza + Metodi di Accesso** (Newell)
- Corrispondono a una LOGICA DEI PREDICATI CON RELAZIONI BINARIE (senza duplicazione)
 - Rappresentazione binaria:
 - modificabilità;
 - modularità.

ESEMPIO

“giovanni dà un libro a maria”.

- **LOGICA DEI PREDICATI:**

- a1) dà(`giovanni`, `libro`, `maria`);

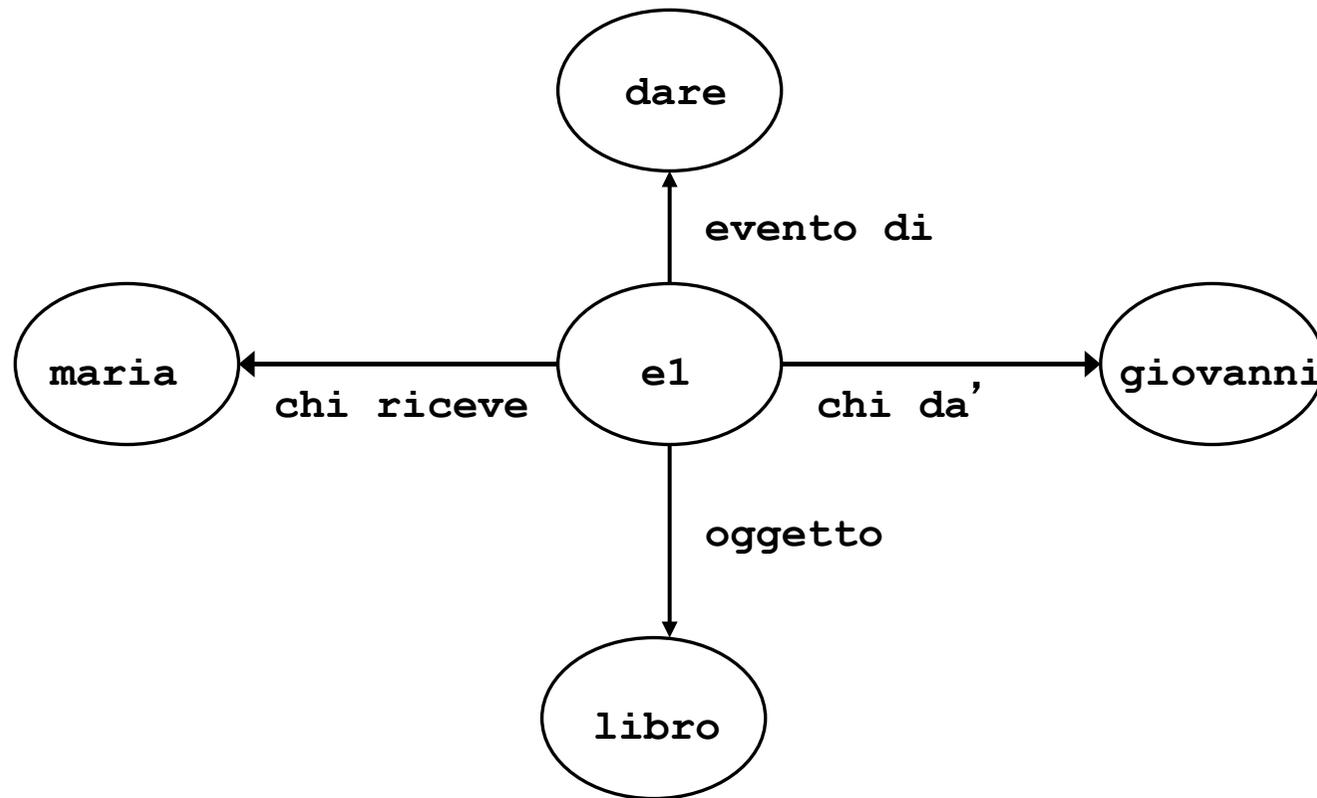
- a1 può essere considerato come un evento `e1` con le seguenti caratteristiche:

- a2) evento (`e1`, `dare`, `giovanni`, `libro`, `maria`);

- oppure alternativamente:

- a3) `evento(e1,dare) and chi-da(e1,giovanni) and`
 `chi-riceve(e1,maria) and oggetto(e1,libro)`

RETI SEMANTICHE



RETI SEMANTICHE

- **ESPRIMONO IN MODO IMPLICITO REGOLE PER:**
- Trattare concetti di EREDITARIETÀ (semplice o multipla) di proprietà;
- Permettere un ricoprimento di proprietà ritenute erranee o aggiornabili in base a nuova conoscenza acquisita (ECCEZIONI E DEFAULT).
- (semantica?).

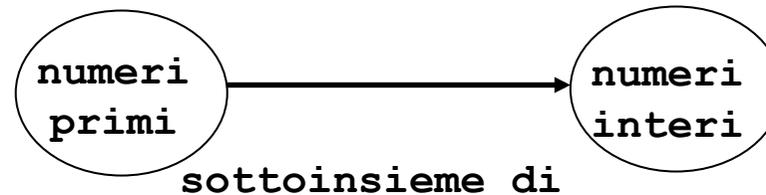
NODI

- NODI: possono rappresentare diverse entità semantiche

- **single entità individuali.**



- **insiemi di entità.**

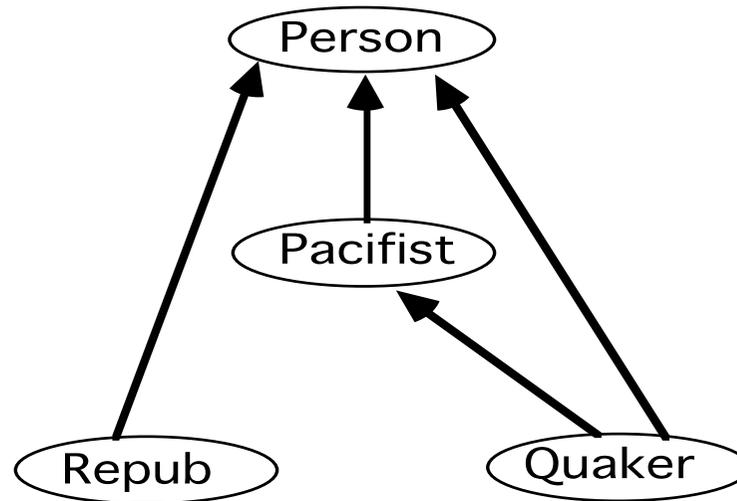


- Netta distinzione fra questi differenti tipi di nodi per ottenere maggiore chiarezza nella notazione.

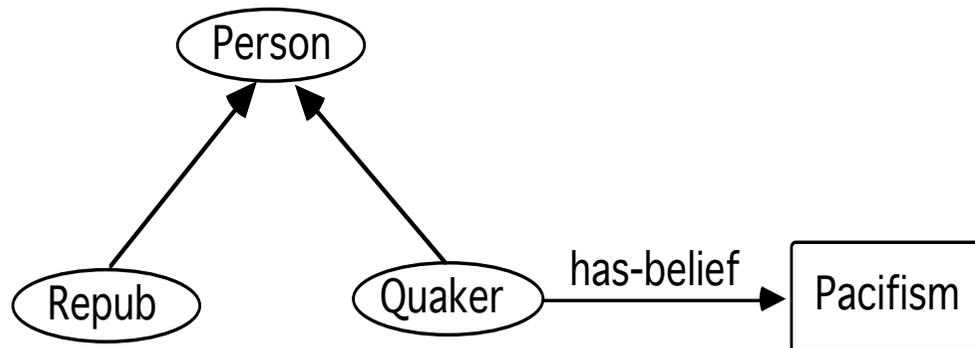
NODI

Sistemi :

- “di sole classi” (e istanze):



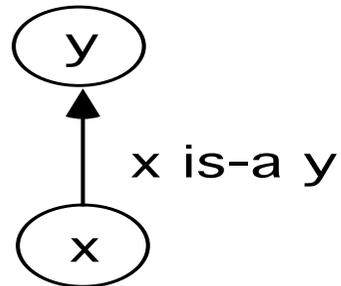
- “di classi e proprietà”:



- Rappresentabili in Logica dei Predicati del I Ordine

TRASLAZIONE IN LOGICA

- Se x è un individuo e y una classe il legame:



- Può essere interpretato come la formula logica:

$y(x)$

Es: *cane*(fred)

- Se x e y sono classi, il legame tra loro può essere interpretato come la formula logica:

$\forall z \ x(z) \Rightarrow y(z)$

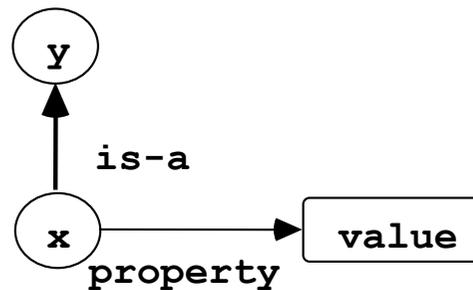
Es: $\forall z \ \textit{cane}(z) \Rightarrow \textit{mammifero}(z)$

TRASLAZIONE IN LOGICA

- Se una classe o un individuo ha delle proprietà, queste possono essere traslate in predicati binari

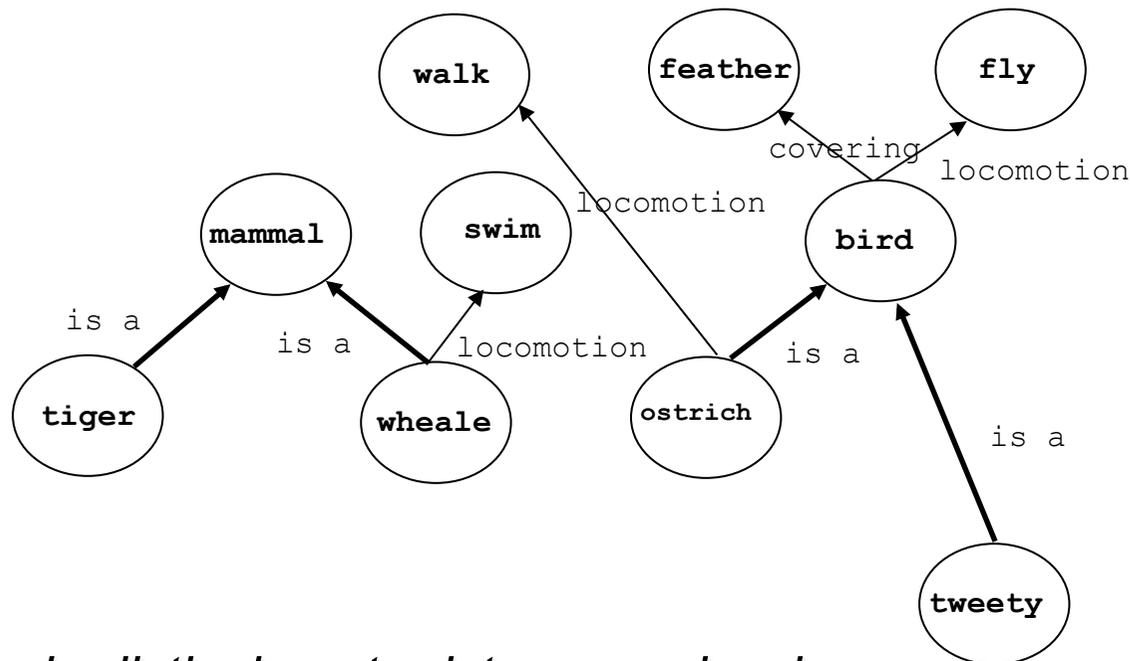
$\forall Z \ x(Z) \Rightarrow \text{property}(Z, \text{value})$
 $\text{property}(x, \text{value})$

classe
individuo



$\forall Z \ \text{cane}(Z) \Rightarrow \text{ha_zampe}(Z, 4)$
 $\text{ha_zampe}(\text{fido}, 4)$

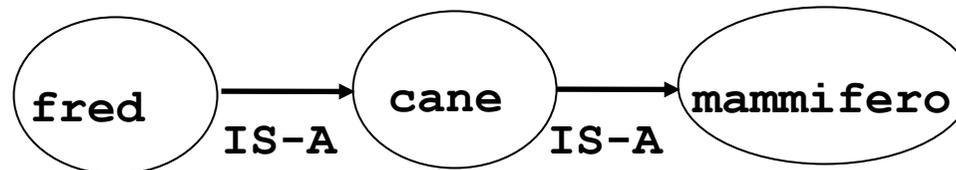
PROBLEMI DELLE RETI SEMANTICHE



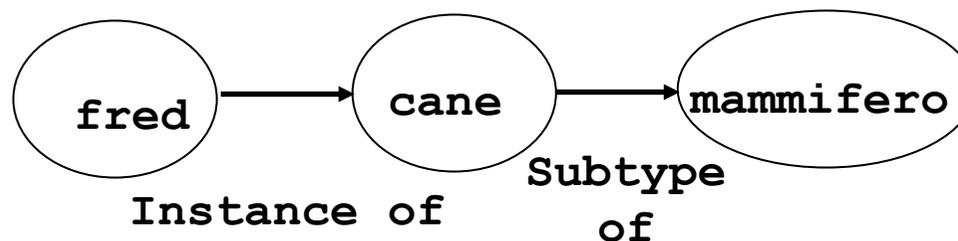
- *Manca la distinzione tra istanze e classi*
 - In realtà, l'unico link “is-a” è “sovraccaricato” (Woods, 1975)
 - Rappresenta sia la relazione di *appartenenza* sia quella di *sottoinsieme*.
 - Richiedono la presenza di un secondo link “instance-of” (ma anche *part-of*, etc.)

PROBLEMI DELLE RETI SEMANTICHE

- IS-A



- "fred è un cane"
- "il cane è un mammifero".



- Due significati diversi:
 - 1) member-of; instance-of;
 - 2) Is-a; subset-of; subtype-of.
- Non esiste distinzione tra attributi associati a una classe e attributi (ereditati) dalle istanze della classe

ATTRIBUTI DI CLASSI E ISTANZE

Esempio:

- *Gli elefanti sono una specie in via di estinzione*
- *Clyde è un elefante*
- *Clyde è in via di estinzione*

- *Gli elefanti sono numerosi*
- *Clyde è un elefante*
- *Clyde è numeroso*

Semantica formale:

- Manca una semantica formale (almeno nelle prime versioni)
- Aumento del potere espressivo e semantica:
 - **grafi concettuali** (Sowa, 1984)
 - famiglie dei **linguaggi terminologici** (alla KL-ONE)

APPROCCI dichiarativi

- Reti semantiche
- Logiche terminologiche (logiche descrittive)

- Frames

Logiche terminologiche (LT) o descrittive (Description Logic, DL)

Possono essere viste come:

1. *Evoluzioni “logiche”* di linguaggi di rappresentazione della conoscenza basati sulle nozioni di *frame* e *reti semantiche*
2. *Contrazioni* della logica del prim' ordine (FOL) per ottenere migliori proprietà computazionali
 - Verso gli anni 80 si ha una sterzata verso la logica delle reti semantiche
 - Il processo di formalizzazione consiste nel ...
 - riformulare i costrutti secondo i canoni della logica
 - eliminare i costrutti che non si prestano a tale riformulazione (*default* ed eccezioni)

Concetti e ruoli

- Ogni LT è caratterizzata da operatori per la costruzione di termini di due tipi:
 - *concetti* (corrispondenti a relazioni unarie)
con operatori *and*, *or*, *not*, *all*, *some*, *atleast*, *atmost*, ... per la costruzione di concetti complessi
 - *ruoli* (corrispondenti a relazioni binarie)

Esempio

La seguente è una formula di una delle LT:

(and paper
 (atmost 2 author)
 (atleast 2 author))[paper372]

equivalente a:

paper(paper372) \wedge
 $\exists x$ author(paper372, x) \wedge
 $\exists y$ author(paper372, y) $\wedge x \neq y \wedge$
author(paper372, z) $\Rightarrow (z = x) \vee (z = y)$

Linguaggi terminologici (famiglia KL-ONE)

- (Brachman, Schmolze, 1985)
- "**Linguaggi terminologici**" è un nome collettivo per formalismi di rappresentazione della conoscenza che si concentrano sulla gestione di entità intensionali (**concetti**)
- Alcuni sistemi esistenti: BACK, CANDIDE, CLASSIC, KRIS, KRYPTON, LOOM, NIKL, YAK
- Distinguono tra **aspetti terminologici** (TBox, descrizioni di concetti) e **asserzionali** (ABox, istanze) nella rappresentazione della conoscenza
- **Descrivono** la **struttura** di un oggetto a livello terminologico mediante **concetti** e **ruoli** .

Linguaggi terminologici (famiglia KL-ONE)

- **Concetto:** Descrizione con una struttura potenzialmente complessa formato mediante un insieme limitato di operatori
- Corrispondono a predicati unari (applicati a singoli individui)
- **Ruoli:** Termini per rappresentare proprietà
- Corrispondono a predicati binari (correlano due individui per volta)
- Ruoli a singolo valore sono detti **attributi**

Linguaggi terminologici (famiglia KL-ONE)

- Il potere espressivo aggiunto dei linguaggi della famiglia KL-ONE viene dalla possibilità di esprimere **concetti composti** e di organizzare **automaticamente** questi in gerarchie sulla base della loro struttura (**classificazione**)
- **Concetto composto**, costruito a partire da altri concetti (composti o primitivi), mediante **operatori**
- I diversi linguaggi si differenziano per il loro vocabolario di operatori terminologici, nella separazione fra KB terminologica e asserzionale e nell'algoritmo di classificazione

Linguaggi terminologici (famiglia KL-ONE)

- I linguaggi terminologici possono essere traslati in logica, ma perdono le loro caratteristiche fondamentali (**strutturazione** e semantica diretta)
- **Terminologia:**
- Corrisponde alla parte TBox
- Insieme di definizioni di concetti

Woman \subseteq **Person**

Parent \equiv **Person** with some **child**

Mother \equiv **Parent** and **Woman**

- Proprietà necessarie e sufficienti a definire un concetto
- Anche chiamati Frame-based Description Languages, Term Subsumption Languages, Terminological Knowledge, Taxonomic Logics o Description Logics

APPROCCI procedurali

- Reti semantiche
- Logiche terminologiche (logiche descrittive)

- Frames

FRAMES

- Molto simili al formalismo delle reti semantiche
- Utilizzati assieme a regole in gran parte degli ambienti evoluti per sistemi esperti (*tools*)
- Introdotti per la prima volta da Minsky nel 1975 (Minsky, 1975)
 - *"Quando si incontra una nuova situazione (o si fa un sostanziale cambiamento nel modo di concepire un certo problema) si seleziona dalla memoria una struttura chiamata **frame** che è uno **schema concettuale** precedentemente memorizzato che viene adattato per rappresentare la realtà ..."*
 - *"Un **frame** è una struttura dati per rappresentare una **situazione stereotipata**, come essere in un certo tipo di camera o andare al compleanno di un bambino. **Attaccate a ogni frame** ci sono vari tipi di **informazioni** . Alcune sono relative a **come utilizzare il frame**, altre a cosa può richiamare in seguito ..."*

FRAMES

Concetto intuitivo: Modulo di conoscenza che descrive qualcosa in termini delle sue *proprietà*

- Ha un *nome* che identifica ciò che è descritto
- Ognuna delle proprietà possedute è rappresentata mediante una coppia *slot/valore* (KRL, NETL, KEE, KAPPA, ART)
- *Esempio:*

Nome	Slot	Valore (<i>default</i>)
Volpe	Is-a	Piccolo-Animale
	Colore	Fulvo
	Furbizia	Molto-sviluppata
Volpe_albina	Is-a	Volpe
	Colore	Bianco

FRAMES

- Differentemente dai linguaggi terminologici, l'**appartenenza a una categoria** (classe o concetto) non viene caratterizzata mediante proprietà **necessarie e sufficienti**, ma nei termini di una maggiore o minore **somiglianza** rispetto a membri tipici della categoria detti **prototipi**
- In generale, tutti i sistemi a frame permettono di ragionare su classi di oggetti usando **rappresentazioni prototipali** che, valide in linea di massima, devono essere adattate e modificate (**eccezioni**)
- Le descrizioni dei prototipi non sono nemmeno condizioni necessarie
- I valori per **default** vengono usati come valori **provvisori e approssimativi**, in assenza dei valori reali
- Come nelle reti semantiche, le **connessioni** fra un frame e gli altri e i rispettivi ruoli che devono rappresentare nella conoscenza globale sono determinati da **valori attribuiti a specifici slots**.

SUBSLOT O FACET

- Gli slot possono contenere *informazioni aggiuntive* sui valori associati (default, tipo, quantità, etc.)
- *Informazioni procedurali*:
 - Connesse anch' esse agli slot del frame (*procedural-attachment*)
 - Indicano come utilizzare tale frame, come trovare i valori per gli slots, etc.
- *Integrazione di conoscenza dichiarativa e procedurale*
 - Le procedure (*demoni*) sono spesso individuate da particolari “facet” che attivano la procedura associata durante:

– la ricerca di valori di uno slot	if-needed
– l'aggiunta di valori	if-added
– la cancellazione di valori	if-removed

ESEMPIO

Nome	Attributo	Facet	Valore
• Resistenza	Is-a	Uguale	Componente
	Valore	Unit	Ohm
	Precisione	Dominio	(1 5 10)
		Default	5
	Potenza	If-needed	Request

Istanza di nome R1:

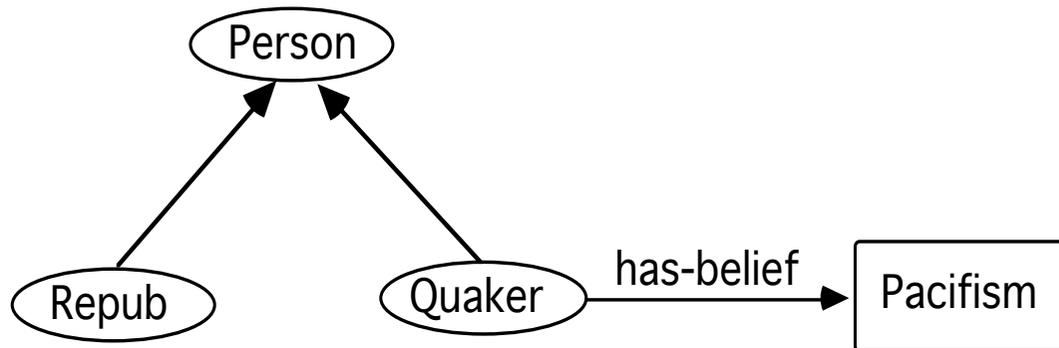
Nome	Attributo	Facet	Valore
• R1	Is-a	Uguale	Resistenza
	Valore	Uguale	5000
	Precisione	Uguale	1

OGGETTI IN IA

- *Strutturazione*
- *Recupero*
- *Inferenza*
 - *Classificazione*
 - *Ereditarietà*
 - *Riconoscimento*
- *Efficienza*

OGGETTI IN IA

Reti semantiche:



OGGETTI IN IA

Frames:

Nome	Attributo	Facet	Valore
Resistenza	Is-a	Uguale	Componente
	Valore	Unit	Ohm
	Precisione	Dominio	(1 5 10)
		Default	5
	Potenza	If-needed	Request

Istanza di nome R1:

Nome	Attributo	Facet	Valore
R1	Is-a	Uguale	Resistenza
		Valore	Uguale 5000
		Precisione	Uguale 1

OGGETTI IN IA

Linguaggi terminologici (Description Logics):

Woman \subseteq Person

Parent \equiv Person with some child

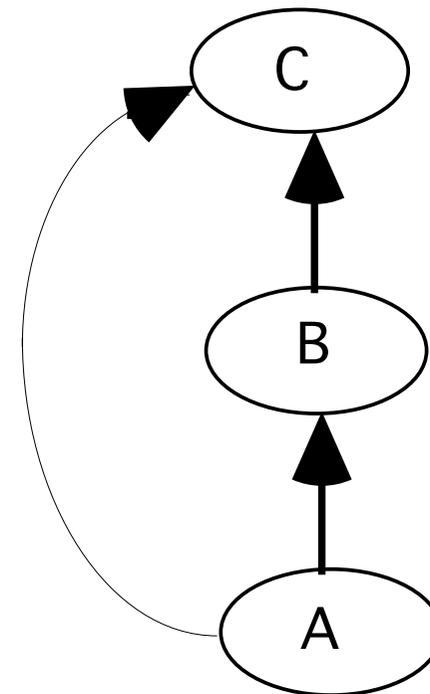
Mother \equiv Parent and Woman

ORGANIZZAZIONE DI OGGETTI E INFERENZE

- Gli oggetti vengono organizzati in una *gerarchia* o *ordinamento parziale* detta *gerarchia di ereditarietà* o *tassonomia*
- *Oggetti* rappresentati da nodi
- *Relazioni gerarchiche* tra oggetti rappresentate connettendo nodi via archi “is-a” o “instance-of”
- Cattura il principio base di organizzazione sia delle reti semantiche, dei sistemi a frames e dei sistemi a oggetti
- La *tassonomia* viene utilizzata per memorizzare informazioni al livello più appropriato di generalità, rendendole automaticamente disponibili agli oggetti più specifici mediante il meccanismo di *ereditarietà*

TASSONOMIA

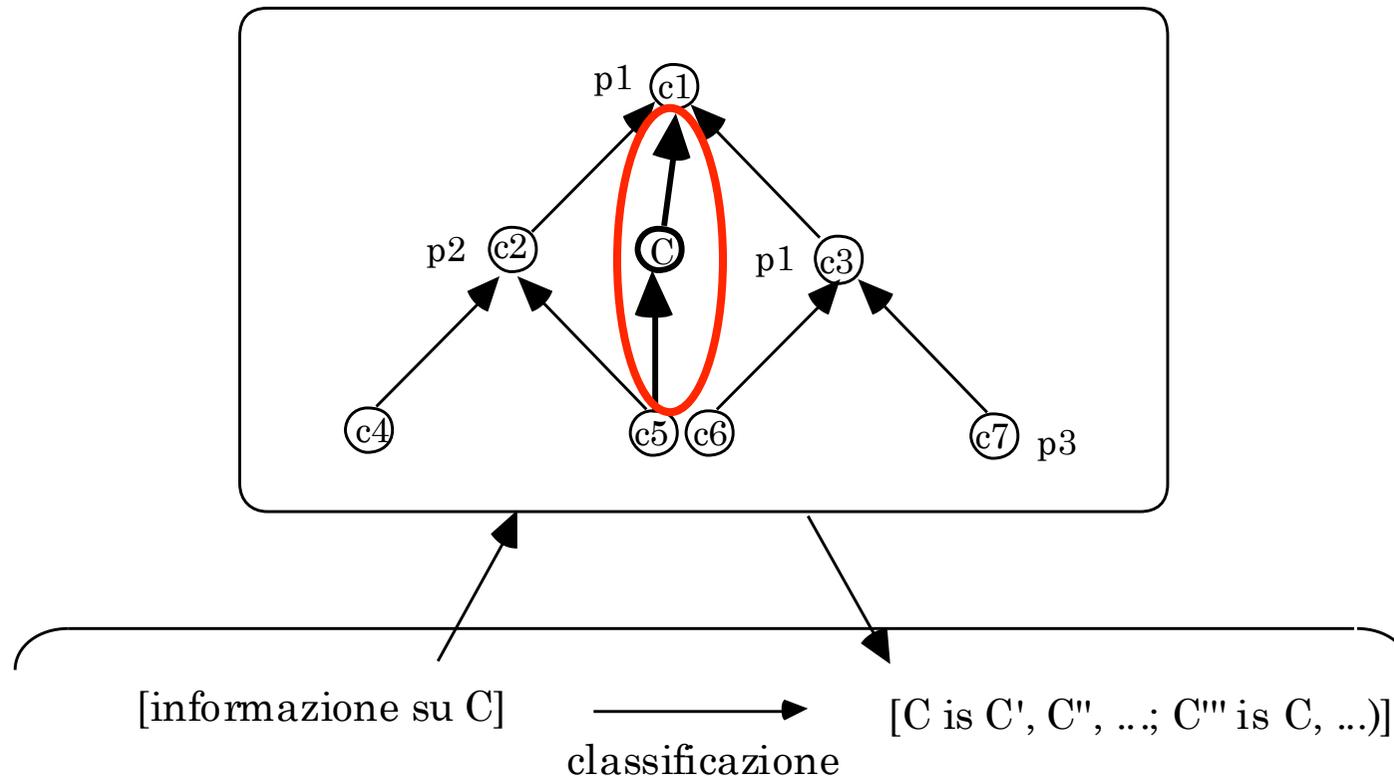
- La *tassonomia* può essere *costruita*:
 - *direttamente* dal programmatore, asserendo i legami “is-a” tra coppie di concetti (reti semantiche, sistemi a frames e a oggetti)
 - *automaticamente* inferendo i legami “is-a” tra coppie di concetti (ad esempio a partire dalla loro struttura nella famiglia KL-ONE dei linguaggi terminologici):
 - **Parent = Person** with some **child**
- Costruita la tassonomia, interrogazioni sui legami gerarchici tra oggetti presenti nella tassonomia sono risolte attraverso l'applicazione della chiusura transitiva



FORME DI INFERENZA

- **Tre tipi di inferenze:**
 - *classificazione*
 - *riconoscimento*
 - *ereditarietà*

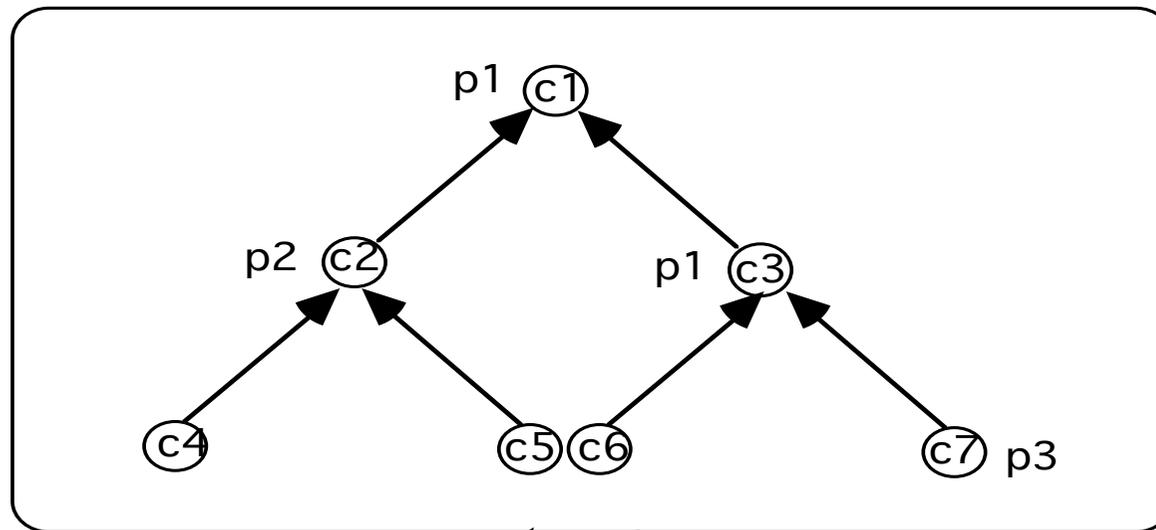
CLASSIFICAZIONE



CLASSIFICAZIONE

- Inserisce un concetto nella tassonomia al livello più appropriato (linguaggi terminologici)
- Utilizza il **test di sussunzione**
- L' algoritmo di classificazione di un nuovo oggetto (concetto) C consiste di due parti:
 - **most specific subsumers**, per individuare i concetti più specifici che sussumono C (nell' esempio c1)
 - **most general subsumees**, per individuare i concetti più generali sussunti da C (nell' esempio c5)
- Richiede che le **classi siano descritte da proprietà necessarie e sufficienti** (definizioni “se e solo se”, possibile solo nei linguaggi terminologici)

RICONOSCIMENTO



[informazione
parziale su X]

—————>
riconosc.

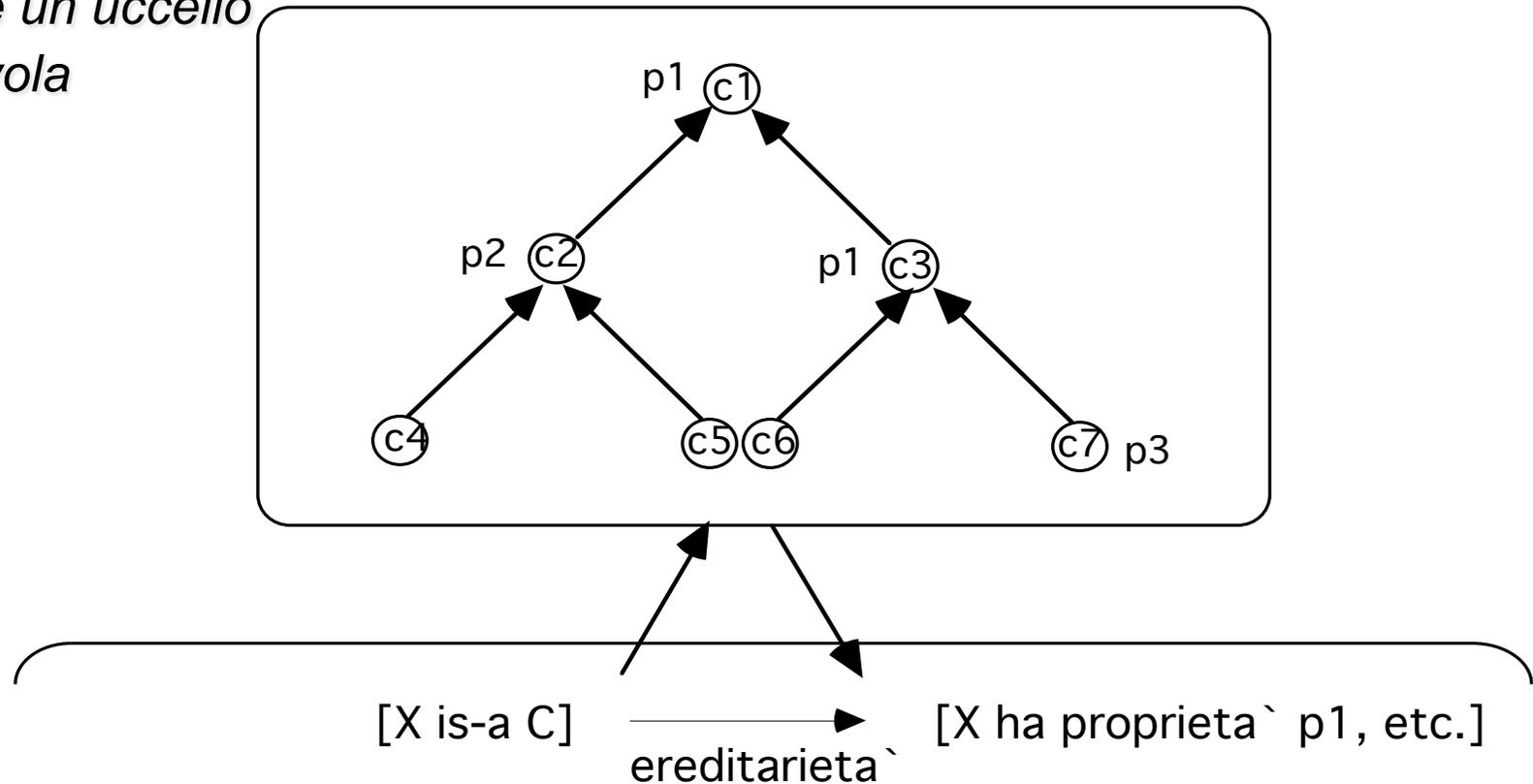
[X is-a C]

RICONOSCIMENTO

- Data una tassonomia, è l'operazione con cui si "classificano" individui cioè istanze (linguaggi terminologici)
- Data una descrizione che consiste di un insieme di proprietà, trovare un concetto che *meglio* corrisponde alla descrizione data
- Inserimento di un individuo sotto i suoi *most specific subsumers* (prima parte dell' algoritmo di classificazione)
- Inferenza corretta nel caso di descrizioni complete (*necessarie e sufficienti*)

EREDITARIETA'

- È una forma di inferenza che consente di determinare le proprietà di un oggetto basandosi sulle proprietà dei suoi *antenati*
- *Gli uccelli volano*
- *Titti è un uccello*
- *Titti vola*

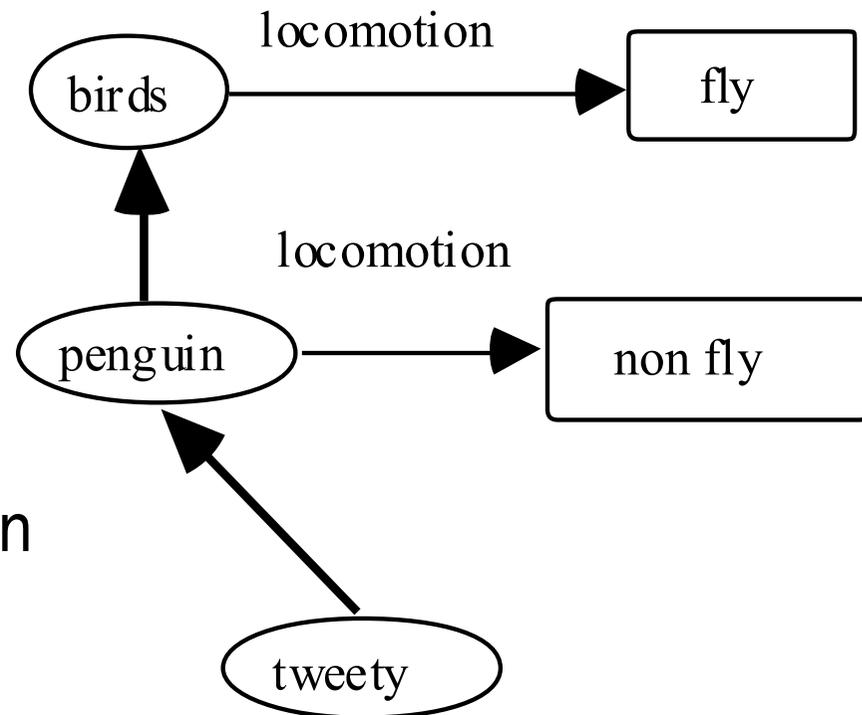
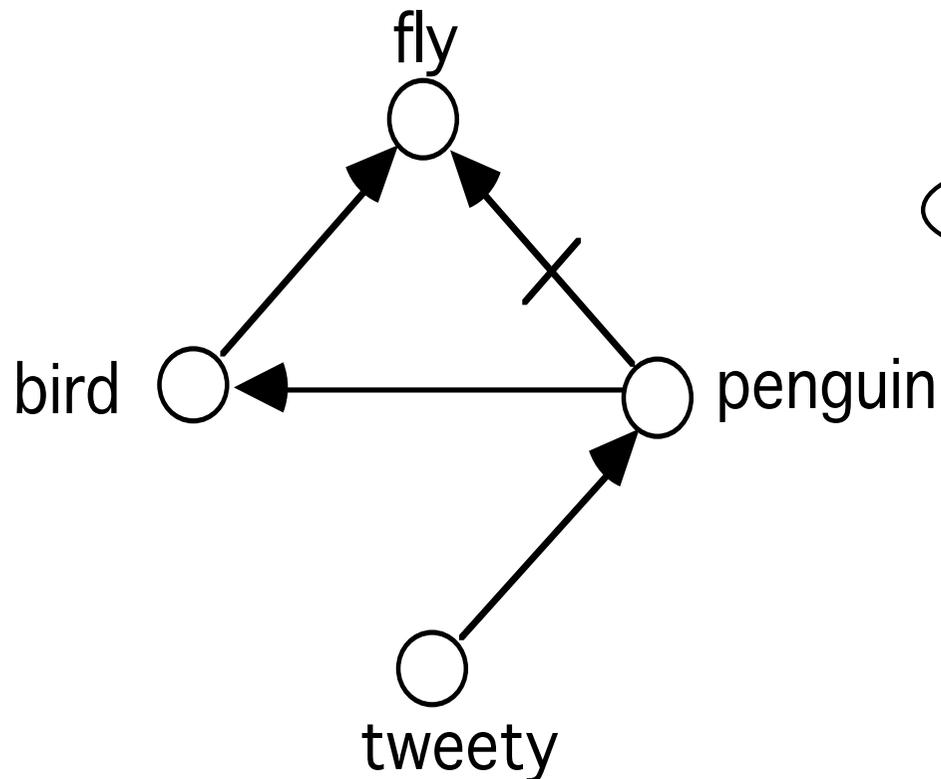


EREDITARIETA'

- Molto spesso ereditarietà, classificazione, riconoscimento sono combinati tra loro
- Presenza di proprietà in conflitto (eccezioni, ereditarietà multipla)
- Molti sistemi sono *non-monotoni*
- *Non-monotonicità:*
- Un sistema di ereditarietà è *non monotono* se ammette archi “annullabili” (defeasible), cioè che ammettono **eccezioni** (gli archi non esprimono condizioni necessarie)

EREDITARIETA'

- Attraverso archi *is-not-a* (reti di ereditarietà) oppure *overriding* di proprietà (frame, oggetti)



EREDITARIETA'

- Attraverso archi *is-not-a* (reti di ereditarietà) oppure *overriding* di proprietà (frame, oggetti)

point:object

x:=0; y:=0;

change_x(dx): x:=x+dx;

change_y(dy): y:=y+dy;

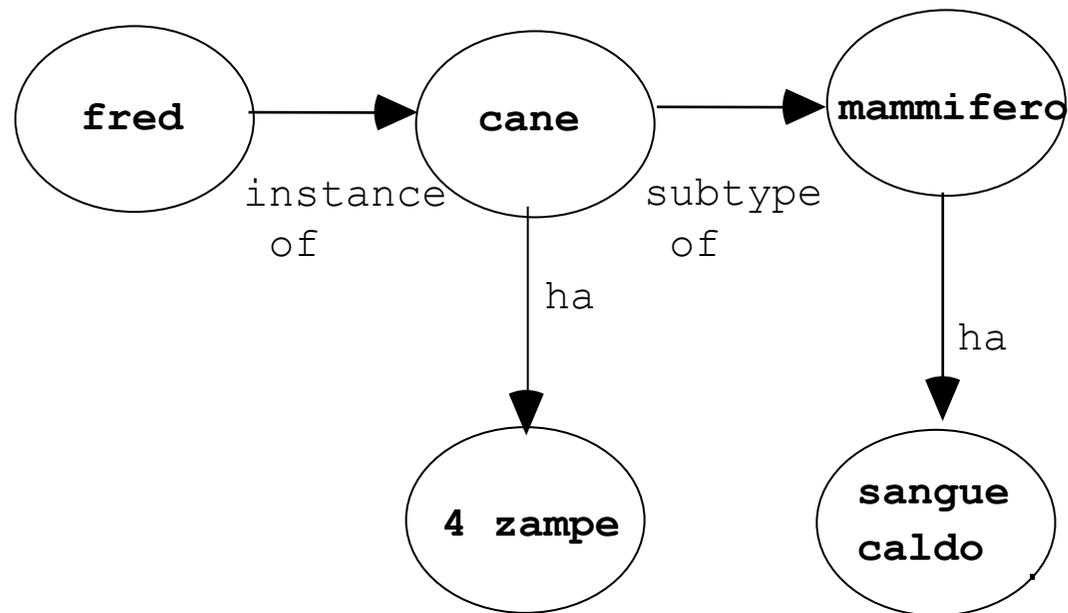
display_point: ...;

colored_point: point

x:=0; y:=0; c:=red;

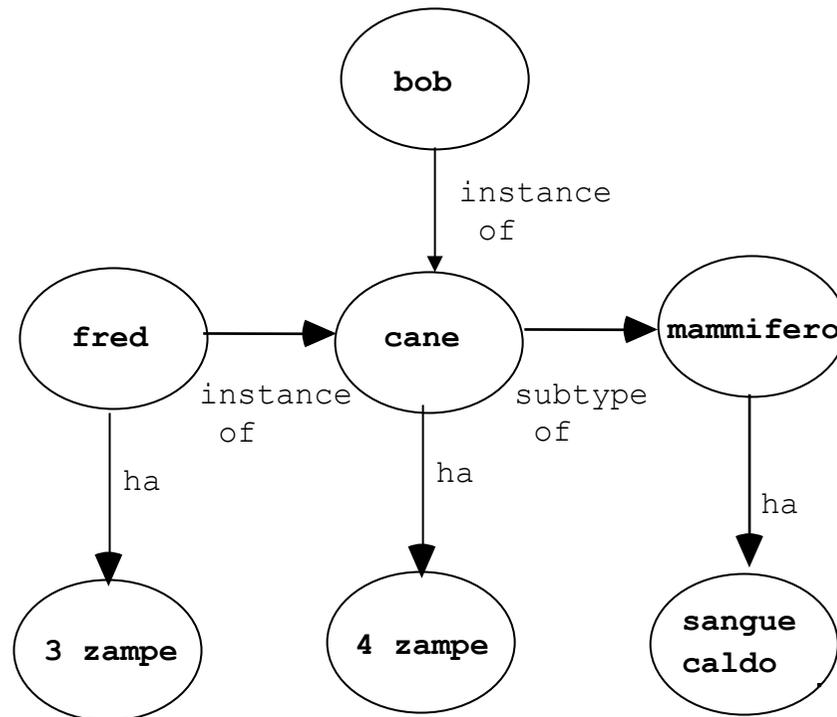
display_point: ...;

GERARCHIA ED EREDITARIETÀ DI PROPRIETÀ



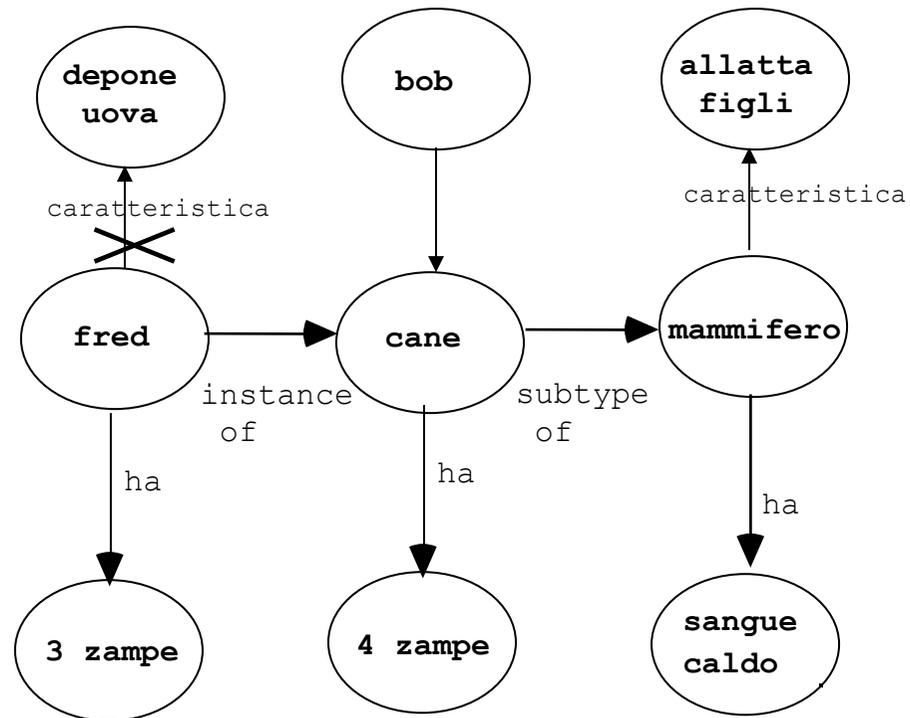
- fred è un cane allora:
- ha 4 zampe e ha il sangue caldo.

DEFAULT

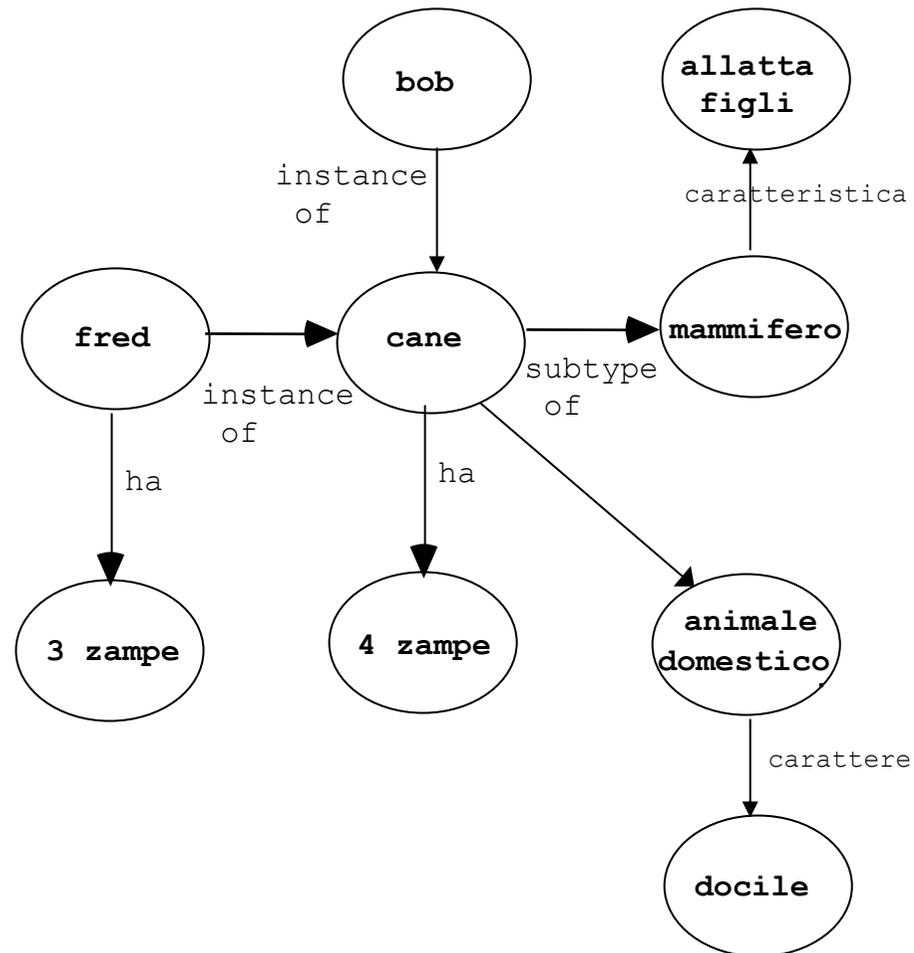


- mediante ricopertura di proprietà ai livelli più bassi della gerarchia fred ha tre gambe (eccezione).
- **ATTENZIONE!!!** ci sono proprietà che non possono essere ricoperte: (alcune sono quantificazioni universali)

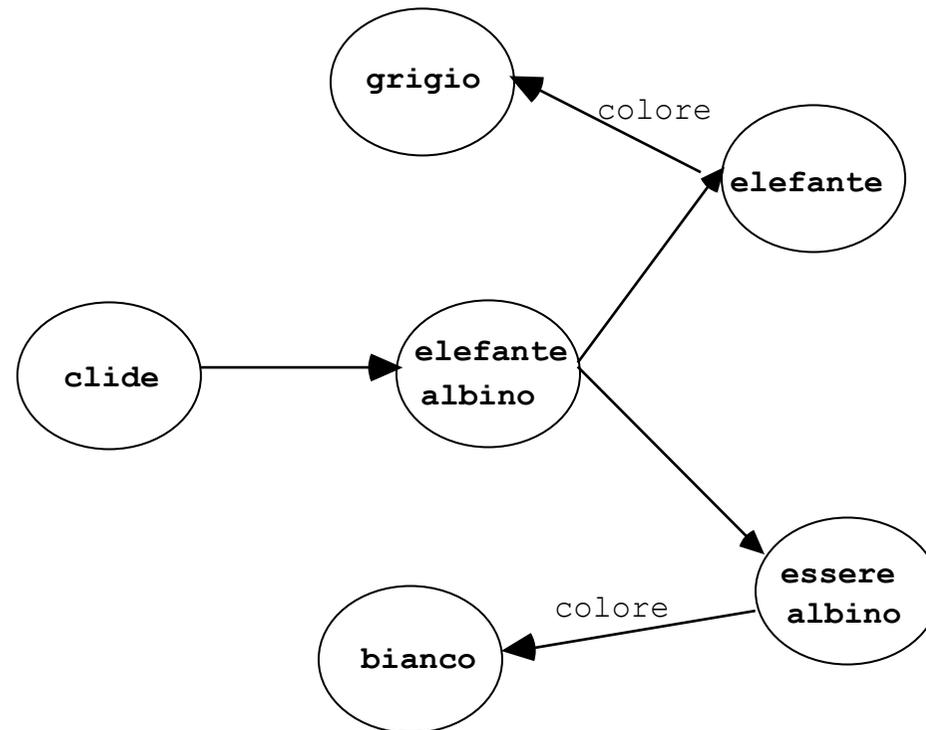
EREDITARIETÀ MULTIPLA:



EREDITARIETÀ MULTIPLA:

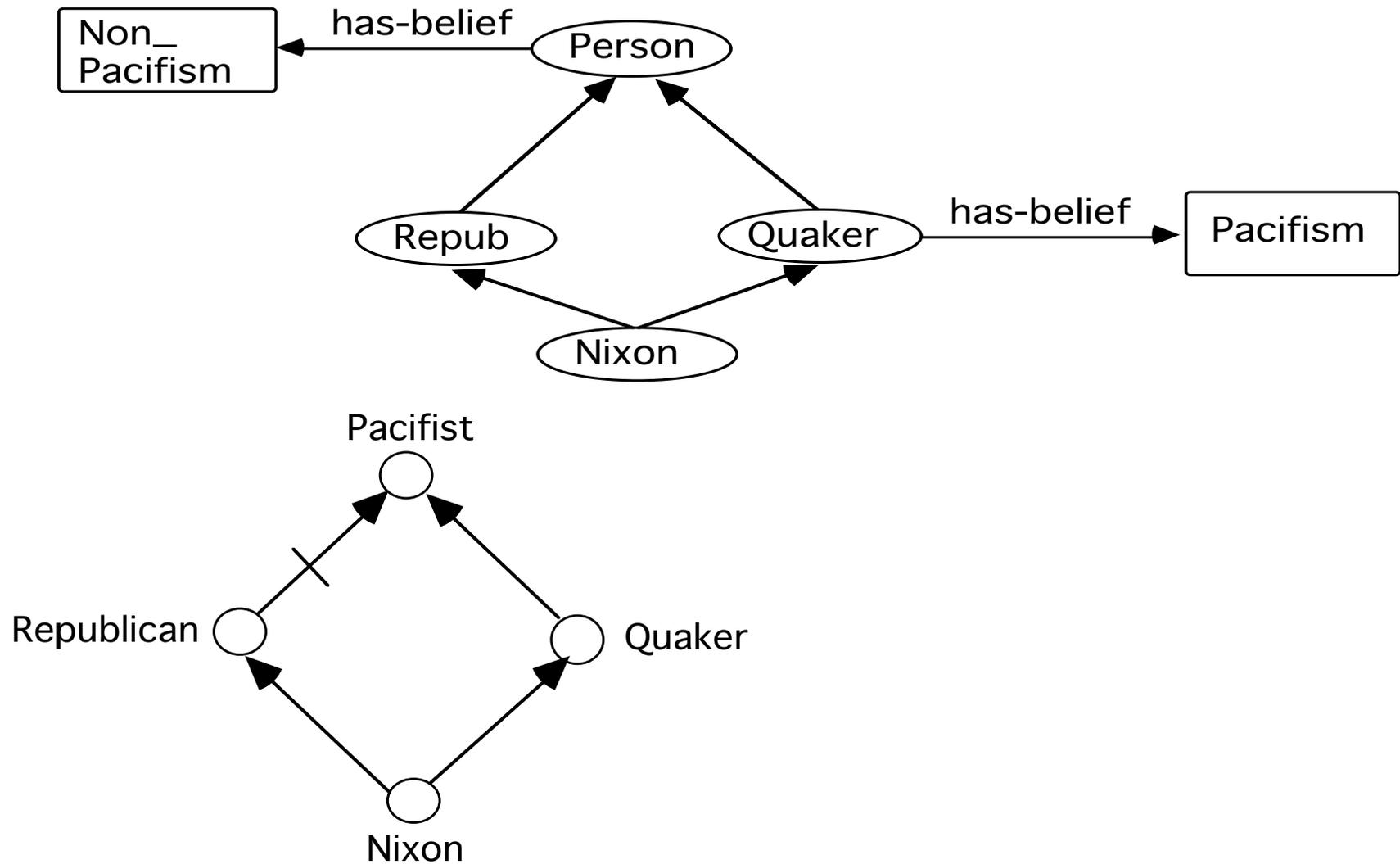


PROBLEMI: EREDITARIETÀ DI PROPRIETÀ IN CONTRADDIZIONE



- Si deve indicare un VERSO DI PERCORRENZA
- Normalmente depth-first con backtracking da sinistra verso destra.

EREDITARIETA' MULTIPLA



EREDITARIETA'

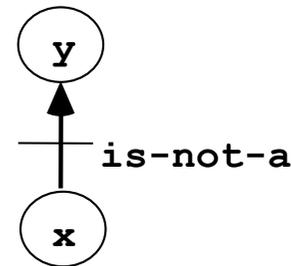
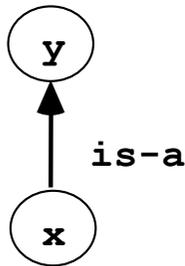
Due approcci:

- nell'ambito della rappresentazione della conoscenza *reti di ereditarietà*
 - derivano dalle reti semantiche
 - si ereditano proprietà
 - esistono archi defeasible (“is-not-a”)
 - ereditarietà multipla
- nell'ambito dei linguaggi di programmazione, *linguaggi a oggetti (C++, Java)*
 - si ereditano attributi (variabili) e metodi (operazioni)
 - sia le variabili sia le operazioni ereditabili possono essere ridefinite (overriding)
 - in alcuni c'è ereditarietà multipla (C++), in altri no (in Java, solo per interfacce)

Reti di ereditarietà (Touretsky, 1987)

- In generale, è un grafo diretto (aciclico) etichettato i cui nodi rappresentano individui e classi e gli archi relazioni “is-a” e “is-not-a” tra i nodi

- Due tipi di archi:



- $y(x)$

$$\forall Z \ x(Z) \Rightarrow y(Z)$$

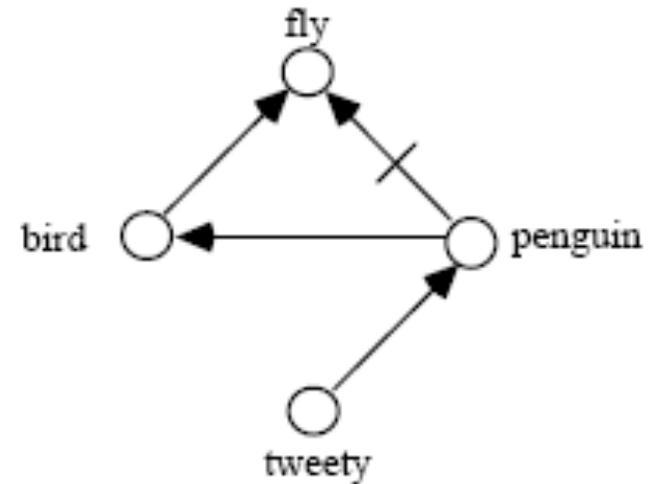
$$\neg y(x)$$

$$\forall Z \ x(Z) \Rightarrow \neg y(Z)$$

- Un sistema di ereditarietà è **bipolare** se ha sia archi is-a che is-not-a. È **unipolare** se ha solo archi is-a

Reti di ereditarietà

Esempio (1):



$\forall X \text{ (bird}(X) \Rightarrow \text{fly}(X))$

$\forall X \text{ (penguin}(X) \Rightarrow \neg \text{fly}(X))$

$\forall X \text{ (penguin}(X) \Rightarrow \text{bird}(X))$

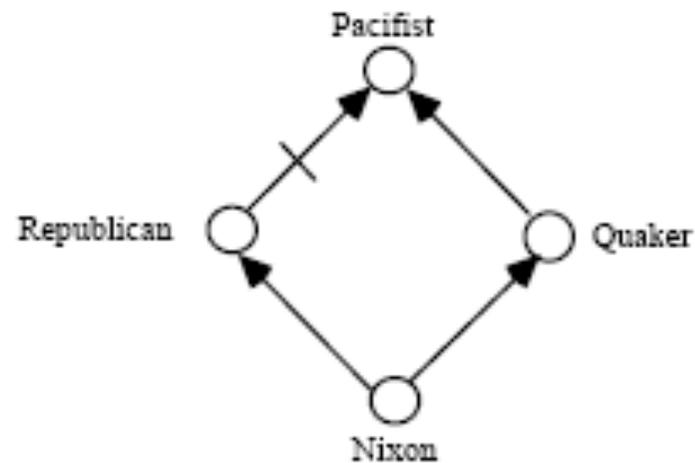
$\text{penguin}(\text{tweety})$

$\forall X \text{ (fly}(X) , \neg \text{fly}(X)) \Rightarrow \perp$

Il sistema è inconsistente

Reti di ereditarietà

Esempio (2):



$\forall X (\text{quacker}(X) \Rightarrow \text{pacifist}(X))$

$\forall X (\text{republican}(X) \Rightarrow \neg \text{pacifist}(X))$

$\text{quacker}(\text{nixon})$

$\text{republican}(\text{nixon})$

$\forall X (\text{pacifist}(X), \neg \text{pacifist}(X)) \Rightarrow \perp$

PATH-BASED REASONING

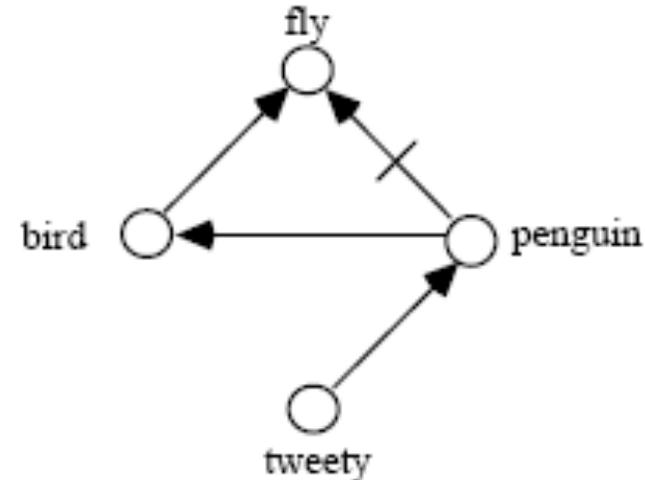
- Sistemi di ragionamento basato su cammini
- Ampiamente accettato perché intuitivo e di facile realizzazione
- *Eccezioni:* (vedi esempio (1))
- I concetti che hanno valori di proprietà in conflitto possono essere ordinati totalmente all'interno della gerarchia is-a
- Il conflitto può essere risolto sulla base della *locazione* identificando il concetto “più vicino” (il *più specifico*) e utilizzando l'informazione in esso (*overriding*)

Eccezioni

Tweety non vola

Il concetto *più specifico*, la sua classe più specifica, è penguin e utilizzando l'informazione in esso (*overriding*) stabiliamo che non vola

Esempio (1):



$$\forall X \text{ (bird}(X) \Rightarrow \text{fly}(X))$$

$$\forall X \text{ (penguin}(X) \Rightarrow \neg \text{fly}(X))$$

$$\forall X \text{ (penguin}(X) \Rightarrow \text{bird}(X))$$

$$\text{penguin}(\text{tweety})$$

$$\forall X \text{ (fly}(X) , \neg \text{fly}(X)) \Rightarrow \perp$$

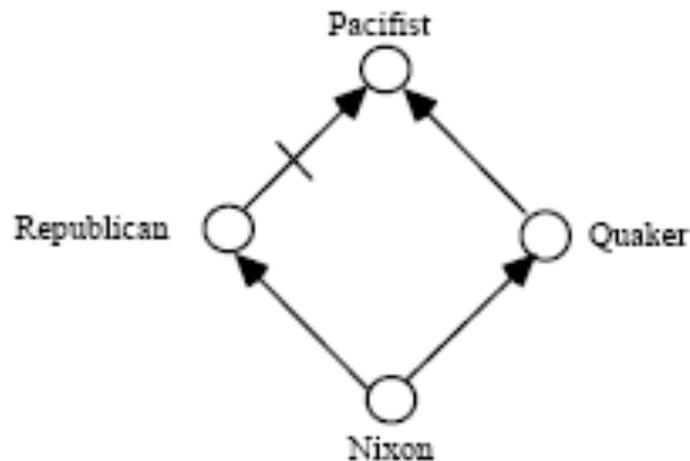
Il sistema è inconsistente

AMBIGUITA' (vedi esempio 2)

- Il conflitto *non* può essere risolto sulla base della locazione dei concetti nella gerarchia (si veda l'esempio di Nixon)
- Possibili soluzioni:
- *Approccio credulo*
 - Enumerazione delle risposte possibili
 - Stessi risultati ottenuti con la traslazione nella logica default
 - *Nixon è Pacifista*
 - *Nixon non è Pacifista*
- *Approccio scettico*
 - Non trae alcuna conclusione
 - *Non posso concludere nulla su Nixon*
- *L'ereditarietà scettica ideale* dovrebbe trarre quelle conclusioni che sono vere in *ogni* estensione credula
- *Priorità*
 - Aumento della potenza espressiva del linguaggio
 - È più probabile che un Repubblicano sia Pacifista piuttosto che un Quacchero sia non Pacifista

Reti di ereditarietà – approccio scettico

Esempio (2):



Non concludo nulla su Nixon (in alcune estensioni, è pacifista, in altre non pacifista)

$\forall X \text{ (quacker}(X) \Rightarrow \text{pacifist}(X))$

$\forall X \text{ (republican}(X) \Rightarrow \neg \text{pacifist}(X))$

$\text{quacker}(\text{nixon})$

$\text{republican}(\text{nixon})$

$\forall X \text{ (pacifist}(X) , \neg \text{pacifist}(X)) \Rightarrow \perp$

EREDITARIETA' PROCEDURALE

- Usata nei sistemi a frame e ad oggetti
- È definita solo in termini di un algoritmo che opera sulla struttura dati rappresentante il grafo di ereditarietà
- Algoritmo:
Dato un oggetto A e una proprietà P, trovarne il valore V:
 1. Se c'è un valore per la proprietà P in A restituisci quel valore.
 2. Altrimenti, sia Class l'insieme di tutte le classi che sono puntate da archi che escono da A;
 3. Se Class è vuoto restituisci fallimento;
 4. Altrimenti seleziona una classe C in Class. Se in C c'è un valore per la proprietà P restituisci quel valore. Altrimenti aggiungi a Class tutte le classi puntate da archi che escono da C.
 5. Ritorna al passo 3.
- Nel caso di ereditarietà semplice si percorre un unico cammino nella gerarchia
- Nel caso di ereditarietà multipla, la struttura dati (ad esempio coda o pila) utilizzata per memorizzare l'insieme Class corrisponde alle due strategie di ricerca breadth-first e depth-first

Abbiamo visto che:

- Molti dei ragionamenti che si fanno sono sulle categorie piuttosto che sugli individui
- Se organizziamo la conoscenza in categorie (e sottocategorie) è sufficiente classificare un oggetto, tramite le proprietà percepite, per inferire le proprietà della categoria/e a cui appartiene (***ereditarietà***)

In conclusione..

- Le relazioni di sottoclasse organizzano la conoscenza in tassonomie (come in botanica, biologia, nelle scienze librarie ...)
- Ontologie di dominio = tassonomie specializzate
- Molte delle idee delle reti semantiche e dei frame sono state raccolte in logiche specializzate (terminologiche o descrittive)
- Queste logiche sono alla base delle proposte per il *Web semantico*
- *Prossime due lezioni/esercitazioni seminariali su Description Logics e Semantic Web (14 e 15 Dicembre)*

Introduzione a Semantic Web e description Logics – Dott. Ing. Riccardo Zese (UNIFE)

- Il seminario è organizzato in due incontri che si terranno il:
- 14 Dicembre 2016, h. 11-13.30 in Aula 5, nel primo incontro verranno introdotte le Description Logic, alcuni operatori di base e una panoramica dell'iniziativa Semantic Web
- 15 Dicembre 2016, h. 8.45-11 Laboratorio Grande - Sarà' introdotto brevemente Protegé uno strumento per la definizione e la popolazione di ontologie; si utilizzerà il reasoner ontologico Pellet, realizzato in Java e open source, per eseguire inferenze.