Riassunto di rappresentazione della conoscenza

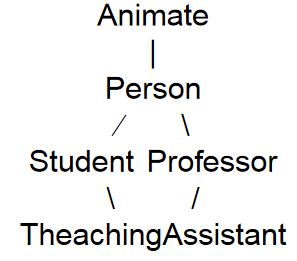
# Logiche descrittive, cosa sono?

Le logiche descrittive sono un formalismo con cui è possibile modellare le ontologie. Un’ontologia è una tassonomia, cioè una rappresentazione di un concetto in un dominio di interesse attraverso classi e sottoclassi (e quindi concetti più specifici o più generali) aventi delle proprietà (realizzate come relazioni tra concetti).

L’invenzione delle logiche descrittive risale a partire dalle reti semantiche e dai primi formalismi alla base del web semantico, oltre che di altri ambiti.

Il linguaggio Ontology Web Language (OWL) è un esempio di linguaggio che utilizza queste logiche, esso permette di modellare e ragionare sulle ontologie nel web semantico.

## Cos’è il web semantico?

Il web semantico è una rete di informazione collegate tra loro, essere è comprensibile ai programmi ed p condivisa dalle applicazioni.

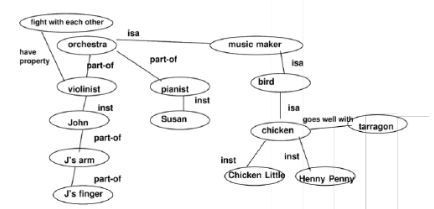
Ogni documento è associato a informazione e dati che ne descrivono la semantica, permettendone quindi il ragionamento.

Le classi sono insieme di individui del dominio e vengono rappresentati in logica come predicati unari, i ruoli invece sono relazioni tra individui indicate come predicati binari.

Per fare un esempio, considerando la tassonomia a destra:

* Animate indica la classe di tutti gli esseri viventi;
* Person è la classe di tutte le persone, eccetera.
* Ogni studente è una persona, così come ogni professore;
* Un teaching assistant è sia uno studente, sia un professore, ciò equivale all’intersezione tra insiemi.

Le linee indicano i ruoli tra classi, in questo caso indicano quali classi sono sottoclassi di altre classi.

In questi contesti si parla anche di inheritance network, cosa sono? Esse sono delle reti in cui in alto ci sono i conctti più generali e in basso quelli più specifici. Quando i concetti più specifici sono connessi a concetti più generali, essi ereditano da questi ultimi tutte le proprietà.

## In cosa consiste il linguaggio?

Il linguaggio delle logiche descrittive consiste in:

* concetti atomici indicati con predicati unari;
* ruoli tra classi e/o individui, dei predicati binari;
* costruttori per definire concetti e ruoli più complessi, alcuni di questi sono gli operatori insiemistici;
* assiomi definiti attraverso i concetti, questi sono generali quando stabiliscono relazioni tra classi, sono fatti se invece riguardano concetti, ruoli o individui.

## Cos’è un concetto?

Un concetto è l’equivalente di un insieme nella matematica degli insiemi, esso può essere:

* atomico quando è definito dal dominio;
* complesso quando è un’operazione tra più concetti atomici.

Per fare un esempio di concetto complesso, dati i concetti base di Madre e Giovane, si può costruire il concetto di MadreGiovane utilizzando l’intersezione di questi:

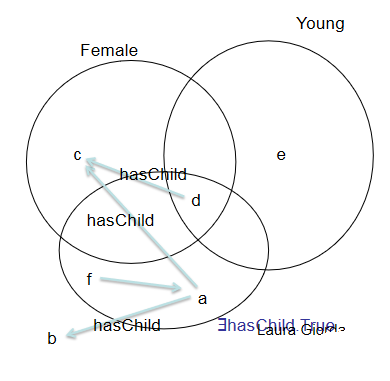
Esistono anche concetti particolari:

* ⊥ indica l’insieme vuoto e rappresenta il falso: indica nessun individuo è sia Madre, sia Padre, quindi i due concetti sono disgiunti.
* T rappresenta tutti gli individui e quindi il concetto di vero;
* Δ ha un significato simile a T, la differenza è che questo dipende dall’interpretazione.
* La negazione di un concetto si indica con ㄱ seguito dal concetto, per fare un esempio, il concetto delle madri vecchie può essere indicato come.

## Cos’è un costruttore?

Un costruttore in logica è qualcosa che permette la definizione di oggetti e relazioni composte, tra questi vi sono gli operatori insiemistici come l’unione, l’intersezione e i connettivi booleani.

## Cos’è una relazione?

Una relazione è un assioma che indica proprietà e ruoli tra due concetti o tra due individui, esse si indicano con dei predicati binari, ad esempio: HasChild(mary,john), indica che mary ha come figlio john.

Con le relazioni è possibile utilizzare i quantificatori per definire concetti.

Considerando il quantificatore esistenziale ヨ, esso indica che esiste almeno un individuo per cui vale la relazione, ad esempio:

* Madri giovani con almeno un figlio:;
* individui con almeno un amico sposato: .

Il quantificatore universale ∀ indica che la relazione vale per ogni individuo oppure per nessuno:

* Madri con tutti (o nessuno) figli medici: ;
* Individui con tutti gli amici sposati:

Tutte le cose viste in precedenza vengono utilizzate nella logica ALC.

## Base di conoscenza

Una base di conoscenza indica tutto ciò che si può considerare vero nel dominio, su di essa è possibile fare inferenza per poter ricavare la conoscenza implicita e quindi verificare dei fatti.

La conoscenza base è formata da:

* una TBox, cioè l’insieme degli assiomi terminologici che descrivono la struttura del dominio.

Questi assiomi valgono quindi per ogni elemento, ad esempio:

* una ABox, l’insieme di assiomi di membership, indicanti le proprietà degli specifici individui.

Elephant(dumbo), Mother(X), YoungMother(mary)

Considerando gli esempi della TBox, si possono notare due operatori:

* indica l’equivalenza, cioè quando due concetti coincidono, nell’esempio il concetto YoungMother ha gli stessi individui di ;
* indica invece un’inclusione, Elephant in questo caso è un animale grosso e grigio, ma non tutti gli animali con queste caratteristiche non sono elefanti.

Si considera in una ABox, anna è quindi un elefante bianco. Per definizione però gli elefanti sono grigi, quindi come si gestisce?

Se , allora White e Grey sono disgiunti, l’elefante bianco è quindi una contraddizione di quello grigio, cosa che genera una KB inconsistente.

In tal caso basta rimodellare la definizione di Elephant per includere anche gli elefanti bianchi:

Nel caso in cui l’intersezione tra Grey e White non genera un insieme vuoto, gli elefanti bianchi possono essere dichiarati con la vecchia definizione di Elephant.

## Cos’è un’interpretazione?

Un’interpretazione è una coppia (Δ,∙) che permette di descrivere una mappatura in cui si valuta ogni concetto e ruolo del dominio Δ attraverso una funzione di valutazione ∙.

La funzione di valutazione associa a ogni concetto un sottoinsieme del dominio e a ogni ruolo una relazione binaria, queste ultime sono precisamente degli insiemi di coppie.

L’interpretazione ha le seguenti proprietà:

* l’interpretazione dell’unione è l’unione delle interpretazioni;
* l’interpretazione della negazione è l’interpretazione del dominio meno quella del concetto;
* L’interpretazione dei quantificatori è un’insieme di coppie.

Un’interpretazione soddisfa una TBox se soddisfa tutti gli statement contenuti in essa;

## Sussunzione

L’inclusione tra concetti permette di indicare la sussunzione: dato , si dice che D sussume C. Da ciò, si ha che C è un concetto più specifico mentre D è più generale, si può quindi costruire un albero per indicare la generalità/specificità dei concetti.

In termini di interpretazione se un concetto D sussume un concetto C, allora l’interpretazione di C è contenuta in quella di D.

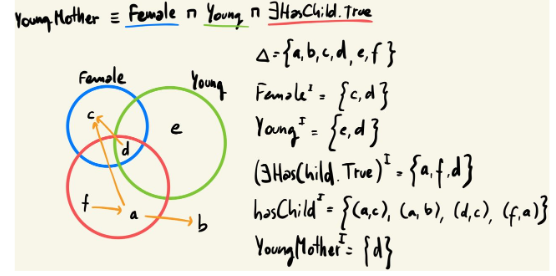
Cosa si può dire sui quantificatori e la sussunzione?

* ∃ non sussume ∀ dato che un individuo avente una relazione R con solo elementi di un concetto C non implica che ne abbia per forza una;
* ∀ non sussume ∃ dal momento che un individuo con una relazione R con un elemento di C non implica che abbia solo relazione con gli individui di quel concetto.

Tutto ciò vale per la TBox, per quando riguarda la ABox invece?

Il concetto di interpretazione rimane sempre lo stesso la differenza è che qui si considerano gli individui, infatti ogni individuo a viene interpretato come un dominio individuale a^I.

Un’interpretazione soddisfa una Abox se soddisfa tutti gli statement contenuti in essa.

Tutto ciò che è stato detto in precedenza si può notare nell’esempio a destra, infatti si può notare che d è una YoungMother dato che soddisfa sia Female, sia Young, si la relazione hasChild.

La relazione hasChild è un insieme di coppie di individui legati appunto da questa relazione, si può infatti notare nel disegno che l’individuo a ha tre figli, d ha c come figlio e f è il genitore di a.

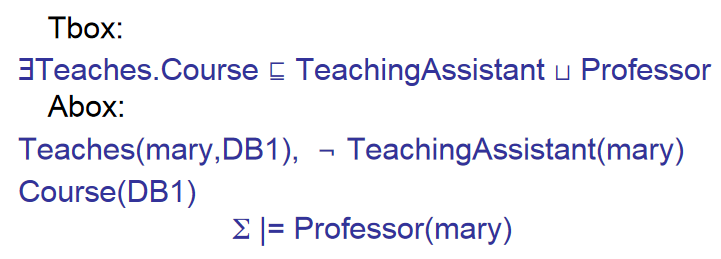
Ultima osservazione ma non meno importante riguarda Female, Young e le loro interpretazioni:

* L’interpretazione di Female è formata da c, h e d, gli individui al suo interno;
* Stessa cosa vale per Young ma per gli individui e, h e d;
* Quindi considerando il concetto, la sua interpretazione conterrà gli individui h e d, quelli in comune tra i due insiemi.

## Cosa si può dire sui modelli e sulla soddisfacibilità?

Un’interpretazione è un modello della base di conoscenza se è soddisfatta da ogni assioma contenuto in quest’ultima.

Allo stesso modo, una base di conoscenza è soddisfacibile o consistente se esiste almeno un modello per essa.

Da ciò si definisce il concetto di conseguenza logica: una formula α è una conseguenza logica di una conseguenza base se è soddisfatta da ogni suo modello.

Considerando la conoscenza base a destra, si può quindi dire che:

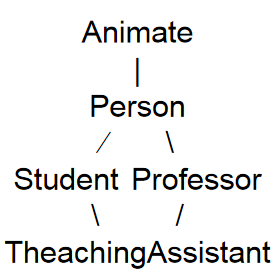
* Un teaching assistant che insegna almeno un corso è un professore (esiste una relazione Teaches con un elemento della classe Course);
* Dato che mary non è un teaching assistant e insegna il corso di DB1 (per via di Teaches(mary,DB1)), si conclude che mary è un professore, ciò è detto instance checking e serve a indicare se un individuo è un’istanza di una data classe.

## Servizi di ragionamento

* Soddisfacibilità di un concetto: dato un concetto C e una KB, se quest’ultima deriva C, allora il concetto non deve essere false. Se un concetto è soddisfacibile, allora non deduce il vuoto;
* Sussunzione;
* Soddisfacibilità della conoscenza base: data una KB, da essa non bisogna derivare il concetto vuoto. Questo punto indica se esiste o meno un modello della KB.
* Instance checking.

## Classificazione

Un problema è di classificazione quando bisogna trovare il posto giusto per un nuovo concetto C all’interno di una tassonomia.

Per fare un esempio, data la tassonomia a destra, dove si può piazzare il concetto ?

Ebbene, C va dopo Person e prima di Student e Professor, questo per i seguenti motivi:

* Person è un concetto più generale di C dato che quest’ultimo viene sussunto;
* Student e Professor sono più specifici di C dal momento che sussunti da quest’ultimo.

# Tipi di logiche descrittive

Oltre a quelli già citati, esistono anche altri costrutti che permettono la definizioni di nuovi concetti e ruoli.

## Restrizione numerica

La restrizione numerica permette di modellare classi aventi un dato numero di relazioni, ad esempio:

* : Insieme di individui aventi solo amici sposati e che al massimo sono 5, non possono avere amici non sposati;
* : insieme di madri con almeno 5 figli;
* : insieme di individui aventi al massimo 5 amici studenti, possono avere anche altri amici che non sono studenti.

Il quantificatore esistenziale si può considerare come caso particolare di restrizione numerica, infatti è equivalente a .

La restrizione numerica può essere di due tipi:

* non qualificata (N): la restrizione viene applicata a una relazione R non legata a un concetto;
* qualificata (Q): la restrizione viene applicata a una relazione R legata a un concetto C;

## Nominals (O)

I nominals sono insiemi atomici che contengono un solo individuo, essi permettono la definizione di concetti complessi, ad esempio:

* Professori che non insegnano il corso di DB1;

A livello di semantica, l’interpretazione dei nominal equivale all’interpretazione dell’individuo al loro interno:

I nominals devono rispettare la seguente proprietà: dato un insieme C contenente degli individui, l’unione di tutti i nominals di C deve equivalere a C stesso.

L'utilizzo dei nominals implica che nomi diversi sono associati a individui diversi, non è quindi presente la unique name assumption.

## Gerarchie di ruoli (H)

Una gerarchia di ruoli indica quando un ruolo è più specifico o generale rispetto a un altro, per indicarlo si utilizza il concetto di sussunzione.

Due ruoli sono equivalenti se si sussumono a vicenda.

## Ruoli transitivi

Un ruolo è transitivo quando vale la seguente condizione: dati degli individui x, y e z e un ruolo R, se valgono R(x,y) e R(y,z), allora vale anche R(x,z) senza definirlo in modo esplicito.

Per indicare che un ruolo R è transitivo, si mette un + come apice e si indica che è incluso in sé stesso:

A livello di semantica, l’interpretazione di un ruolo R equivale alla sua controparte transitiva:

La logica ALC unita ai ruoli transitivi prende il nome di logica S.

## Ruoli inversi (I)

I ruoli inversi indicano l’inverso di una relazione, essi si indicano mettendo un -1 come apice.

Prendendo come esempio HasChild e HasParent, questi sono ruoli inversi, quindi possono essere definiti nel seguente modo:

## SHOIN, SHOIQ e SROIQ

La logica ALC combinata con tutte le aggiunge citate prende il nome di logica SHOIN, esiste anche una variante detta SHOIN (D) in cui sono presenti tipo di dato semplici.

La logica SHOIQ è una variante di SHOIN in cui la restrizione numerica è qualificata, questa aggiunta però rende la complessità esponenziale, tuttavia esistono logiche descrittive di questo tipo a bassa complessità.

SROIQ invece è una semplificazione di SHOIQ in cui le gerarchie tra ruoli vengono generalizzate, ciò serve per evitare ruoli troppo generali e quindi l'indecidibilità.

Come funziona la semplificazione di SROIQ? essenzialmente utilizza un meccanismo simile alle grammatiche, infatti i risultati sulla decidibilità sono legati alla gerarchia di Chomsky.

## Ruoli funzionali, equivalenza e composizione

I ruoli funzionali sono essenzialmente funzioni sugli ruoli, con essi si possono modellare concetti come il seguente:

* : in questo cosa, il ruolo HasMother è funzionale;

L’equivalenza indica se due individui sono uguali o meno.

La composizione di ruoli permette di definire ruoli a partire da altri, ad esempio:

* hasParent∙hasBrother⊑hasUncle: uno zio è il fratello di un genitore.

# Regole di inferenza per la logica ALC

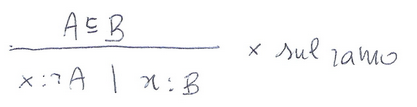
L’idea è quella di definire delle per ogni operatore, insieme a esse ne vengono definite altrettante per gestire gli stessi operatori ma con la negazione.

A partire da un tableau relativo alla conoscenza base, si continua a inferire ogni formula finchè non si raggiunge una contraddizione, creando in questo modo un albero.

I rami non chiusi possono portare a modelli, alcuni di questi potrebbero essere infiniti dal momento che continuano a generare nuovi individui.

La tendenza è quella di utilizzare prima le regole che non generano figli e/o non sdoppiano la computazione.

## Unfolding

La regola di unfolding permette di specificare se è vera l’inclusione tra concetti per un individuo x.

Dati due concetti A e B in cui B sussume A, l’unfolding si applica creando due rami, nel primo si verifica se l’individuo x non è un elemento di A, nel secondo invece si controlla se x è un individuo del concetto B.

## Operatore OR

L’operatore OR porta alla creazione di due rami dell’albero, in essi si controlla se l’individuo x appartiene a uno dei due concetti.

## Operatore AND

L’operatore AND non porta alla creazione di nessun ramo dell’albero, al contrario porta a ottenere un’informazione più specifica sull’individuo x.

## Operatore esistenziale

L’operatore esistenziale implica che un individuo x ha almeno una relazione R con un individuo z, quindi bisogna controllare se z è un concetto di C.

## Operatore universale

Con l’operatore universale, l’individuo x ha solo relazioni con gli elementi y, quindi controllare se y è nell’insieme C.

## Regole per la negazione degli operandi

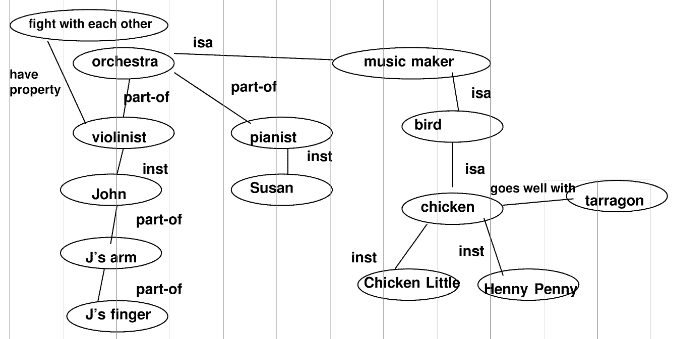
Oltre alla normali regole per gestire i vari operatori, si hanno anche delle regole per gestire le loro negazioni, precisamente:

* la negazione dell’OR equivale all’applicazione dell’AND e viceversa;
* la negazione del quantificatore esistenziale è simile a quella dell’universale, qui però bisogna controllare se l’individuo non appartiene al concetto di interesse;
* un discorso simile al precedente vale anche per la negazione del quantificatore universale.

## Quando una formula è derivabile da una KB?

Una formula α è derivabile da una KB quando la risoluzione di genera una contraddizione, questo metodo vale solo per l’instance checking.

# Reti semantiche

Una rete semantica è un grafo avente le seguenti caratteristiche:

* i nodi sono i concetti;
* gli archi indicano le relazioni tra concetti.

Le reti semantiche sono attualmente utilizzati per la comprensione del linguaggio naturale da parte delle macchine e per la rappresentazione del senso comune.

La ricerca si è focalizzata a dare una semantica chiara per le reti e a fornire meccanismi di inferenza corretti e completi, ciò ne ha permesso un’enorme semplificazione.

La foto in alto a destra è un esempio di rete semantica relativa alla composizione di un’orchestra, per verificare se alcuni fatti sono veri, basta interrogarla: il dito di John fa parte dell’orchestra? esistono galline che compongono musica?

## Logiche descrittive

Le logiche descrittive sono un sottoinsieme delle logiche del primo ordine che permettono proprietà esse sono risolvibili in tempo polinomiale.

L’obiettivo delle logiche descrittive è fornire un linguaggio per l’interrogazione e l’organizzazione, cosa molto più espressione di un linguaggio di un database e più efficiente di una logica del primo ordine.

### Algoritmo per la sussunzione

Per verificare se un concetto A sussume B:

1. appiattisce A e B rimuovendo tutti gli AND annidati;
2. raccoglie tutti gli argomenti per un dato ruolo
3. Assumendo che A sia nella forma (AND a1 … an) e B come (AND b1 … bm), si restituisce true se ai è un atomo SOME, quindi uno dei bj è ai. Si restituisce sempre true se ai è (ALL r x), quindi uno dei bj è (ALL r y) in cui x sussume y.

La complessità di questo algoritmo è quadratica per la lunghezza n dell’espressione.

Un utilizzo dell’algoritmo di sussunzione è della classificazione, infatti prendendo un concetto nuovo, bisogna trovargli un posto nella gerarchia in modo che sussuma un nodo ma che non venga sussunto dai figli di quest’ultimo.

## Inheritance network

Nella pratica, le logiche descrittive vengono utilizzate per le inheritance network, delle reti semantiche particolari in cui i concetti più specifici ereditano tutte le proprietà di quelli generali.

E’ in qualche modo possibile ereditare solamente certe proprietà?

Una prima soluzione potrebbe essere l’inserimento di proprietà complementari, ciò però rende la conoscenza inconsistente e di conseguenza non si può fare.

Una seconda soluzione è quella di aggiungere più classi per coprire il maggior numero di casi possibili, tuttavia non è semplice e potrebbe generare proprietà indesiderate.

### Esempio

Tutti gli uccelli sono animali che volano:

Tra tutti gli uccelli, spiccano i pinguini:

Dalle cose dette in precedenza, per transitività si ottiene che i pinguini volano, quando in realtà non è così.

Come si procede?

Si possono inserire delle eccezioni nelle sottoclassi in cui non si intende ereditare le proprietà, come si fa?

* Una soluzione è l’utilizzo di un attributo flag per contrassegnare la presenza di eccezioni in una classe, tuttavia ne servirebbe uno per ogni attributo;
* si introducono delle inheritance network con eccezioni, rappresentabili con un’apposito arco. Gli archi da una classe a una sottoclasse senza eccezioni vengono detti is-a, queste non sono più inclusioni strette bensì violabili in caso di eccezioni. Gli archi verso sottoclassi con eccezioni vengono detti is not a;

Prendendo come esempio Bird(opus), si può derivare che opus vola, ma se è un pinguino?

In tal caso basta concludere che non volta e non concludere più che vola, ciò rende la logica non monotona.

La monotonicità nella logica indica che l’aggiunta di informazioni alla KB non modifica l’implicazione delle precedenti conseguenze logiche, quindi l’aggiunta di nuova conoscenza restringe l’insieme dei modelli e tutto ciò che era vero prima continua ad esserlo.

## Logiche non monotone

Le logiche non monotone sono particolari logiche per cui non vale la proprietà di monotonicità della conseguenza logica:

Di conseguenza, aggiungendo nuova informazione alla conoscenza base, tutto ciò che era vero prima, adesso non lo è più e quindi bisogna ritrattare le soluzioni

L’utilizzo delle logiche non monotone permette la rappresentazione delle eccezioni nelle inheritance network. In caso di eredità multiple, quale criterio bisogna considerare per trovare le feature che valgono in un sottoclasse?

* La lunghezza del cammino non va bene dato che non sempre funziona, specie quando sono presente archi ridondanti;
* Si utilizza il concetto di specificità tra classi, cioè definire una gerarchia delle classi in modo da poter ragionare su di essa.

Precisamente vengono definiti i seguenti concetti:

* I cammini positivi sono tutti quelli formati solamente da archi positivi;
* i cammini negativi sono invece formati da archi positivi ma che terminano con un arco negativo;
* i non-cammini sono invece cammini in cui gli archi negativi non si trovano alla fine;

### Esempio

Gli elefanti sono animali solitamente grigi:

Gli elefanti reali sono elefanti ma di solito non sono grigi:

Un elefante reale piccolo è sia un elefante, sia un elefante reale:

In presenta di cammini multipli, come si procede? Un cammino è ereditabile se:

* è costruibile ricorsivamente nella rete;
* non presenta conflitti, cioè non ci sono cammini di segno opposto aventi stessa partenza e stessa destinazione già presenti nel contesto;
* non preemptivo, in tal caso se c’è un cammino che presenta un cammino conflittuale il quale presenta informazione più dirette sulla destinazione.

## Ragionamento non monotono

I ragionamento non monotono avviene facendo assunzioni, ciò è dovuto al fatto che la conoscenza è spesso incompleta ed è necessario ottenere soluzioni plausibili con ciò che è disponibile.

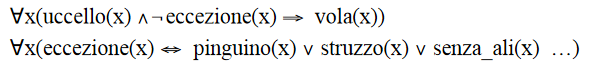
La conoscenza sul mondo è dato dalle regole che indicano le proprietà tipiche degli oggetti, eccezioni comprese.

Il ragionamento non monotono ha quindi bisogno di ritrattare le nuove informazioni qualora si ottenga un’informazione contraria, mostrandone la scorrettezza.

Le regole delle logiche non monotone iniziano solitamente con “Tipicamente”, “Di solito”, eccetera, cosa significa?

Ebbene, queste regole valgono solamente in assenza di particolare eccezioni.

Le regole a destra indicano come la logica monotona può simulare la variante non monotona, qui si ha che:

* la prima regola indica che ogni uccello di solito vola, qui il “di solito” indica tutti i casi indicati in “eccezione(x)”, che appunto devono essere esclusi;
* la seconda regola modella le eccezioni, in questo caso tutti gli uccelli che non volano.

Per dimostrare ad esempio che Tweety vola, bisogna dimostrare che non è un pinguino, uno struzzo, eccetera, c’è un modo migliore per farlo?

## Ipotesi del mondo chiuso

L’ipotesi del mondo chiuso è un’assunzione per cui, se un fatto positivo non è derivabile dalle formule di una base di conoscenza, allora si considera falsa, l’informazione negativa non è quindi rappresentata in modo esplicito.

Questa ipotesi è definita come una chiusura rispetto a Δ unita a tutte le assunzioni fatte:



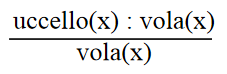
Nella formula sopra, Th è la chiusura deduttiva e indica che, dato un’insieme di regole S, una formula α fa parte della chiusura Th(S) se è derivabile da S.

Precisamente, l’ipotesi del mondo chiuso prende la teoria Δ e tutti gli atomi A negati non derivabili da essa, di tutto ciò si prendono tutte le formule derivabili attraverso la chiusura deduttiva.

L’ipotesi del mondo chiuso è non monotona per le seguenti ragioni:

* Dato un insieme S, si considera C come l’insieme di tutte le conclusioni ottenute col mondo chiuso;
* Se si aggiungono informazioni a S, C potrebbe diminuire;

## Logica di default

La logica di default utilizza regole di inferenza che non sono standard, essa permette di definire proprietà che valgono di default, ad esempio:

Considerando la regola a destra, essa può essere letta come “se x è un uccello e assume che vola, allora si può derivare che x vola”.

In generale, una regola di default è nella seguente forma:



In essa, α è il prerequisito, β è la giustificazione e infine γ è la conclusione, tutto ciò si legge “Se α vale e si assume β in modo consistente, allora si può concludere γ”.

## Teorie di default

Una teoria di default è una coppia <D,W> in cui:

* D è un insieme di regole di default;
* W è un insieme di formule della logica classica.

Tutto ciò si può vedere come un’estensione della logica classica con delle regole di default.

Inoltre se la giustificazione β risulta consistente, allora si può omettere nella regola di default, diventando quindi una normale regola di inferenza.

### Esempio

Considerando la teoria di default a destra, se si aggiunge pinguino(Tweety) a W, allora da questo insieme si può derivare not(vola(Tweety)), la quale blocca l’applicazione della regola di default.

In questo caso c’è una sola estensione per la teoria di default, ci sono però casi in cui esistono più estensioni.

Considerando la teoria a destra, entrambe le regole di D hanno un requisito derivabile in W, quindi se si applica la prima, non è possibile applicare la seconda e viceversa.

Da ciò, si conclude che la teoria ha due estensioni.

## Cos’è un’estensione?

Un’estensione di una teoria di default è un insieme di regole contenenti l’insieme W, premesse e conseguenze (non le giustificazioni) delle regole di default applicabili più eventuali formule presenti per via della chiusura deduttiva (deduttivamente chiuso).

Una teoria di default può avere zero o più estensioni, da ciò si può dire che la conseguenza logica di una formula da una teoria di default può essere:

* scettica quando la formula che segue da una teoria è vero in ogni estensione;
* credula quando la formula è vera in almeno un estensione.

Nell’ultimo esempio, entrambe le estensioni sono credule ma non scettiche, infatti non c’è un’estensione per cui Nixon risulta sia pacifista, sia guerrafondaio.

## Come si verifica un estensione?

La verifica di un’estensione avviene utilizzando un metodo lineare, data una teoria <W,D> e un estensione E:

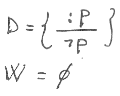
* Si ricava l’estensione E0, equivalente all’insieme W;
* Nel passo generico i, Si ricava l’estensione Ei+1 come la chiusura deduttiva di Ei unita a tutti i conseguenti applicabili nelle regole di default;
* Precisamente si includono i conseguenti delle regole di default per cui la premessa è nell’insieme E mentre la giustificazione no.

Come mai possono esserci differenti estensioni? Ebbene, ciò succede in base all’ordine con cui vengono prese in considerazione le regole di default, infatti, nell'esempio dei quaccheri e dei repubblicani:

| La prima estensione si ricava prendendo le regole di default nell’ordine dato dall’insieme D: | La seconda estensione viene fuori utilizzando l’ordine inverso delle regole: |
| --- | --- |

### Esempio

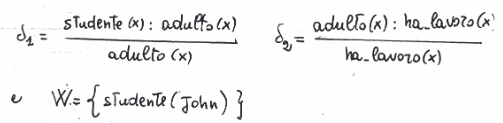
Si considera teoria di default fatta nel seguente modo:

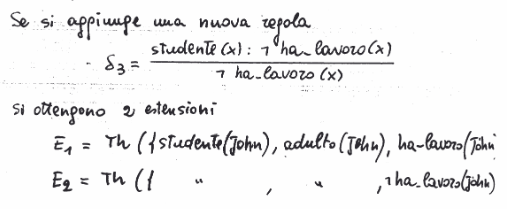


Qui non sono presenti estensioni, infatti provando a verificarne una, risulta impossibile applicare la regola di default.

## Default normali

I default normali sono particolari regole in cui conseguenza e giustificazione sono uguali. Qui bisogna stare attenti dato che l’applicazione di un default non deve contraddire la giustificazione di tutti quelli applicati in precedenza.

Considerando la teoria di default a destra, l’applicazione delle regole d1 e d2 indica che, di default, uno studente ha un lavoro (a causa della transitività), cosa che non è ragionevole.

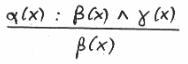
Per rendere tutto sensato si aggiunge una terza regola, ciò permette di ottenere due estensioni:

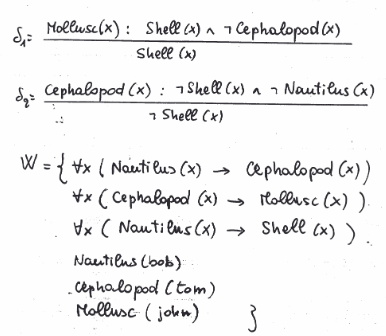
* la prima indica gli studenti con un lavoro;
* la seconda gli studenti disoccupati.

Se non si vuole utilizzare la transitività, bisogna ricorrere ai default semi normali.

## Default semi normali

I default semi normali sono particolari default aventi la seguente struttura:



I default semi normali permettono di bloccare la transitività, tuttavia essi non godono della proprietà di semi-monotonicità e possono non avere estensioni.

Questo tipo di default può essere utilizzato per rappresentare gerarchie con eccezioni.

Considerando la teoria di default a destra, si ha che:

* se x è un mollusco e si assume che abbia il guscio e che non è un cefalopode, allora ha il guscio;
* se x è un cefalopode e si assume nè che abbia il guscio, nè che sia un nautilus, allora non ha il guscio;
* ogni nautilus è un cefalopode;
* ogni cefalopode è un mollusco;
* ogni nautilus ha il guscio;
* bob è un nautilus, tom è un cefalopode e john è un mollusco.

Da ciò si ricava un’unica estensione, ossia che bob e john hanno il guscio mentre tom non ce l’ha.

## Non monotonicità nella programmazione logica

Nella programmazione logica, la non monotonicità è stata introdotta attraverso la negazione per fallimento, essa ha una semantica operazionale, è possibile darne una semantica dichiarativa?

Sì, esistono relazioni strette tra approcci di programmazione con negazione e logiche non monotone.

## Negazione per fallimento

La negazione per fallimento è possibile attraverso i goal negati, questi presentano la forma ?−not p e vengono gestiti nel seguenti modo:

* se ?−not p ha successo, ?−p fallisce;
* se invece ?−not p fallisce, allora ?−p ha successo;
* nei casi in cui la computazione va all’infinito, ?−not p è indefinito.

Questo tipo di negazione introduce non monotonicità dal momento che un goal può avere successo da un programma P, L’aggiunta di nuove clausole però potrebbe farlo fallire.

## Modello minimo

Considerando la logica proposizionale, la semantica model-theoretic è data dalla relazione di conseguenza logica della logica classica. Questo tipo di semantica su programma P è l’insieme dei simboli proposizionali A derivabili da quest’ultimo.

L’insieme ottenuto equivale a un modello del programma.

Da ciò si definisce il concetto di modello minimo come l’intersezione di tutti i possibili modelli di un programma logico.

## Completamento di Clark

Il completamento di Clark è un modo per modellare la semantica della negazione per fallimento.

Dato un programma logico P (visto come un insieme di regole nella logica classica), il completamento di P è un’insieme di formule tali che gli atomi derivabili da P siano conseguenze logiche del completamento.

Attraverso il completamento, le condizioni sufficienti sulle verità degli atomi vengono considerate necessarie.

Considerando il caso proposizionale, il completamento di un programma P:

* converte le regole , , in ;
* se un atomo q non è definito nel programma p, allora il completamento contiene not q;

Nel caso predicativo invece, il completamento deve contenere anche degli assiomi di uguaglianza per l’ipotesi dei nomi unici.

Per i programmi positivi, la derivazione di un programma P è corretta e completa rispetto alla derivazione del suo completamento.

Il completamento non è esente da problemi, infatti risulta inconsistente quando il programma contiene:

* regole nella forma ;
* regole che fanno andare la computazione in loop, serve quindi un sistema di loop checking.

E’ comunque possibile definire delle restrizioni sintattiche sui programmi per garantire consistenza e completezza.

Attraverso il completamento, la negazione per fallimento diventa la negazione classica.

## Logiche preferenziali

Le logiche preferenziali sono particolari logiche appartenenti alla classe delle logiche condizionali, il loro scopo è quello di valutare le conseguenze nel caso valesse qualcosa, il cosiddetto ragionamento controfattuale.

Nelle logiche preferenziali definite delle regole condizionali che differiscono dalle classiche implicazioni, A |~ B indica infatti che normalmente tutti gli A sono B, tuttavia non sempre è così.

Le proprietà da rispettare sono le seguenti:

* Riflessività: A |~ A;
* Equivalenza a sinistra: se A e B sono equivalente e A |~ C , allora B |~ C;
* Indebolimento a destra: se B⊨C e A |~ B allora A |~ C;
* Proprietà dell’AND: Se A |~ B e A |~ C allora A |~ B∧C:
* Proprietà dell’OR: Se A |~ C e B |~ C allora A∨B |~ C;
* Monotonicità cauta: Se A |~ B e A |~ C allora A∧B |~ C;
* Monotonicità razionale: Se A |~ C e non (A |~ ¬B) allora A∧B |~ C.

Un fattore importante nell logiche preferenziali è il concetto di tipicità, considerando un mondo w come lo scenario più plausibile, allontanandosì da esso ci saranno sempre meno scenari plausibili. Cosa vuol dire ciò? Essenzialmente w è il mondo più tipico rispetto a tutti gli altri, la tipicità indica quindi quanto uno scenario appartiene a una classe.

## Interpretazione preferenziale

Un’interpretazione preferenziale è una tripla (W,v,<) in cui:

* W è un insieme di mondi, le interpretazioni classiche;
* v è un una funzione di valutazione che assegna un valore di verità a ogni mondo e a ogni proposizione;
* < è una relazione non riflessiva, transitiva e ben fondata (non esistono catene di preferenze infinite), indicante quale mondo preferire. Nel caso in cui questa relazione sia modulare, allora il modello è detto ranked.

Un relazione di conseguenza in queste logiche può essere:

* preferenziale se l’insieme S è formato da regole condizionali A |~ B che soddisfano le proprietà di riflessività e monotonicità cauta. Per ogni relazione preferenziale S esiste un modello preferenziale che soddisfa tutte le regole condizionali al suo interno;
* razionale se l’insieme S è formato da regole condizionali A |~ B tali che soddisfano riflessività e monotonicità cauta e razionale. Per ogni relazione razionale S esiste un modello preferenziale ranked che soddisfa tutte le regole condizionali al suo interno.

## Motivazioni

La tipicità relativa degli individui di un dominio dipende dagli aspetti considerato per il confronto, le cosiddette relazioni di preferenza, ad esempio:

* bob è preferito a jim dal momento che ama di più lo sport;
* jim è preferito a bob perchè è un nuotatore più bravo.

Quello che si vuole ottenere è un entailment multi preferenziale che:

* soddisfa le proprietà delle logiche preferenziali di un sistema;
* permette di gestire la specificità, cioè utilizzare una classe più specifica al posto di una più generale;
* permette di gestire l’irrilevanza, quindi che le informazioni irrilevanti non portino a conclusioni differenti. Per fare un esempio, se normalmente gli uccelli volano, allora tutti gli uccelli aventi un colore specifico volano, il colore in questo caso è un dato irrilevante.
* evita il blocco dell’eredità e il drowning problem;
* evita molte conclusioni forti della logica RC.

## Operatore di Tipicità

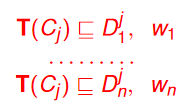
Si può definire per le logiche descrittive l'operatore di tipicità T che, dato un concetto C, T(C) isole le istanze del concetto C considerate più normali.

Quindi considerando l’inclusione T(C)⊑D, il suo significato è “tutti gli elementi di C tipici sono D”, “normalmente gli individui di C sono D”, eccetera.

La logica preferenziale presenta:

* una semantica preferenziale per l'inclusione di tipicità delle logiche descrittive;
* una semantica razionale per le entailment preferenziale;
* una chiusura razionale e una lessicografica.

Le logiche preferenziali permettono di avere una conoscenza base pesata in base alle preferenze, come si può vedere tutto ciò?

* La ABox rimane sempre la stessa;
* La TBox è formata dai concetti più un insieme Tstrict, una sorta di TBox contenente tutte le inclusioni strette;
* Le proprietà di tipicità per alcuni concetti, a questi vengono associati dei pesi, questi non sono a somma zero e possono essere positivi o negativi.

Data un’interpretazione preferenziale I con un dominio finito, per ogni concetto c in C si definisce un preordine totale <=c sul dominio tale che:

* x <=c y indica che x è almeno tipico quanto y per il concetto c.

Considerando tutte le inclusioni T(c) contenute in un concetto D per un peso w, si può definire W(x) come il peso di un individuo x nel seguente modo:

* la somma dei pesi di tutti i concetti C nel caso in cui x sia un’individuo di C;
* -∞ altrimenti.

Più sono plausibili le inclusioni per c soddisfatte da x, più sarà alto il peso W(x).

### Esempio

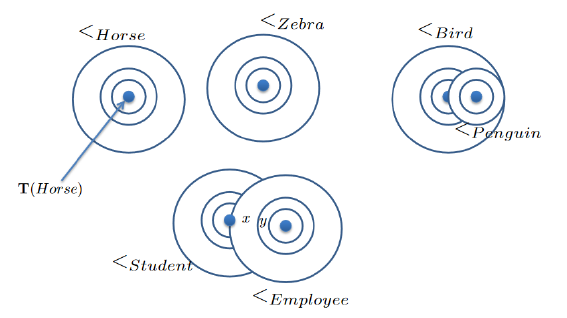
Data la conoscenza base pesata descritta nella foto a destra.

La Tbox pesata Thorse è definita come un’ordinamento degli elementi del dominio rispetto alla loro tipicità sull’attributo Horse.

Per determinare quale cavallo è più tipico, si fa un confronto sulle sommatorie dei pesi, chi è più tipico tra spirit e buddy? Secondo la base di conoscenza:

* spirit è un cavallo con la criniera lunga ma non corre veloce e non presenta strisce, il suo peso è di 20;
* buddy invece, oltre alla criniera lunga, ha le strisce e corre veloce, il suo peso è 20+20-80=-40.

NOTA: se un individuo non appartiene a un dato concetto, il relativo peso non viene considerato.

Da ciò si conclude che spirit è un cavallo più tipico rispetto a buddy e quindi preferito a quest’ultimo.

Dati due individui x e y del dominio, una relazione di preferenza per un concetto c indica che x è preferito a y se il peso di x è maggiore o uguale a quello di y.

Tutto ciò si può modellare come mostrato nella foto a destra, al centro di ogni cerchio è presente l’elemento più tipico per un dato concetto.

La combinazione delle preferenze ne permette la valutazione rispetto a ogni concetto, si può quindi far prevalere delle preferenze per relazioni specifiche rispetto a quelle più generali.

## Logiche Modali

Le logiche modali permettono di rappresentare gli stati mentali di un agente, quindi ciò che crede, conosce e/o sa che vale.

Tutto ciò infatti non può essere fatto con la logica del primo ordine dato che i predicati non possono contenere formule con argomenti.

Per fare degli esempio:

* Belief(john, rain) non indica che john dice che piove, lui potrebbe infatti avere credenze false. Tutto ciò per dire che la semantica di Belief non è proprio funzionale.

L’operatore di Belief è referenzialmente opaco, cioè si presenta in contesti opachi in cui non è possibile sostituire una formula f con una equivalente (come si fa di solito nella logica classica).

Per fare un esempio:

* Belief(john, father(zeus,chronos)) indica che john crede che zeus sia il padre di chronos;
* Anche se zeus e jupiter sono lo stesso individuo, se john non lo sa, allora non può derivare che jupiter sia padrea di chronos.

Per risolvere questo problema, si possono utilizzare dei metalinguaggi aventi termini per denotare formule, un’altra soluzione invece sono le logiche modali.

L’idea delle logiche modali è quella di considerare le credenze come dei connettivi, permettendo quindi di utilizzare la stessa semantica della logica proposizionale.

Precisamente, le logiche modali aggiungono operatori riguardanti le necessità:

* ⬜A: indica che A è necessariamente vera e rimane tale in ogni possibile mondo;
* ♢A: indica che A è possibile in almeno uno dei mondi possibili.

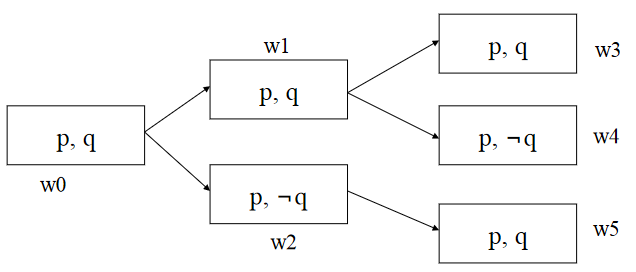
In altre parole, i due operatori introdotti indicano i concetti di:

* verità necessaria, cioè quando un fatto è vero in ogni caso, ad esempio: la radice quadrata di 2 è un numero irrazionale;
* verità contingente, un fatto che può essere vero adesso ma non per forza lo sarà in futuro, ad esempio: il presidente del consiglio è Mario Draghi (fatto vero nel 2020 ma diventerà falso dalla prossime elezioni).

Questi operatori sono duali, precisamente .

Esempio

Considerando la foto a destra in cui sono rappresentate le relazioni tra i vari mondi, si ha che:

* ⬜p è vero in tutti i mondi raggiungibili da w0, quindi w0, w1 e w2;
* ⬜⬜p indica che è necessario che sia necessario che in ogni mondo p sia vero, ciò vale solo per w0;
* ♢q è vero in w0, infatti nella foto non è negato.

Tutte queste formule vengono applicate a tutti i mondi possibili, infatti ⬜p è vero in w0 perchè p è vero nei mondi immediatamente successivi a w0, lo stesso discorso vale anche per w1 e w2. ⬜p non è vero per w3, w4 e w5 dal momento che non hanno mondi a seguire.

## Semantica modale

Un modello della semantica modale è una tripla <W,L,R> in cui:

* W è l’insieme di tutti i mondi possibili;
* R=WxW la relazione di accessibilità, cioè l’insieme di relazioni tra mondi;
* L è una funzione di valutazione che restituisce le formule atomiche vere in un dato mondo, tutte queste formano il modello.

La soddisfacibilità di un modello è legata al mondo, infatti:

* p è conseguenza logica dell’interpretazione M nel mondo e se p è un atomo in L(w);

M |=wp iff p ∈ L(w)

* Lo stesso discorso vale anche l’OR tra formule;

M |=wφ ∨ ψ iff M |=wφ or M |=wψ

* ¬φ è conseguenza logica di M se φ non lo è

M |=w¬φ iff M |≠wφ

* ⬜φ è conseguenza logica di M per un mondo w se, per ogni mondo w’ destinatario di una relazione con w, φ è conseguenza logica di M per w’;

M |=w⬜φ iff (∀w’: R(w,w’) ⇒M |=w’φ)

* ◊φ è conseguenza logica di M per w se esiste un mondo w’, destinatario di una relazione con w, tale che φ è conseguenza logica di M per w’.

M |=w◊φ iff (∃w’: R(w,w’) and M |=w’φ)

Inoltre, una formula φ è valida se è vera in tutti i mondi possibili, è soddisfacibile se esiste almeno un mondo in cui è soddisfatta.

## Proprietà

Le proprietà valgono per tutti i modelli, esse sono:

* Assioma T (riflessività): Ogni mondo è raggiungibile da sè stesso;
* Assioma D (serialità): Ogni mondo ha almeno una relazione con un’altro mondo;
* Assioma 4 (transitività): se l’agente sa una formula, allora sa di saperla;
* Assioma 5 (euclidea): se una formula è possibile, allora l’agente sa che è possibile;
* Assioma K (necessità): l’agente conosce le conseguenze di ciò che sa, di conseguenza fa inferenza attraverso il modus ponens. Da ciò si può dire che, se una formula f è valida, allora anche ⬜f è valida.

Le proprietà hanno associate delle teorie di corrispondenza, in esse gli assiomi sono veri in tutti e soli i modelli le cui relazioni soddisfano le proprietà date:

* riflessività(T): ⬜φ⇒φ;
* serialità(D):⬜φ⇒◊φ;
* transitività(4):⬜φ⇒⬜⬜φ;
* euclidea(5):◊φ⇒⬜◊φ.
* necessità(K):⬜(a → b) = ⬜a → ⬜b

Da ciò è possibile definire dei sistemi in base a come vengono combinati le varie proprietà:

* il sistema KT è conosciuto anche come T;
* KT4 è detto anche S4;
* KD45 è chiamato anche S5 debole;
* KT45 è conosciuto anche come S5.

L’assioma K vale in ogni caso dato che si riferisce alle logiche normali.

Il problema di validità per queste logica è decidibile e, a eccezione di S5 che è NP-Completo, sono tutte PSpace complete.

## Logiche epistemiche

Le logiche epistemiche sono particolari logiche in cui l’operatore ⬜ viene interpretato come:

* conoscenza (K) quando l’agente conosce una data formula f;
* credenza (B) quando l’agente crede una formula f.

Per quanto riguarda la conoscenza, tutte le proprietà e gli assiomi valgono, quindi se valgono l’assioma K e la necessitazione, l’agente è detto onniscente. Ciò vale perchè:

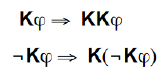
* l’agente può trarre conclusioni dalle premesse che ha;
* l’agente conosce ogni formula valida e sa applicare il modus ponens.

## Assiomi per conoscenza e credenza

L’assioma D indica che la conoscenza dell’agente è non contradditoria: se l’agente conosce una formula f, allora non conosce not f, ciò vale anche per le credenze.



L’assioma T indica che, se l’agente conosce una formula f, allora f è vera. Questo assioma non vale per le credenze, si può infatti credere qualcosa che risulta falso.



Gli assiomi 4 e 5 riguardano rispettivamente l’introspezione positiva e negativa:

* la prima indica che, se l’agente conosce una formula f, allora sa di sapere f;
* la seconda invece indica che, se l’agente non conosce f, allora sa di non sapere f.

Detto ciò, il sistema modale S5 viene di solito scelto come logica per la conoscenza, per le credenze invece si utilizza S5 debole.

## Esempio: problema dei saggi

Ci sono tre saggi che sanno che almeno uno di loro ha una macchia bianca sulla fronte, ognuno può vedere la fronte degli altri ma non la propria.

Sui saggi si hanno le seguenti informazioni:

* il primo dice che non sa se ha la macchia bianca;
* il secondo pure;
* il terzo invece sa di averla.

Com’è possibile? Considerando lo stesso problema ma con due saggi A e B, si modellano le informazioni nel seguente modo:

* A sa che, se lui non ha la macchia bianca, B lo sa;

KA(¬white(A) ⇒KB(¬white(A)))

* A sa che B sa che, se A non ha la macchia bianca, allora è lui ad averla;

KA(KB(¬white(A) ⇒ white(B)))

* A sa che B non sa di avere la macchia bianca.

KA(¬KB(white(B)))

Da tutto ciò, si conclude che A sa di avere la macchia bianca.

## Logiche dinamiche

Le logiche dinamiche sono particolari logiche modali che permettono di rappresentare sequenze di azioni, con esse è possibile costruire veri e propri programmi e associare delle modalità a ognuno di essi.

Il linguaggio della logica dinamica utilizza quello dei programmi regolari, quest’ultimo è precisamente formato dalle seguenti operazioni:

* a indica un’azione atomica;
* p;q è una sequenza di azioni;
* p+q indica una scelta non deterministica tra p e q;
* p\* è un interazione;
* p? indica un test.

La logica dinamica arricchisce il linguaggio regolare con altre operazioni, tra queste vi sono:

* [p]q: indica che dopo il termine dell’azione p, bisogna trovarsi in un stato che soddisfa q;
* 〈p〉q: indica che dopo il termine dell’azione p, è possibile trovarsi in uno stato che soddisfa q.

## Logiche deontiche

Le logiche deontiche sono logiche modali per rappresentare i concetti di obbligo(O) e permesso(P), corrispondenti rispettivamente a ⬜ e ◊.

Obblighi e permetti sono tra loro duali.

La logica deontica è di solito modellata come un sistema KD, quindi è una logica modale aventi i seguenti assiomi:

* Un obbligo è anche un permesso;



* Un divieto è una formula non permessa.



## Logiche temporali

Le logiche temporali sono logiche modali utili per gestire il tempo, ne esistono di diversi tipi ma, per semplicità, in questo riassunto si considera una logica con queste caratteristiche:

* tempo discreto;
* un inizio ben preciso;
* infinità verso il futuro.

Con le logiche temporali si hanno sequenze di mondi possibili rappresentanti le conseguenze delle azioni, queste sequenze possono essere:

* lineari, cioè ogni mondo è connesso a un solo mondo successivo (una struttura appunto lineare);
* ad albero, qui ogni mondo è connesso a più mondi successivi, indicando i possibili scenari.

Considerando la logica temporale lineare, oltre ai normali operatore logici, vengono anche aggiunti i seguenti:

* X a indica che la formula a è vera nel mondo successivo;
* a U b indica è vero nel caso in cui a vale fino a quando non vale b.

Da questi si definiscono altri due operatori temporali:

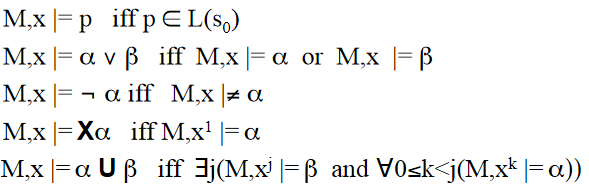
* F a = true U a: a vale in almeno un mondo possibile;
* G a = not F not a: a vale in ogni mondo possibile a partire da quello iniziale.

## Semantica temporale

Un modello M è una tripla <S,L,X> in cui:

* S è l’insieme di mondi dell’universo;
* L:S→ 2^P è una funzione che associa un’interpretazione proposizionale a ogni mondo;
* X:N→ S è una sequenza lineare infinita di mondi.

Per valutare M, si verifica se una formula a è vera per una sequenza x, ciò vale per ogni operatore:



Da ciò si può notare come la logica temporale sia un caso particolare di quella modale, in essa la relazione R è funzionale, cioè che ogni mondo è connesso solamente a uno e uno solo mondo.



## Model Checking

Utilizzando le logiche modali, un sistema può essere rappresentato come un automa a stati finiti in cui i nodi sono i mondi mentre gli archi sono le relative transizioni.

Per i mondi infiniti si utilizza un automa detto LTL e permette di verificare le proprietà nelle logiche temporali lineari.

L’obiettivo è quello di verificare il comportamento di un’agente, come si fa in questi casi?

Si utilizza l’automa di Buchi, un automa con la stessa struttura della controparte a stati finiti solo che permette di accettare sequenze infinite.

Una sequenza infinita w è accettata dall’automa se passa in un stato contrassegnato come finale un numero infinito di volte.

Data una formula LTL f costruita da un set di proposizioni P, è possibile costruire da essa l’automa di Buchi su un alfabeto 2^P che accetta esattamente le sequenze infinite che soddisfano f.

Per provare la validità della formula f, si può mostrare che la negazione è insoddisfacibile, come si fa?

Ebbene, si prova a costruire l’automa di Buchi per not f e mostrare che il linguaggio accettato è vuoto.

Il problema di validità per LTL è PSpace completo.

## SPIN

SPIN è un sistema che verifica le proprietà sul modello, come funziona? Essenzialmente, data una formula f, la trasforma in un automa a stati finiti non deterministico che riconosce stringhe infinite.

Come effettua la verifica?

Partendo da not f, l’automa riconosce i modelli di not f e li combina coi modelli del sistema attraverso il prodotto.

Dopodichè, se viene trovata un’esecuzione del modello che è anche un’esecuzione di not f, allora il sistema falsifica f.

QUesto sistema di verifica equivale alla dimostrazione per refutazione.