Laurea Specialistica in Informatica
a.a. 2007-2008

Interazione Uomo-Macchina II:

Interfacce Intelligenti

Fiorella de Rosis

Laboratori di
Irene Mazzotta e Nicole Novielli

Programma del Corso

Introduzione
Unità 1: **Ragionamento logico:**
- **Formalizzazione**
- **Risoluzione**

Unità 2: Generazione di linguaggio naturale
- Teorie
- Metodi

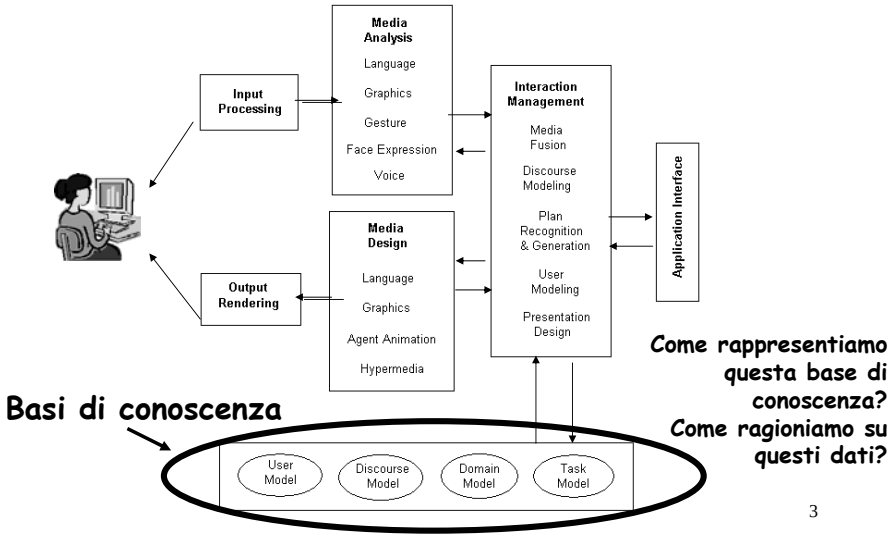
Unità 3: Ragionamento incerto
- Reti Causali Probabilistiche
- Reti dinamiche

Unità 4: Modelli di Utente
- Modelli logici
- Modelli con incertezza

Unità 5: Comprensione del linguaggio naturale
Unità 6: *Simulazione di dialoghi*
- Modelli basati su ATN
- Modelli basati su IS

Unità 7: Affective Computing

Architettura di una interfaccia intelligente



1. Formalizzazione

Perché è importante imparare a formalizzare

I sistemi di simulazione di dialoghi
e di generazione di messaggi in linguaggio naturale

lavorano

Rappresentando internamente in forma simbolica frasi in LN
(e viceversa)

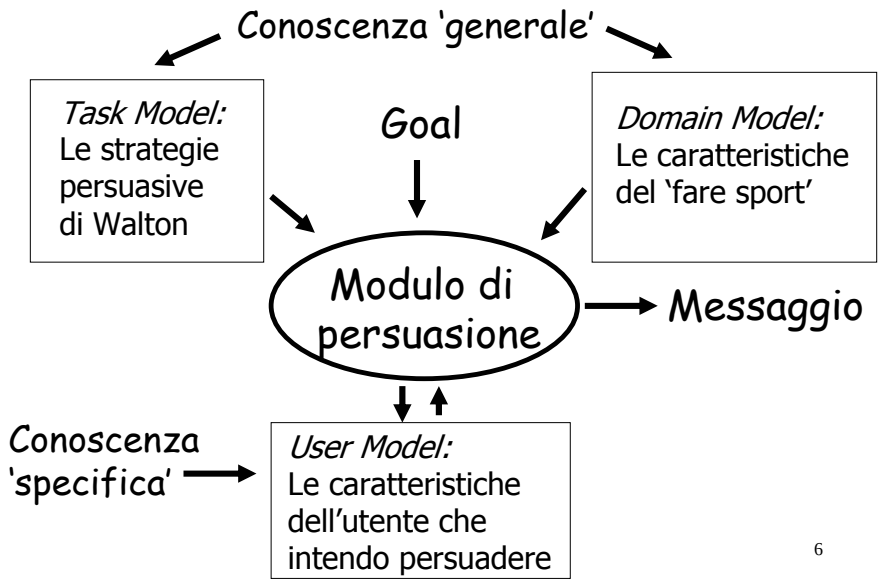
ragionando sulla forma simbolica
e
traducendo il risultato del ragionamento in messaggi in linguaggio naturale o multimediali

Riprendiamo l'Es 2.4: Un dialogo di 'advice-giving'

S: Dovresti andare a correre, Giuseppe!
U: Perché?
S: Perché sei giovane, ci tieni alla tua salute, e correre fa bene alla salute.
U: Ma ti pare che, a vent'anni, già devo pensare alla salute?
S: Secondo me si. Ma comunque, correre aiuta anche a tenersi in forma.
U: E chi te l'ha detto?
S: Lo dicono studi epidemiologici svolti in diversi paesi, da istituti di ricerca qualificati.
U: Ma io detesto correre.
…

Vediamo come formalizzare la conoscenza necessaria per simulare un dialogo di questo tipo

Fonti di conoscenza



Conoscenza 'generale'

1. Task model: le strategie persuasive di Walton

a. *Appeal to positive consequences*

"Se ritieni che compiere una determinata azione comporti conseguenze importanti per te e sei in grado di compierla, dovresti farla".

b. *Appeal to negative consequences*

"Se ritieni che compiere una determinata azione comporti conseguenze che desideri evitare e puoi evitare di compierla, dovresti farlo".

2. Domain model

"Fare sport fa bene alla salute e alla forma fisica. Fare una vita sedentaria aumenta il rischio di ingrassare e di perdere tono muscolare".

"Il running è un particolare tipo di sport. Tutti i giovani senza particolari problemi di salute sono in grado di fare running".

… Conoscenza 'generale'

3. Discourse model

"Fare sport fa bene alla salute e alla forma fisica", *oppure*
"Fare un po' di sport ti fa sentire bene e in forma"… ecc
(stesso contenuto, diversa 'realizzazione linguistica', cioè diverso stile).

Conoscenza 'specifica'

4. User model

"Giuseppe è giovane,
non ha particolari problemi di salute
e tiene alla sua forma fisica.
Come molti giovani,
non si preoccupa invece, per ora, della sua salute."

I goal possono essere diversi:

Posso consigliare a Giuseppe di andare in piscina?
Quale sport è più adatto per lui?
Quali argomenti sono convincenti per lui?
Come formulo il mio messaggio argomentativo?

Abbiamo bisogno di un metodo per

- *Formalizzare* discorsi in linguaggio naturale e
- *Ragionare* sulla 'conoscenza' così espressa per derivare nuovi fatti (i quesiti a cui intendiamo rispondere).

Il 'calcolo degli enunciati, in genere, non basta: non permette di unificare 'conoscenza generale' e 'conoscenza specifica'.

Utilizzeremo una *teoria del prim'ordine* per formalizzare le diverse componenti della base di conoscenza:

- il contenuto di un modello di utente,
- le strategie di generazione di discorsi in linguaggio naturale, …
- gli 'atti comunicativi' che compongono il dialogo,

…

e per ragionare su queste 'basi di conoscenza'.

Notazioni

Termini:

Variabili (denotate con caratteri minuscoli): x, y, z, …

Costanti (denotate con caratteri maiuscoli): A, B, C, …

Funtori (denotati con stringhe del tipo F°) applicati a uno o più termini:

Father°(y) per 'il padre di y', Father°(MARIA)

Formule atomiche:

Predicati n-ari (denotati con stringhe che iniziano con una maiuscola) applicati a n *termini*, che hanno valore T/F:

Father(x,y) per 'x è il padre di y'; Father(PIETRO,MARIA)

Connettivi: $\wedge, \vee, \neg, \rightarrow$

Quantificatori: $\forall$, $\exists$

Formule:

combinazioni appropriate di formule atomiche con connettivi e quantificatori

Dato un particolare dominio applicativo,

occorre scegliere in modo opportuno
gli elementi del linguaggio
e tradurre correttamente
le frasi in linguaggio naturale nel dominio
in formule 'valide',
facendo sì che
la loro interpretazione sia 'vera'

Esempi: Frasi in linguaggio naturale

Formalizziamo soltanto l'*oggetto* degli 'atti comunicativi'

Da Es 2.5

Oz: Il mio nome è Valentina

Name(S,Valentina) … S é lo 'Speaker'

Da Es 2.5

*Oz: Mangiare a orari fissi **aiuta ad** evitare di saltare i pasti*

$\forall x\ (\text{Person}(x) \wedge \text{EatAtFixedTime}(x)) \rightarrow \text{AvoidJumpMeal}(x)$*

↑

conoscenza 'implicita'

Notare l'affermazione di carattere generale rappresentata con un quantificatore universale

* Vedremo il ruolo dell'incertezza!!

… continuando gli esempi

Da Es 2.3

S: That's United flight 659; it arrives back into San Jose at 6.42

Company(F, United) $\wedge$ Number(F, 659) $\wedge$ Arrives(F, SanJose, 6.42)

Da Es 2.5

Oz: Sono qui per darti dei suggerimenti su come migliorare la tua dieta

Goal S (Suggest(S, HowToImprove°(diet)))

Speaker -> S

'sono qui per' -> Goal

'darti dei suggerimenti' -> Suggest

'migliorare la dieta' -> HowToImprove°(diet))

… e come rappresenteremmo 'migliorare la *tua* dieta?'

Notare che questa non è una formula in un linguaggio del prim'ordine!

Perché???

Riprendiamo a ragionare sull'Es 2.4

L' esempio nel campo della persuasione

- **Task model: le strategie persuasive di Walton**

a. Appeal to positive consequences

"Se *ritieni che* … per ora omettiamolo…
compiere una determinata azione comporti conseguenze
Implies(a,g), a denota l'azione e g le conseguenze
importanti per te
Likes(x,g), x denota l'interlocutore
e sei in grado di compierla,
CanDo(x,a),
dovresti farla"
ShouldDo(x,a).

Denoto:
Con a l'azione
Con x chi la compie
Con g una delle conseguenze
Con il predicato Implies il verbo 'comportare'
Con il predicato Likes l'attributo 'importante'
Con il predicato CanDo il verbo 'essere in grado di compiere'

L'intera frase (valida per chiunque):

$\forall x\ \forall a\ \forall g\ \ ((\text{Implies}(a,g) \wedge \text{Likes}(x,g) \wedge \text{CanDo}(x,a)) \rightarrow (\text{ShouldDo}(x,a))$

b. *Appeal to negative consequences*

"Se ritieni che compiere una determinata azione comporti conseguenze che desideri evitare e sei in grado di evitarle di compierla, dovresti farlo".

$\forall x\ \forall a\ \forall g\ ((\text{Implies}(a,g) \wedge \neg\ \text{Likes}(x,g) \wedge \text{CanAvoid}(x,a)) \rightarrow (\text{ShouldAvoid}(x,a))$

- **Domain model**

"Fare sport fa bene alla salute e alla forma fisica."

$\forall s\ \text{Sport}(s) \rightarrow \text{Implies}(s,\text{GoodHealth})$

$\forall s\ \text{Sport}(s) \rightarrow \text{Implies}(s,\text{GoodShape})$

"Il running è un particolare tipo di sport". Tutti i giovani senza particolari problemi di salute sono in grado di fare running.

Sport(R)

$\forall x\ \forall s\ (\text{Sport}(s) \wedge \text{Person}(x) \wedge \text{Young}(x) \wedge \text{Healthy}(x)) \rightarrow \text{CanDo}(x,s))$

- **User model**

"Giuseppe è giovane,
non ha particolari problemi di salute
e tiene alla sua forma fisica.
Come molti giovani,
non si preoccupa invece, per ora, della sua salute."
Young(G)
Healthy(G)
Likes(G,GoodShape)
……

Notare che lo user model contiene (in questo caso) soltanto conoscenza specifica. Ma, come vedremo in seguito, può essere organizzato in modo da permettere di combinare conoscenza specifica con conoscenza generale, derivando nuovi *fatti* da fatti noti.

Facciamo il lavoro inverso:

Da formule a discorsi

Traduciamo queste formule in buone frasi in linguaggio naturale:

$\forall x\ \forall y\ (Person(x) \wedge Hungry(x) \wedge Food(y)) \rightarrow ShouldEat(x,y))$

"Bisognerebbe mangiare solo se si ha fame" (Es 2.5)

$\forall x\ HealthyDiet(x) \rightarrow FewFats(x)$

"Limitare la dose di grassi … rappresenta un elemento fondamentale di una dieta sana" (Es 2.5).

Nella generazione di messaggi in linguaggio naturale, occorre

- rappresentare in forma simbolica la struttura del messaggio e le sue componenti e
- Tradurre questa struttura in un messaggio in linguaggio naturale.
  Dato: $\forall x\ \forall y\ (Person(x) \wedge Hungry(x) \wedge Food(y)) \rightarrow ShouldEat(x,y))$
  Tradurlo in *'Bisognerebbe mangiare solo se si ha fame'*
  (o cose più complesse…)

Nella simulazione di dialoghi, occorre

- rappresentare in forma simbolica le 'mosse' del sistema e tradurle in linguaggio naturale
  Input: $\forall x\ \forall y\ (Person(x) \wedge Hungry(x) \wedge Food(y)) \rightarrow ShouldEat(x,y))$
  Output: *'Bisognerebbe mangiare solo se si ha fame'*

  e

- interpretare le mosse dell'utente (in linguaggio naturale) traducendole in forma simbolica
  Input: *'Ma io ho sempre fame!'*
  Output: $\forall z\ Time(z) \rightarrow Hungry(U,z)$

2. Il sistema deve essere capace di ragionare, sulla conoscenza formalizzata

Per scopi diversi:
Interpretare la mossa dell'utente,
Aggiornare il modello di utente nel corso di un dialogo,
Aggiornare la 'storia' del dialogo,
Scegliere la prossima 'mossa' da effettuare, ecc.

Ci sono molti altri modi di fare inferenza 'logica'
Ne studieremo, in particolare, uno:
quello basato sul

Principio di Risoluzione.

Abbiamo bisogno di un metodo per derivare nuovi fatti da fatti noti, nella base di conoscenza (sull'utente, sul dominio, sulle azioni da compiere).

Ad esempio:

- da	$Age(U,24)$ derivo $Young(U)$
- da	$Sport(s) \rightarrow Implies(s, GoodHealth)$
e	$Sport(AIKIDO)$
devo derivare	$Implies(AIKIDO, GoodHealth)$

… o cose più complesse.

Sceglieremo uno dei possibili sistemi logici: quello basato sul *principio di risoluzione*, che non elabora formule, ma *clausole.*

… **Continuiamo a lavorare**
sull'esempio nel campo della persuasione
formalizzato in termini di linguaggio del prim'ordine…

$\forall x\ \forall a\ \forall g$ ((Implies(a,g) $\wedge$ Likes(x,g) $\wedge$ CanDo(x,a)) $\rightarrow$ (ShouldDo(x,a))
$\forall x\ \forall a\ \forall g$ ((Implies(a,g) $\wedge \neg$ Likes(x,g) $\wedge$ CanAvoid(x,a)) $\rightarrow$ (ShouldDo(x,a))
$\forall s$ Sport(s) $\rightarrow$ Implies(s,GoodHealth)
$\forall s$ Sport(s) $\rightarrow$ Implies(s,GoodShape)
$\forall x\ \forall s$ (Sport(s) $\wedge$ Person(x) $\wedge$ Young(x) $\wedge$ Healthy(x)) $\rightarrow$ CanDo(x,s))

Sport(R)

Young(G)
Healthy(G)
Likes(G,GoodShape)
……

Possibili quesiti:

Posso consigliare a Giuseppe di andare in piscina?
Quale sport è più adatto per lui?
Quali argomenti sono convincenti per lui?
Come formulo il mio messaggio argomentativo?

Espressioni logiche in forma di clausole

Letterale: una formula atomica, eventualmente negata:
Esempi: P, $\neg$Q, $\neg$ R(x), F(x).
P, F(y) sono letterali 'positivi'; $\neg$Q, $\neg$ R(x) sono letterali 'negativi'

Clausola: una disgiunzione di letterali, rappresentata con la notazione degli insiemi:
Esempi: {P, $\neg$Q} è una clausola che sta per P $\vee$ $\neg$Q;
{$\neg$R(x), F(x)} è una clausola che sta per {$\neg$R(x), F(x)},
e cioè per R(x) $\rightarrow$ F(x)

Una *clausola di Horn* ha al massimo un letterale positivo

Liste di clausole

Una lista di clausole è intesa come *congiunzione* delle clausole in essa comprese.
Esempio: la lista
{$\neg$R(x), F(x)}
{R(A)}
equivale a: {$\neg$R(x), F(x)} $\wedge$ {R(A)}

In sostanza, quindi, una lista di clausole corrisponde ad una formula in *forma normale congiuntiva.*
Nell'esempio precedente:
(R(x) $\rightarrow$ F(x)) $\wedge$ R(A)

Conversione da formule a clausole (1)

```
Procedure Convert (x):

1     Begin        x <- Implications_out (x)
2                  x <- Negations_in (x)
3                  x <- Standardize_variables (x)
4                  x <- Existentials_out (x)
5                  x <- Universals_out (x)
6                  x <- Disjunctions_in (x)
7                  x <- Operators_out (x)
8                  x <- Rename_variables (x)
```

Genesereth & Nilsson 1986

Conversione da formule a clausole (2)

Esempi:

Implications_out (x): $R(x) \rightarrow F(x)$ diventerebbe:

$\neg R(x) \vee F(x)$

Negations_in (x): $\neg(R(x) \vee F(x))$ diventerebbe

$\neg R(x) \wedge \neg F(x)$,

$\neg \forall x\ \varphi$ diventerebbe $\exists x \neg \varphi$,

$\neg \exists x\ \varphi$ diventerebbe $\forall x \neg \varphi$, …

Conversione da formule a clausole (3)

Esempi:

Existentials_out (x):

se l' $\exists$ non occorre nell'ambito di un $\forall$, può essere eliminato, sostituendo tutte le occorrenze della sua variabile con una costante (*costante di Skolem*):

$\exists x\ R(x)$ diventa $R(Sk)$

se l' $\exists$ occorre nell'ambito di un $\forall$, può essere eliminato, sostituendo tutte le occorrenze della sua variabile con un termine che corrisponde ad una funzione applicata alle altre variabili nel campo dell' $\forall$ (*funzione di Skolem*):

$\forall x\ \exists y\ P(x,y)$ diventa $\forall x\ P(x,Sk°(x))$

Conversione da formule a clausole (4)

Universals_out (x): tutti i $\forall$ vengono eliminati

$\forall x\ P(x,F(x))$ diventa $P(x,F(x))$

Disjunctions_in (x); la formula viene trasformata in forma normale congiuntiva

$(P \wedge Q) \vee S$ diventa $(P \vee S) \wedge (Q \vee S)$

$(P \wedge Q) \vee R \vee S$ diventa $(P \vee R \vee S) \wedge (Q \vee R \vee S)$, …

Operators_out (x); le formule ottenute vengono trascritte in forma di clausole

$(P \vee R \vee S) \wedge (Q \vee R \vee S)$ diventa $\{P, R, S\}$; $\{Q, R, S\}$

Rename_variables (x): nessuna variabile deve apparire in più di una clausola

…Trasformiamo un sottoinsieme del nostro esempio sulla persuasione…

$\forall x\ \forall a\ \forall g\ \ ((Implies(a,g) \wedge Likes(x,g) \wedge CanDo(x,a)) \rightarrow (ShouldDo(x,a))$

$\forall s\ Sport(s) \rightarrow Implies(s,GoodShape)$

$\forall x\ \forall s\ (Sport(s) \wedge Person(x) \wedge Young(x) \wedge Healthy(x)) \rightarrow CanDo(x,s))$

Sport(R)

Young(G)

Healthy(G)

Likes(G,GoodShape)

$\{\neg Implies(a,g), \neg Likes(x,g), \neg CanDo(x,a), ShouldDo(x,a)\}$

implication-out ; universals-out

$\{\neg Sport(s), Implies(s,GoodShape)\}$

implication-out ; universals-out

$\{\neg Sport(s), \neg Person(x), \neg Young(x), \neg Healthy(x), CanDo(x,s)\}$

implication-out ; universals-out

$\{Sport(R)\}$

$\{Young(G)\}$

$\{Healthy(G)\}$

$\{Likes(G,GoodShape)\}$

Unificazione

E' il processo mediante il quale si determina se due espressioni (formule o letterali) possono essere rese identiche mediante un'opportuna sostituzione di valori per le variabili in esse contenute.

Esempio: Implies(a,g) e Implies(R, GoodHealth)
sono unificabili con la sostituzione: {a/R, g/GoodHealth}

Un altro esempio di *unificazione*:
La sostituzione {x/A, y/B, F°(A,B)/C} rende identiche ('unifica') le due formule: (P(x,y,F°(x,y)) e P(A,B,C)

Principio di Risoluzione

Date: una clausola Φ che contiene il letterale φ e
una clausola Ψ che contiene il letterale $\neg\psi$
se φ e ψ sono *unificabili* con una sostituzione λ
e Φ', Ψ' sono le clausole ottenute applicando λ (rispettivamente) a Φ e Ψ,
allora
si può dedurre, da Φ e Ψ, una nuova clausola:
$((\Phi' - \{\varphi\}) \cup (\Psi' - \{\neg\psi\}))\ \lambda$

Esempio:
1 $\{\neg P(x), Q(x)\}$
2 $\{\neg Q(A), R(B)\}$
_____
3 $\{\neg P(A), R(B)\}$ (1,2) con $\lambda = \{x/A\}$

Modus Ponens

Notiamo che il modus ponens è un caso particolare del principio di risoluzione:

$P(x) \rightarrow Q(x)$	$\{\neg P(x), Q(x)\}$
$P(A)$	$\{P(A)\}$
_____	_____
$Q(A)$	$Q(A)$

Risoluzione 'grezza'

Input: un set Δ di n clausole C(k), con k = 1,…, n
1 i = 1; j = 2;
2 Prova a risolvere C(i) con C(j): questo è possibile, ottieni una clausola C(n+1) che aggiungi a Δ; poni n=n+1;
altrimenti: i = i+1 e ripeti la 2 finché i=j-1;
3 i = 1; j = j+1;
4 ripeti da 2 *finché la condizione di 'stop' è verificata oppure j=n*.

Nota: la condizione di stop varia a seconda degli obiettivi per cui la procedura di risoluzione è utilizzata.

Un esempio di Risoluzione 'Grezza'

1 {¬Implies(a,g), ¬Likes(x,g), ¬CanDo(x,a), ShouldDo(x,a)}
2 {¬Sport(s), Implies(s,GoodShape)}
3 {¬Sport(s), ¬Person(x), ¬Young(x), ¬Healthy(x), CanDo(x,s)}
4 {Sport(R)}
5 {Young(G)}
6 {Healthy(G)}
7 {Likes(G,GoodShape)}

---

8 {¬Likes(x,GoodShape),¬CanDo(x,s), ShouldDo(x,s), ¬Sport(s)} (1,2)
9 {¬Implies(s,g), ¬Likes(x,g), ShouldDo(x,s), ¬Sport(s), ¬Person(x), ¬Young(x), ¬Healthy(x)} (1,3)
10 {Implies(R, GoodShape)} (2,4)
11 {¬Person(x), ¬Young(x), ¬Healthy(x), CanDo(x,R)} (3,4)
12 {¬Sport(s), ¬Person(G), ¬Healthy(G), CanDo(G,s)} (3,5)
13 {¬Sport(s), ¬Person(G), ¬Young(G), CanDo(G,s)} (3,6)
14 {¬Implies(s,GoodShape), ¬CanDo(G,s), ShouldDo(G,s)} (1,7)

.....

15 … ecc

Strategie di Risoluzione

Per rendere più veloce la ricerca di soluzioni, si possono applicare due classi di strategie:

a. *Cancellazione di clausole* dall'insieme esaminato

b. *Definizione di un ordine* nella considerazione delle clausole

Strategie di Cancellazione

1. Eliminazione di letterali puri: letterali che non compaiono in modo complementare (negato) in nessun'altra clausola
   1. {¬Mother(x,y), Parent(x,y)}
   2. {Father(P,F)}
   3. {Mother(A,F)}
   4. {¬Parent(x,F), Ans(x)}

   Father(P,F) è un letterale puro.
2. Eliminazione di tautologie: clausole che contengono una coppia di letterali complementari

   {¬F(x,y), P(x,y), F(x,y)}
1. Eliminazione di sussunzioni: una clausola Φ subsume una clausola Ψ iff esiste una sostituzione λ che rende Φ inclusa in Ψ: $\Phi\lambda \subseteq \Psi$

   {¬Mother(x,y), Parent(x,y)} subsume

   {¬Mother(x,F), ¬Female(x), Parent(x,F)}, per la sostituzione y/F

Una strategia basata sull'ordine: Risoluzione Ordinata

La risoluzione è consentita soltanto sul *primo letterale* di ciascuna delle due clausole.

Esempio:

1 {P,Q}
2 {¬P,R}
3 {¬Q,R}
4 {¬R}

---

5 {Q,R} 1,2
6 {R} 3,5
7 {}4,6

Vediamo come usare questa strategia

Risoluzione Ordinata nel nostro esempio

1 {¬Implies(a,g), ¬Likes(x,g), ¬CanDo(x,a), ShouldDo(x,a)}
2 {¬Sport(s), Implies(s,GoodShape)}
3 {¬Sport(s), ¬Person(x), ¬Young(x), ¬Healthy(x), CanDo(x,s)}
4 {Sport(R)}
5 {Person(G)}
6 {Young(G)}
7 {Healthy(G)}
8 {Likes(G,GoodShape)}

---

9 {¬Implies(R,GoodShape) } (2,4)
10 {¬Person(x), ¬Young(x), ¬Healthy(x), CanDo(x,R)} (3,4)

---

11 {¬Likes(x,GoodShape), ¬CanDo(x,R), ShouldDo(x,R)} (1,9)
12 {¬Young(G), ¬Healthy(G), CanDo(G,R)} (5,10)
13 {¬CanDo(G,R), ShouldDo(G,R)} (8,11)
14 {¬Healthy(G), CanDo(G,R)} (6,12)
15 {CanDo(G,R)} (7,14)
16 {ShouldDo(G,R)} (13,15)

ci fermiamo? Dipende…

Possibili Usi Della Risoluzione

1.Dimostrazione di inconsistenza.

Condizione di stop = derivazione della clausola vuota

Se da un insieme Δ si ottiene, per risoluzione,
la clausola vuota,
allora Δ è inconsistente.

Infatti:

A ∧¬A si rappresenta, in forma di clausole, come

1 {A}
2 {¬A} e permette di dedurre:

---

3 {}

Possibili Usi Della Risoluzione

2.Risposta a domande T/F

Dato un insieme Δ e una formula ω,
si può rispondere al quesito:
"Δ implica ω?"
verificando se esiste una contraddizione fra Δ e ¬ω,
cioè se l'insieme di clausole {Δ ∪ ω} è consistente.

Per far ciò:
si aggiunge ¬ω all'insieme Δ;
si risolve;
se si trova la clausola vuota, si può concludere con una risposta positiva (Δ implica ω),
altrimenti il risultato resta ignoto (per il Teorema di Completezza)

Possibili Usi Della Risoluzione

3. Quesiti del tipo 'Fill-in the blank'

Dato un insieme Δ e una formula ω che contengono la variabile x,
si può rispondere al quesito:
"Per quale valore della variabile x Δ implica ω(x)?"
verificando qual è (se esiste) il valore di x
che produce una contraddizione fra Δ e ¬ω(x),
cioè che rende soddisfacibile l'insieme di clausole {Δ ∪ ω(x)}.

Per far ciò:
si aggiunge all'insieme Δ la clausola {¬ω(x), Ans(x)};
si risolve, ponendo come condizione di stop la deduzione di una formula con il solo letterale 'Ans';
se si deduce la clausola {Ans(A)}, si può concludere che
"Δ implica ω(x) per x/A"
altrimenti il risultato resta ignoto (per il Teorema di Completezza)

Quando utilizzare le tre modalità? Qualche esempio

1. *Verifiche di consistenza sulla conoscenza 'generale'* (relativa al dominio o al task), per sua natura 'frammentaria'
2. *Aggiornamento del modello dell'utente*: derivazione di caratteristiche dell'utente non note, dato un insieme di caratteristiche note e un insieme di regole di associazione fra caratteristiche.
3. *Aggiornamento della conoscenza sul dominio*.
4. *Individuazione dell'azione da suggerire* fra diverse azioni possibili; o dell'informazione da fornire, fra diverse alternative.

Uso della modalità 'risposta a domande T/F'

Sistema per il suggerimento di programmi televisivi.

So che l'utente è di genere femminile, ha 23 anni e sportiva.

Posso dedurre che le interessa vedere il derby Roma-Lazio?

Uso della modalità 'fill-in the blank'

Sistema per il suggerimento di programmi televisivi.

So che l'utente è di genere femminile, ha 23 anni e sportiva.

Quali spettacoli posso dedurre che le interessa vedere?

Quali spettacoli (molto probabilmente) detesta?

Verifica di Consistenza in un modello di Utente

Ho un modello di utente rappresentato in modo logico, che aggiorno dinamicamente nel corso dell'interazione. Devo verificare, periodicamente, la sua consistenza.

Il modello contiene una *componente specifica* e una *generale* (che permette di inferire nuovi fatti, da quelli memorizzati):

Female(M)
Age(M, 23)
MakeSport(M) *componente specifica*
Likes(M, Reading)
Sport(Football)
Show(Football)
$\forall$s (Show(s) $\rightarrow$ Likes(M,s)) *generale*
Lonely(M)
$\forall$x $\forall$s (Female(x) $\wedge$ Likes(x,Reading) $\wedge$ Lonely(x)) $\wedge$ Sport(s)) $\wedge$ Show(s))$\rightarrow$ $\neg$Likes(x,s)

Female(M)
Age(M, 23)
MakeSport(M)
Likes(M, Reading)
Sport(Football)
Show(Football)
$\forall$s (Show(s) $\rightarrow$ Likes(M,s))
Lonely(M)
$\forall$x $\forall$s (Female(x)$\wedge$Likes(x,Reading)$\wedge$Lonely(x))$\wedge$Sport(s))$\wedge$Show(s))$\rightarrow$ $\neg$Likes(x,s)

In clausole:
1 {Female(M)}
2 {Age(M, 23)}
3 {MakeSport(M)}
4 {Likes(M, Reading)}
5 {Sport(Football)}
6 {Show(Football)}
7 {$\neg$Show(s), Likes(M,s)}
8 {Lonely(M)}
9 {$\neg$Female(x), $\neg$Likes(x,Reading), $\neg$Lonely(x)), $\neg$Sport(s), $\neg$Show(s), $\neg$Likes(x,s)}

Derivo prima $\neg$Likes(M,Football), *con vari passaggi, da 1,4,8,5,6,9*
Poi Likes(M,Football), *da 6,7*
E quindi la clausola vuota. *Il modello è contraddittorio*

Uso della modalità 'risposta a domande T/F'

Female(M)
Age(M, 23)
MakeSport(M)
Likes(M, Reading)
Sport(Football)
Show(Football)
Lonely(M)
$\forall x\ \forall s$ (Female(x)$\wedge$Likes(x,Reading)$\wedge$Lonely(x))$\wedge$Sport(s))$\wedge$Show(s))$\rightarrow \neg$Likes(x,s)
A Maria piace il football?

In clausole:
1 {Female(M)}
2 {Age(M, 23)}
3 {MakeSport(M)}
4 {Likes(M, Reading)}
5 {Sport(Football)}
6 {Show(Football)}
8 {Lonely(M)}
9 {$\neg$Female(x), $\neg$Likes(x,Reading), $\neg$Lonely(x)), $\neg$Sport(s), $\neg$Show(s), $\neg$Likes(x,s)}
10 {$\neg$Likes(M, Football)}

Derivo prima $\neg$Likes(M,Football), *con vari passaggi, da 1,4,8,5,6,9*
E quindi la clausola vuota.
concludo che 'A Maria non piace il football'

Uso della modalità 'fill-in the blank'

Female(M)
Age(M, 23)
MakeSport(M)
Likes(M, Reading)
Movie(Romance)
Movie(Cartoons)
Lonely(M)
$\forall x\ \forall z$ (Female(x)$\wedge$Likes(x,Reading)$\wedge$Lonely(x))$\wedge$Movie(z))$\rightarrow$ Likes(x,z)
Quali spettacoli piacciono a Maria?

In clausole:
1 {Female(M)}
2 {Age(M, 23)}
3 {MakeSport(M)}
4 {Likes(M, Reading)}
5 {Movie(Romance)}
6 {Movie(Cartoons)}
8 {Lonely(M)}
9 {$\neg$Female(x), $\neg$Likes(x,Reading), $\neg$Lonely(x)), $\neg$Movie(z), Likes(x,s)}
10 {$\neg$Likes(M, z), Ans(z)}

Derivo prima Ans(Romance) e poi Ans(Cartoons)
concludo che 'A Maria piacciono i film d'amore e i cartoni animati'

3. Ricerca delle soluzioni nello spazio degli stati

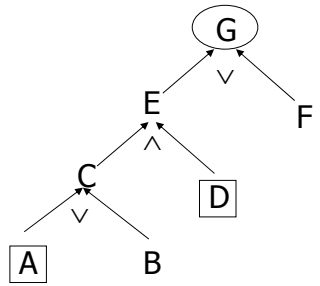
Due direzioni possibili:

Si parte dall'obiettivo che s'intende raggiungere ('stato goal') e si applicano gli inversi degli 'operatori' di trasformazione disponibili, fino ad arrivare ad uno stato che corrisponde allo 'stato iniziale (*backward*).

Si parte dallo stato iniziale, e si applicano gli operatori fino a raggiungere lo stato 'goal' (*forward*).

Ricerca delle soluzioni nello spazio degli stati

$A \vee B \rightarrow C$
$C \wedge D \rightarrow E$
$E \vee F \rightarrow G$



Stato iniziale: A, D
Goal: G

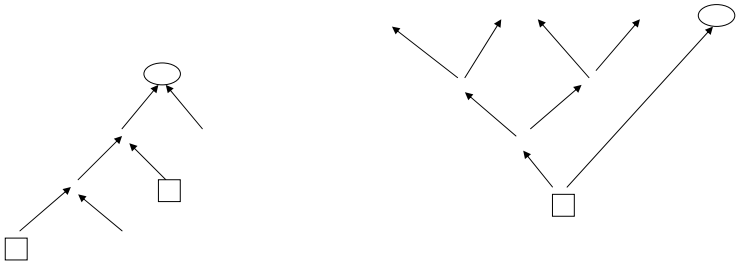
Ricerca backward: G?; E? or F?; E?: C? and D?; D; C?: A? or B?; A
G = T

Ricerca forward: A; C; D; E; G
G = T

Ricerca delle soluzioni nello spazio degli stati

Il Principio di Risoluzione permette di simulare i due metodi di ricerca; la convenienza del metodo dipende dalla struttura del problema.

Esempio:



Simulazione di Ragionamento Backward vs Forward

La risoluzione ordinata consente di simulare ricerche di soluzioni 'backward' o 'forward' a seconda di come sono ordinati i letterali nelle clausole considerate.

Facciamo l'ipotesi che tutte le clausole in Δ siano *clausole di Horn*. Possiamo ordinare gli elementi di ogni clausola in uno dei due modi seguenti:

a. *Ordinamento forward:* il letterale positivo si trova *alla fine* della clausola
{¬Mather(x,y), Parent(x,y)}

b. *Ordinamento backward:* il letterale positivo si trova *all'inizio* della clausola
{Parent(x,y), ¬Mather(x,y)}

Come funziona il ragionamento forward

Da dati iniziali positivi
Si deducono dati intermedi positivi
Che, alla fine, si risolvono con il goal negato
(propagazione 'in avanti' delle evidenze disponibili)

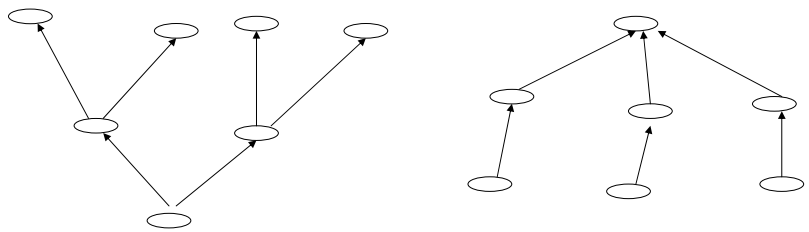
Come funziona il ragionamento backward

Da dati iniziali negativi (il goal negato)
Si deducono dati intermedi negativi (sobgoal negati)
Fino a risolvere con le evidenze disponibili.

(deduzione per *decomposizione del goal in subgoal*)

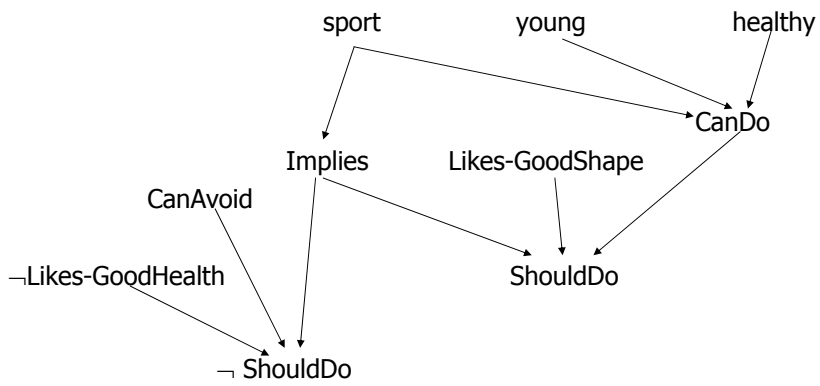
Simulazione di Ragionamento Backward vs Forward

L'ordinamento più conveniente dipende dalla struttura del grafo di ricerca:



backward forward

Struttura del grafo nel problema di persuasione



… Il problema della persuasione in modo backward…

1 { ShouldDo(x,a) ,¬Implies(a,g), ¬Likes(x,g), ¬ CanDo(x,a)}
2 { Likes(x,g), ¬Implies(a,g), ¬CanAvoid(x,a), ¬ShouldDo(x,a)}
3 { Implies(s, GoodShape), ¬ Sport(s)}
4 { Implies(s, GoodHealth),¬Sport(s),}
5 { CanDo(x,s)),¬Sport(s), ¬Young(x), ¬Healthy(x)}
6 {Sport(R)}
7 {Sport(A)}
8 {Young(G)}
9 {Healthy(G)}
10 {Likes(G, GoodShape)}
11 {¬Likes(G,GoodHealth)}
12 {¬ShouldDo(G,a), Ans(a)}

---

13 {¬Implies(a,g), ¬Likes(G,g), ¬ CanDo(G,a), Ans(a)} (1,12)
14 { ¬ Sport(s), ¬Likes(G,GoodShape), ¬ CanDo(G,s), Ans(s)} (3,13)
15 { ¬ Sport(s), ¬Likes(G,GoodHealth), ¬ CanDo(G,s), Ans(s)} (4,13)
16 { ¬Likes(G,GoodShape), ¬ CanDo(G,R), Ans(R)} (6,14)
17 { ¬Likes(G,GoodShape), ¬ CanDo(G,A), Ans(A)} (7,14)
18 { ¬Likes(G,GoodHealth), ¬ CanDo(G,R), Ans(R)} (6,15)
19 { ¬Likes(G,GoodHealth), ¬ CanDo(G,A), Ans(A)} (7,15)
20 { ¬ CanDo(G,R), Ans(R)} (10,16)
21 { ¬ CanDo(G,R), Ans(A)} (10,17)
22 { ¬Sport(R), ¬Young(G), ¬Healthy(G), Ans(R)} (5,20)
23 { ¬Sport(A), ¬Young(G), ¬Healthy(G), Ans(A)} (5,21)
24 { ¬Young(G), ¬Healthy(G), Ans(R)} (6,22)
25 { ¬Young(G), ¬Healthy(G), Ans(A)} (7,22)
26 { ¬Healthy(G), Ans(R)} (8,24)
27 { ¬Healthy(G), Ans(A)} (8,25)
28 { Ans(R)} (9,26)
29 { Ans(A)} (9,27)

Risultati intermedi negativi.

Mi porto dietro 'Ans' in tutto il processo.

Strategia 'ordinata'

(due riposte perché entrambi gli sport aiutano a stare in forma)

… in modo forward…

1 {¬Implies(a,g), ¬Likes(x,g), ¬ CanDo(x,a), ShouldDo(x,a)}
2 {¬Implies(a,g), ¬CanAvoid(x,a), ¬ShouldDo(x,a), Likes(x,g)}
3 {¬ Sport(s), Implies(s, InShape)}
4 {¬Sport(s), Implies(s, GoodHealth)}
5 {¬Sport(s), ¬Young(x), ¬Healthy(x)), CanDo(x,s))}
6 {Sport(R)}
7 {Sport(A)}
8 {Young(G)}
9 {Healthy(G)}
10 {Likes(G, GoodShape)}
11 {¬Likes(G,GoodHealth)}
12 {¬ShouldDo(G,a), Ans(a)}
13 Implies(R,GoodShape) (3,6)
14 Implies(R,GoodHealth) (4,6)
15 {¬Young(x), ¬Healthy(x)), CanDo(x,R) } (5,6)
16 {Implies(A,GoodShape) } (3,7)
17 {Implies(A,GoodHealth) } (4,7)
18 {¬Young(x), ¬Healthy(x)), CanDo(x,A) } (5,7)
19 {¬Likes(x,GoodShape), ¬ CanDo(x,R), ShouldDo(x,R)} (1,13)
20 {¬CanAvoid(x,R), ¬ShouldDo(x,R), Likes(x,GoodShape)} (2,13)
21 {¬Likes(x,GoodHealth), ¬ CanDo(x,R), ShouldDo(x,R)} (1,14)
22 {¬CanAvoid(x,R), ¬ShouldDo(x,R), Likes(x,GoodHealth)} (2,14)
23 {¬Healthy(G)), CanDo(G,R) } (8,15)
24 {¬Likes(x,GoodShape), ¬ CanDo(x,A), ShouldDo(x,A)} (1,16)
25 {¬CanAvoid(x,A), ¬ShouldDo(x,A), Likes(x,GoodShape)} (2,16)
26 {¬Likes(x,GoodHealth), ¬ CanDo(x,A), ShouldDo(x,A)} (1,17)
27 {¬CanAvoid(x,A), ¬ShouldDo(x,A), Likes(x,GoodHealth)} (2,17)
28 {¬Healthy(G)), CanDo(G,A) } (8,18)
29 {¬ CanDo(G,R), ShouldDo(G,R)} (10,19)
30 {CanDo(G,R) } (9,23)
31 {¬ CanDo(G,A), ShouldDo(G,A)} (10,24)
32 {CanDo(G,A) } (9,28)
33 {ShouldDo(G,R)} (29,30)
34 {ShouldDo(G,A)} (30,31)
35 {ShouldDo(G,A)} (31,32)
36 {Ans(R)} (12,33)
37 {Ans(A)} (12,34)

4. Cosa fare dei risultati

Assumiamo che il quesito precedente abbia prodotto le risposte:

{Ans(R)}

{Ans(A)}

Questi risultati dovranno essere tradotti in un messaggio in linguaggio naturale, del tipo:

"Dovresti andare a correre, Giuseppe! Oppure, potresti fare Aikido!

… Vedremo come questo si può fare, con un 'generatore di linguaggio naturale'

Estendiamo il quesito

Ma

può essere necessario, per aumentare la *forza* del messaggio persuasivo, spiegare anche il *perché del suggerimento,* generando quindi un messaggio più complesso:

"Dovresti andare a correre, Giuseppe! **Il suggerimento**

Sei giovane, sano, ci tieni ad essere in forma, **I dati su U**

e correre aiuta a mantenersi in forma!" **La regola generale**

Come farlo? Occorre, in qualche modo, tenere traccia del processo di derivazione che ha portato ad ottenere la conclusione (o le conclusioni).

1	{¬Implies(a,g), ¬Likes(x,g), ¬ CanDo(x,a), ShouldDo(x,a)}	
2	{¬Implies(a,g), ¬CanAvoid(x,a), ¬ShouldDo(x,a), Likes(x,g)}	
3	{¬ Sport(s), Implies(s, InShape)}	
4	{¬Sport(s), Implies(s, GoodHealth)}	
5	{¬Sport(s), ¬Young(x), ¬Healthy(x)), CanDo(x,s))}	
6	{Sport(R)}	
7	{Sport(A)}	
8	{Young(G)}	
9	{Healthy(G)}	
10	{Likes(G, GoodShape)}	
11	{¬Likes(G,GoodHealth)}	
12	{¬ShouldDo(G,a), Ans(a)}	
13	Implies(R,GoodShape)	(3,6)
14	Implies(R,GoodHealth)	(4,6)
15	{¬Young(x), ¬Healthy(x)), CanDo(x,R) }	(5,6)
16	{Implies(A,GoodShape) }	(3,7)
17	{Implies(A,GoodHealth) }	(4,7)
18	{¬Young(x), ¬Healthy(x)), CanDo(x,A) }	(5,7)
19	{¬Likes(x,GoodShape), ¬ CanDo(x,R), ShouldDo(x,R)}	(1,13)
20	{¬CanAvoid(x,R), ¬ShouldDo(x,R), Likes(x,GoodShape)}	(2,13)
21	{¬Likes(x,GoodHealth), ¬ CanDo(x,R), ShouldDo(x,R)}	(1,14)
22	{¬CanAvoid(x,R), ¬ShouldDo(x,R), Likes(x,GoodHealth)}	(2,14)
23	{¬Healthy(G)), CanDo(G,R) }	(8,15)
24	{¬Likes(x,GoodShape), ¬ CanDo(x,A), ShouldDo(x,A)}	(1,16)
25	{¬CanAvoid(x,A), ¬ShouldDo(x,A), Likes(x,GoodShape)}	(2,16)
26	{¬Likes(x,GoodHealth), ¬ CanDo(x,A), ShouldDo(x,A)}	(1,17)
27	{¬CanAvoid(x,A), ¬ShouldDo(x,A), Likes(x,GoodHealth)}	(2,17)
28	{¬Healthy(G)), CanDo(G,A) }	(8,18)
29	{¬ CanDo(G,R), ShouldDo(G,R)}	(10,19)
30	{CanDo(G,R) }	(9,23)
31	{¬ CanDo(G,A), ShouldDo(G,A)}	(10,24)
32	{CanDo(G,A) }	(9,28)
33	{ShouldDo(G,R)}	(29,30)
34	{ShouldDo(G,A)}	(30,31)
35	{ShouldDo(G,A)}	(31,32)
36	{Ans(R)}	(12,33)
37	{Ans(A)}	(12,34)

Consideriamo la risposta Ans(R). Contrassegniamo in rosso le clausole utilizzate per arrivare alla conclusione

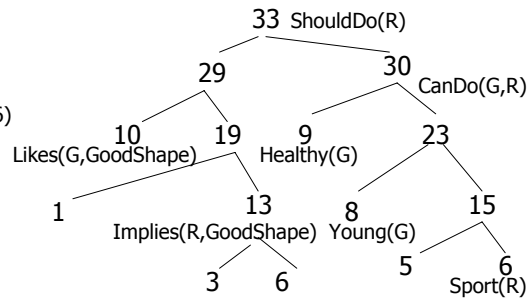
Indichiamo le clausole utilizzate nel processo di inferenza

Partiamo dalla prima risposta: 36 Ans(R) e ripercorriamo, all'indietro, il processo di derivazione di questa clausola, contrassegnando le clausole che hanno contribuito al processo di inferenza: in corsivo le clausole unitarie.

1	**{¬Implies(a,g), ¬Likes(x,g), ¬ CanDo(x,a), ShouldDo(x,a)}**	
3	**{¬ Sport(s), Implies(s, InShape)}**	
5	**{¬Sport(s), ¬Young(x), ¬Healthy(x)), CanDo(x,s))}**	
6	***{Sport(R) }***	
8	***{Young(G) }***	
9	***{Healthy(G) }***	
10	***{Likes(G, GoodShape) }***	
12	**{¬ShouldDo(G,a), Ans(a)}**	
13	**{Implies(R,GoodShape) }**	**(3,6)**
15	**{¬Young(x), ¬Healthy(x)), CanDo(x,R) }**	**(5,6)**
19	**{¬Likes(x,GoodShape), ¬ CanDo(x,R), ShouldDo(x,R)}**	**(1,13)**
23	**{¬Healthy(G)), CanDo(G,R) }**	**(8,15)**
29	**{¬ CanDo(G,R), ShouldDo(G,R)}**	**(10,19)**
30	***{CanDo(G,R) }***	***(9,23)***
33	***{ShouldDo(G,R) }***	***(29,30)***
36	**{Ans(R)}**	**(12,33)**

Ora isoliamo le clausole in corsivo

6	{Sport(RUNNING)}
8	{Young(G)}
9	{Healthy(G)}
10	{Likes(G, GoodShape)}
12	{¬ShouldDo(G,a), Ans(a)}
13	Implies(RUNNING,GoodShape) (3,6)
30	CanDo(G,RUNNING) (9,23)
33	ShouldDo(x,RUNNING) (29,30)
36	Ans(RUNNING) (12,33)



Colleghiamo con 'connettivi linguistici'

ShouldDo((RUNNING)
Likes(G, GoodShape) *and*
Implies(Running,GoodShape).
In addition
CanDo(G,RUNNING) *as*
(Healthy(G) *and* Young(G))

(ho omesso Sport(RUNNING))

Ricapitoliamo il significato di questa unità

Abbiamo studiato:

- come le diverse basi di conoscenza utilizzate all'interno di una interfaccia intelligente (modello di utente, atti comunicativi dei partecipanti ad un dialogo, modelli del task e del dominio) possano essere rappresentate in modo logico,
- come sia possibile distinguere fra 'conoscenza generale' (valida per ragionare in ogni situazione) e 'specifica' (valida per un particolare utente e in un particolare contesto),
- come sia possibile ragionare sulle basi di conoscenza così rappresentate, per rispondere a quesiti di natura diversa: consistenza di modelli di utente, loro aggiornamento, individuazione di possibili risposte a quesiti, ecc
- come sia possibile (oltre ad individuare risposte a quesiti coerenti con il contenuto della base di conoscenza), giustificare le scelte fatte ricostruendo un tracing del ragionamento che ha condotto alla scelta e traducendolo in un messaggio in linguaggio naturale,
- quali sono i limiti di questa rappresentazione e questa forma di ragionamento

Riferimenti

M.R.Genesereth e N J Nilsson: Logical Foundations of Artificial Intelligence. Morgan Kaufman, 1986. Capitoli 5 e 6