Iniziato	giovedì, 3 febbraio 2022, 09:20
Stato	Completato
Terminato	giovedì, 3 febbraio 2022, 09:56
Tempo impiegato	35 min. 57 secondi
Valutazione	**17,50** su un massimo di 18,00 (**97**%)

Domanda 1

Completo

Punteggio ottenuto 4,50 su 4,50

Spiegare come Franklin e Graesser rispondono alla domanda "Is it an Agent or just a Program?", ovvero come distinguere un agente da un generico programma.

Franklin e Graesser definiscono un agente autonomo come un sistema situato in un environment, di cui ne fa parte e sul quale agisce nel tempo, secondo i propri piani e in ottica di un obiettivo futuro. La loro interpretazione quindi lega gli agenti agli ambienti e alle loro interazioni. Sostengono che se si cambia l'environment di un agente, si perde anche l'agente stesso. Un programma, invece, non e' un agente autonomo, perche' i suoi output non sono normalmente prodotti in ottica di obiettivi futuri o in funzione dell'interazione con l'environment.

Commento:

Domanda 2

Completo

Punteggio ottenuto 4,50 su 4,50

Logica temporale Branching-Time: Spiegare come viene modellato il tempo nella logica temporale Branching-Time e come sono fatte le formule di questa logica.

Nella logica temporale Branching-time il tempo viene modellato attraverso una struttura ad albero, dove ogni istante di tempo rappresenta un nodo che puo' avere infiniti istanti successori che a loro volta possono essere seguiti da altre diramazioni. Le formule sono suddivise in State formulas e Path formulas.

Le path formulas riguardano i cammini infiniti della struttura ad albero e la loro sintassi e' la seguente:
a
pi v p

-pi

Xpi

pi U p
dove a e' una state formula e pi e p sono path formulas e X e U sono operatori modali con la semantica di LTL. Il significato di X pi e' quindi "nexttime pi", ovvero pi sara' vero al prossimo istante. Il significato di pi U p ad un certo istante t e' "pi until p", ovvero pi e' vero dall'istante t fino a un instante t' e dall'istante t' p sara' vero.

Le state formulas invece riguardano i cammini (infiniti) collegati ad uno stato. La loro sintassi e':

p

a v b

-a

A pi

E pi

dove p e' un atomo proposizionale, a e b sono state formulas, e pi e' una path formula. A e E sono operatori modali. La semantica e' la seguente:

- A pi e' vera in un modello M e in uno stato S0 se e solo se pi e' vera in ogni cammino uscente da S0

- E pi e' vera in un modello M e in uno stato s0 se e solo se esiste un cammino uscente da s0 in cui pi e' vera

Commento:

Domanda 3

Completo

Punteggio ottenuto 4,00 su 4,50

Spiegare il significato delle “commitment rules” del linguaggio Agent-0, facendo riferimento al seguente esempio:

```
(COMMIT
   (?agent REQUEST (DO ?time ?action))

   (AND (B ?now (Friend ?agent))
            (CAN myself ?action)
            (NOT ((CMT ?anyone) (DO ?time ?any_action))))

   (myself (DO ?time ?action)))
```

Le "commitment rules" rappresentano i commitment che gli agenti si impegnano ad eseguire in una situazione. Le regole sono divise in 3 blocchi:

- il primo blocco indica la condizione del messaggio

- il secondo blocco indica la condizione mentale dell'agente

- il terzo blocco indica il commitment dell'agente

In questo esempio abbiamo che l'agente afferma che:

Se un agente ?agent mi richiede di eseguire un'azione ?action al momento ?time (condizione messaggio), allora se io credo che adesso sono amico con ?agent e se io posso eseguire l'azione ?action e non sono impegnato verso nessuno ad eseguire una qualsiasi altra azione al tempo ?time (condizione mentale), allora mi impegno ad eseguire ?action al tempo ?time (commitment).

Commento:

Domanda 4

Completo

Punteggio ottenuto 4,50 su 4,50

Si consideri l'Agent Control Loop 3 seguente:

```
B := B 0 ;
I := I 0 ;
while true do
    get next percept ϱ;
    B := brf (B, ϱ);
    D := options(B, I );
    I := filter (B, D, I );
    π := plan(B, I );
    execute(π)
```

Spiegare quali sono i problemi di tale versione.

Il problema di questa versione e' che l'agente e' overcommitted, sia nei mezzi che nei fini. Il piano viene calcolato ed eseguito completamente, senza far interrompere l'agente per rivalutare le proprie azioni in base ai side-effect delle azioni precedenti o eventuali eventi esterni. Inoltre, l'agente non ha neanche modo di verificare se durante l'esecuzione le sue intenzioni siano ancora adeguate o meno. Per risolvere il problema si puo' per esempio considerare una estensione del ACL v3 inserendo una esecuzione parziale passo passo del piano con una ripianificazione continua e verifica dello stato delle intenzioni.

Commento: