Domanda 9

Completo

Punteggio ottenuto 2,00 su 2,50

Contrassegna domanda

Spiegare la differenza tra coordinazione, cooperazione e negoziazione.

La coordinazione e la cooperazione riguardano la collaborazione tra due sistemi non competitivi, la negoziazione e' un caso di accordo tra due agenti in competizione. Con coordinazione si intende la capacita' di lavorare in un ambiente condiviso, cooperare si intende la collaborazione che hanno due agenti in un ambiente condiviso, la negoziazione permette a degli agenti competitivi di riuscire a lavorare insieme trovando un accordo che lo permetta.

Commento:

Domanda 8

Completo

Punteggio ottenuto 1,66 su 2,50

Contrassegna domanda

Si consideri il seguente protocollo a commitment:

1. cfp means create(C(i, p, propose, accept v reject))
2. accept means none
3. reject means release(C(p, i, accept, done v failure))
4. propose means create(C(p, i, accept, done v failure))
5. refuse means release(C(i, p, propose, accept v reject))
6. done means none
7. failure means none

Scrivere l'insieme di commitment presenti nello stato sociale dopo l'esecuzione delle sequenze di azioni:

1. cfp; propose.
2. propose; accept.
3. cfp; refuse.

1 Lo stato sociale e' {C(i, p, T, accept v reject), C(p, i, accept, done v failure)}

2 Lo stato sociale e' {C(p, i, T, done v failure)}

3 Lo stato sociale e' {C(i, p, propose, accept v reject)}

Commento:

Domanda 7
Completo
Punteggio ottenuto 2,50 su 2,50
Contrassegna domanda

Si considerino l'Agent Control Loop V3, V4 e V5 seguenti. Quali problemi affronta l'Agent Control Loop Version 4 rispetto alla Version 3 e quali problemi affronta l'Agent Control Loop Version 5 rispetto alla Version 4 e come.

```
Agent Control Loop Version 3

B := B0;
I := I0;
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)
end while
```

```
Agent Control Loop Version 4

B := B0;
I := I0;
while true do
    get next percept ρ;
    B := brf(B, ρ);
    D := options(B, I);
    I := filter(B, D, I);
    π := plan(B, I);
    while not empty(π) do
        α := hd(π);
        execute(α);
        π := tail(π);
        get next percept ρ;
        B := brf(B, ρ);
        if not sound(π, I, B) then
            π := plan(B, I)
        end-if
    end-while
end-while
```

```
Agent Control Loop Version 5

B := B0;
I := I0;
while true do
    get next percept ρ;
    B := brf(B, ρ);
    D := options(B, I);
    I := filter(B, D, I);
    π := plan(B, I);
    while not( empty(π)
            or succeeded(I, B)
            or impossible(I, B)) do
        α := hd(π);
        execute(α);
        π := tail(π);
        get next percept ρ;
        B := brf(B, ρ);
        if not sound(π, I, B) then
            π := plan(B, I)
        end-if
    end-while
end-while
```

Nell'ACL v.4 a differenza della v.3 i piani non vengono eseguiti subito tutti ma viene aggiunto un ciclo che ci permette di eseguire e scomporre il piano in azioni (i piano sono gestiti come una lista di azioni), avendo la possibilita' di aggiornare per ogni azione eseguita le credenze in base alla percezione e ponendo anche un controllo effettuato ad ogni azione eseguita, dove si controlla se il piano e' ancora coerente con le credenze e le intenzioni per poter continuarlo ad eseguirlo. Nella v.5 invece a differenza della v. 4 sono stati aggiungi dei controlli al ciclo dell'esecuzione del piano, difatti ora da quel ciclo non si esce piu' solo se le lista dei piani e' vuota ma anche se l'obiettivo e' stato raggiunto prima che possa finire il piano oppure se ci si e' accorti che l'obbiettivo e' impossibile da raggiungere.

Domanda 7
Completo
Punteggio ottenuto 2,50 su 2,50
Contrassegna domanda

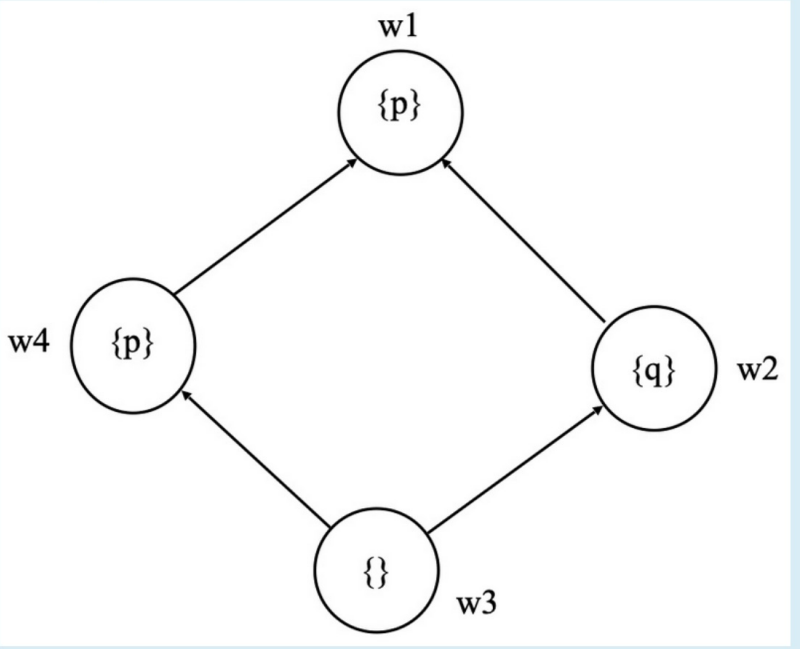
Domanda 6

Parzialmente corretta

Punteggio ottenuto 1,25 su 2,50

Contrassegna domanda

Si consideri il seguente modello di Kripke.



Dire quali delle seguenti affermazioni sono vere.

Scegli una o più alternative:

- [x] a. $[](p \vee q)$ è vero nel mondo w3 ✓
- [ ] b. $[]p$ è vero nel mondo w3
- [ ] c. $([]p \vee []q)$ è vero nel mondo w3
- [ ] d. $[][]p$ è vero nel mondo w3

Risposta parzialmente esatta.

Hai selezionato correttamente 1.

La risposta corretta è: $[](p \vee q)$ è vero nel mondo w3, $[][]p$ è vero nel mondo w3

Domanda 5

Completo

Punteggio ottenuto 2,50 su 2,50

Contrassegna domanda

Nella logica epistemica si assume che per l'operatore **K** (conoscenza) valgano le seguenti proprietà:

1. La conoscenza è non contraddittoria
2. Ciò che si conosce è vero
3. Vale la proprietà della introspezione positiva
4. Vale la proprietà dell'introspezione negativa

Fornire i quattro assiomi che rappresentano queste proprietà.

1. $K\varphi \rightarrow \neg K\neg\varphi$ $\quad B\varphi \rightarrow \neg B\neg\varphi$ Assioma di serialita'
2. $K\varphi \rightarrow \varphi$ Assioma di riflessivita'
3. $K\varphi \rightarrow KK\varphi$ $\quad B\varphi \rightarrow BB\varphi$ Assioma di transitivita'
4. $\neg K\neg\varphi \rightarrow K(\neg K\neg\varphi)$ $\quad \neg B\neg\varphi \rightarrow B(\neg B\neg\varphi)$ Assioma euclideoN

Commento:

Domanda 4

Completo

Punteggio ottenuto 1,00 su 2,50

Contrassegna domanda

Cosa si intende per strategia pura e strategia mista nella teoria dei giochi?

Per strategia mista si intende un tipo di strategia legata alle probabilita', quindi la scelta per il giocatore e' guidata anche dalle probabilita'.

Nella strategia pura il giocatore dovra' scegliere in base alle proprie informazioni la strategia migliore, come ad esempio puo' essere una strategia dominante.

Commento:

Domanda 3

Completo

Punteggio ottenuto 2,00 su 2,50

Contrassegna domanda

Spiegare che cosa si intende per **strategia dominante** nella teoria dei giochi. Si consideri la seguente **matrice di payoff**, dove sia Alice che Bob possono eseguire o l'azione a1 o a2. Spiegare se Alice ha una strategia dominante.

	Alice: a1	Alice: a2
Bob: a1	A = 0, B = 0	A = 1, B = 3
Bob: a2	A = 3, B = 1	A = 2, B = 2

Per strategia dominante si intende la strategia migliore che un giocatore puo' adottare a prescindere da quello che scelga l'altro giocatore. Ad esempio la strategia dominante per Alice viene scelta in base alla matrice di payoff senza contare cosa possa scegliere Bob (non lo si puo' sapere a priori). In questo specifico esempio ne Bob ne Alice hanno una strategia dominante.

Commento:

Domanda 2

Completo

Punteggio ottenuto 2,30 su 2,50

Contrassegna domanda

Spiegare sinteticamente la motivazione e la struttura della subsumption architecture proposta da Brooks.

In questa architettura proposta da Brooks l'agente ha un organizzazione a livelli e il suo comportamento e' determinato tra l'interazione tra qesti livelli. Ogni livello ha un suo compito ed un suo comportamento. I livelli piu' bassi eseguono le azioni piu' semplici e piu' rapide e sono quelli che costituiscono le azioni piu' reattive e immediate dell'agente. In generalu tutti i livelli sono strutturati come regole. Questi livelli sono definiti come automi a stati finiti che possono passare da uno stato all'altro continuando ad interagire fra loro. Il nome e' dovuto al fatto che ogni livello sussume i livelli sottostanti aggiungendo nuove capacita'.

Commento:

Domanda 12

Parzialmente corretta

Punteggio ottenuto 1,17 su 2,50

Contrassegna domanda

Si consideri il seguente frammento di codice in AgentSpeak(l):

```
a(1).
a(2).
a(3).
a(4).
a(5).

+!start1 : a(X) & .print("Start1 X: ", X) & X=3 <- .print("Start1 (body) X: ", X) .

+!start2 : true <- ?a(Y); .print("Start2 (body) Y: ", Y) .

+!start3 : a(X) & .print("Start3 X: ", X) & X=3 <- ?a(Y); .print("Start3 (body) Y: ", Y); Y=3 .
```

Quali delle seguenti affermazioni sono vere?

Scegli una o più alternative:

- [ ] a.
  L'evento +!start1 ha come output:
  [test] Start1 X: 1
  [test] Start1 X: 2
  [test] Start1 X: 3
  [test] Start1 (body) X: 1
  [test] Start1 (body) X: 2

  [test] Start1 (body) X: 3
- [x] b. L'evento +!start1 ha come output:
  [test] Start1 X: 1
  [test] Start1 X: 2
  [test] Start1 X: 3
  [test] Start1 (body) X: 3 ✓
- [x] c. L'evento +!start3 ha come output:
  [test] Start3 X: 1
  [test] Start3 X: 2
  [test] Start3 X: 3
  [test] Start3 (body) Y: 1
  [test] No failure ✓
- [ ] d.
  L'evento +!start2 ha come output:
  [test] Start2 (body) Y: 1
- [x] e.
  L'evento +!start2 ha come output:
  [test] Start2 (body) Y: 1

  [test] Start2 (body) Y: 2

  [test] Start2 (body) Y: 3 ✗
- [ ] f. L'evento +!start3 ha come output:
  [test] Start3 X: 3
  [test] Start3 (body) Y: 3
- [ ] g. L'evento +!start3 ha come output:
  [test] Start3 X: 1
  [test] Start3 X: 2
  [test] Start3 X: 3
  [test] Start3 (body) Y: 1
  [test] Start3 (body) Y: 2

  [test] Start3 (body) Y: 3
- [ ] h. L'evento +!start2 ha come output:
  [test] Start2 (body) Y: 3

Risposta parzialmente esatta.

Hai selezionato correttamente 2.
La risposta corretta è: L'evento +!start1 ha come output:
[test] Start1 X: 1
[test] Start1 X: 2
[test] Start1 X: 3
[test] Start1 (body) X: 3,
L'evento +!start2 ha come output:
[test] Start2 (body) Y: 1, L'evento +!start3 ha come output:
[test] Start3 X: 1
[test] Start3 X: 2
[test] Start3 X: 3
[test] Start3 (body) Y: 1
[test] No failure

Domanda 11

Completo

Punteggio ottenuto 2,50 su 2,50

Contrassegna domanda

Spiegare la differenza tra **practical reasoning** e **theoretical reasoning**.

Il pratical reasoning e' un ragionamento guidato dalle azioni, mentre il theoretical reasoning e' un ragionamento guidato dalla credenze.

Commento:

Domanda 10

Completo

Punteggio ottenuto 2,50 su 2,50

Contrassegna domanda

Si consideri il seguente programma Jade:

```
import jade.core.Agent;
import jade.core.behaviours.Behaviour;

public class Esame20190618 extends Agent {
   private int counter = 0;
   protected void setup() {
      System.out.println("Adding first behaviour");
      addBehaviour(new Behaviour() {
         public void action() {
            if ((counter % 2) == 0)
               System.out.println(counter);
         }
         public boolean done() {
            return counter >= 20;
         }
      } );
      System.out.println("Adding second behaviour");
      addBehaviour(new Behaviour() {
         public void action() {
            if ((counter % 2) == 1)
               System.out.println(counter);
         }
         public boolean done() {
            return counter >= 20;
         }
      } );
      System.out.println("Adding third behaviour");
      addBehaviour(new Behaviour() {
         public void action() {
            counter++;
         }
         public boolean done() {
            return counter >= 20;
         }
      } );

   }
   protected void takeDown() {
      System.out.println("Bye...");
   }
}
```

Dire cosa viene stampato a console durante l'esecuzione.

```
Adding first behaviour
Adding second behaviour
Adding third behaviour
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
```

```
Adding first behaviour
Adding second behaviour
Adding third behaviour
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
```

Domanda 1

Completo

Punteggio ottenuto 2,00 su 2,50

Contrassegna domanda

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)))
```

In questo esempio, un agente (?agent) richiede se al tempo (?time) puo' essere eseguita l'azione (?action), B riceve la richiesta. Le condizioni di messaggio e le condizioni mentali per soddisfare questa richiesta sono le seguenti:

B controlla se ora (?now) e' amico con l'agente (?agent), se puo' eseguire l'azione (?action), se non ha nessun commit e se non sta facendo nessun'altra azione (?any_action) allo stesso tempo (?time), allora puo' eseguire l'azione (?action) al tempo (?time).

Commento:

Domanda 11

Risposta non data

Punteggio max.: 3,00

Contrassegna domanda

Si consideri il seguente frammento di codice Jason:

```
a(1).
a(2).
b(2).
h(X) : a(X) & b(Y) & X = Y <- body1. (p1)
h(3) : a(X) & b(Y) & X = Y <- body2, (p2)
h(X) : a(X) & b(Y) & X = Y <- ?b(1), body3. (p3)
```

Quali sono le affermazioni vere?

Scegli una o più alternative:

- a. (p1) è il solo piano rilevante per l'evento +!h(2).
- b. (p1) e (p3) sono piani rilevanti per l'evento +!h(2).
- c. (p1) è il solo piano applicabile per l'evento +!h(2).
- d. (p1) e (p3) sono piani rilevanti per l'evento +!h(1).
- e. Nessun piano rilevante per l'evento +!h(3).
- f. Nessun piano applicabile per l'evento +!h(3).
- g. (p2) è il solo piano rilevante per l'evento +!h(3).
- h. (p1) e (p2) sono piani rilevanti per l'evento +!h(2).
- i. (p1) e (p3) sono piani applicabili per l'evento +!h(2).
- j. (p1), (p2) e (p3) sono piani rilevanti per l'evento +!h(1).
- k. Non esiste un piano applicabile per l'evento +!h(1).
- l. (p1) è un piano rilevante per l'evento +!h(1).
- m. (p2) è il solo applicabile per l'evento +!h(3).
- n. (p1) è un piano applicabile per l'evento +!h(1).

Risposta errata.

La risposta corretta è: (p1) e (p3) sono piani rilevanti per l'evento +!h(1)., Non esiste un piano applicabile per l'evento +!h(1)., (p1) e (p3) sono piani rilevanti per l'evento +!h(2)., (p1) e (p3) sono piani applicabili per l'evento +!h(2)., (p2) è il solo piano rilevante per l'evento +!h(3)., (p2) è il solo applicabile per l'evento +!h(3).

Domanda 10

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

Contrassegna domanda

Si consideri il seguente frammento di codice Jade:

```
public class MyThreeStepBehaviour extends Behaviour {
   private int step = 0;
   public void action() {
      // istruzione1
      switch (step) {
      case 0: // perform operation X
         step++;
         break;
      case 1: // perform operation Y
         step++;
         break;
      case 2: // perform operation Z
         step++;
         break;
      }
      // istruzione2
   }
   public boolean done() {
      return step == 3;
   }
}
```

Quali delle seguenti affermazioni sono vere?

Scegli una o più alternative:

- [ ] a. istruizione1 viene eseguita 1 volta.
- [x] b. istruzione1 e istruzione2 vengono eseguite 3 volte. ✓
- [ ] c. istruzione2 non viene mai eseguita.
- [ ] d. istruizone1 e istruzione2 vengono eseguite 1 volta.
- [ ] e. istruzione1 non viene mai eseguita.

Risposta corretta.

La risposta corretta è: istruzione1 e istruzione2 vengono eseguite 3 volte.

Domanda 9

Parzialmente corretta

Punteggio ottenuto 0,33 su 2,00

Contrassegna domanda

Si consideri lo scheduling dei comportamenti di un agente Jade. Quali delle seguenti affermazioni sono vere?

Scegli una o più alternative:

- [x] a. La funzione di setup() è invocata una sola volta. ✓
- [ ] b. Il metodo action() è eseguito completamente ogni volta che il comportamento è selezionato dal pool dei comportamenti attivi dell'agente.
- [ ] c. Un comportamento ha tanti metodi action() quanti sono i task indipendenti che tale comportamento deve raggiungere.
- [ ] d. Il metodo action() è eseguito parzialmente sino ad una condizione di sincronizzazione con altro comportamento presente pool dei comportamenti attivi dell'agente.
- [x] e. Un comportamento ciclico non valuta mai il risultato dell'esecuzione del metodo done(). ✗
- [x] f. Un comportamento risiede nel pool dei comportamenti attivi fino all'esecuzione del metodo takeDown(). ✗
- [x] g. Il valore restituito dal metodo done() determina se un comportamento deve o no essere inserito nel pool dei comportamenti attivi dell'agente. ✓

Risposta parzialmente esatta.

Hai selezionato troppe opzioni.
La risposta corretta è: La funzione di setup() è invocata una sola volta., Il metodo action() è eseguito completamente ogni volta che il comportamento è selezionato dal pool dei comportamenti attivi dell'agente., Il valore restituito dal metodo done() determina se un comportamento deve o no essere inserito nel pool dei comportamenti attivi dell'agente.

Domanda 8

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

Contrassegna domanda

Riordinare le seguenti istruzioni di un Agent Control Loop.

1. B := B0; I := I0; ✓
2. while true do ✓
3. get next percept ✓
4. B := brf(B, ρ) ✓
5. D := options(B, I) ✓
6. I := filter(B, D, I) ✓
7. π := plan(B, I) ✓
8. execute(π) ✓
9. end while ✓

Risposta corretta.

La risposta corretta è: 1. – B := B0; I := I0;, 2. – while true do, 3. – get next percept, 4. – B := brf(B, ρ), 5. – D := options(B, I), 6. – I := filter(B, D, I), 7. – π := plan(B, I), 8. – execute(π), 9. – end while

Domanda 6

Parzialmente corretta

Punteggio ottenuto 2,00 su 3,00

Contrassegna domanda

Per modellare con la logica modale diverse proprietà degli agenti può essere necessario modificare la logica aggiungendo degli **assiomi** o, in alternativa, restringere la classe dei modelli formulando delle **proprietà** dei frame. In diversi casi la formulazione con assiomi **corrisponde** alle proprietà dei frame.

Dati i seguenti assiomi e proprietà:

(a1) $[\,]\varphi \Rightarrow \varphi$

(a2) $[\,]\varphi \Rightarrow \Diamond\varphi$

(p1) $\forall w \in W \;.\; \exists w' \in W \;.\; (w,w') \in R$

(p2) $\forall w \in W \;.\; (w,w) \in R$

dire quali delle seguenti affermazioni sono vere.

Scegli una o più alternative:

- [ ] a.
  L'assioma (a2) corrisponde alla proprietà (p2).
- [x] b.
  L'assioma (a1) corrisponde alla proprietà (p2). ✓
- [ ] c.
  L'assioma (a1) corrisponde alla proprietà (p1).
- [ ] d.
  La proprietà (p2) modella la serialità.
- [x] e.
  L'assioma (a2) corrisponde alla proprietà (p1). ✓
- [ ] f.
  La proprietà (p2) modella la riflessività.

Risposta parzialmente esatta.

Hai selezionato correttamente 2.
La risposta corretta è:
La proprietà (p2) modella la riflessività.,

L'assioma (a1) corrisponde alla proprietà (p2).,

L'assioma (a2) corrisponde alla proprietà (p1).

Domanda 5

Parzialmente corretta

Punteggio ottenuto 2,00 su 4,00

Contrassegna domanda

La seguente matrice di payoff si riferisce al "game of chicken" in cui i due giocatori **I** e **J** guidano ciascuno la propria auto verso un precipizio. Ogni giocatore può sterzare o andare diritto.

	I : diritto	I : sterza
J : diritto	I = 0, J = 0	I = 1, J = 3
J : sterza	I = 3, J = 1	I = 2, J = 2

Dire quali delle seguenti affermazioni sono vere.

Scegli una o più alternative:

- a. [I : diritto, J : diritto] è un *equilibrio di Nash*.
- b. [I : diritto, J : sterza] è un *equilibrio di Nash*.
- c. "sterza" è una *strategia dominante* per I.
- d. [I : sterza, J : sterza] è *Pareto optimal*. ✓

Risposta parzialmente esatta.

Hai selezionato correttamente 1.

La risposta corretta è: [I : diritto, J : sterza] è un *equilibrio di Nash.*, [I : sterza, J : sterza] è *Pareto optimal*.

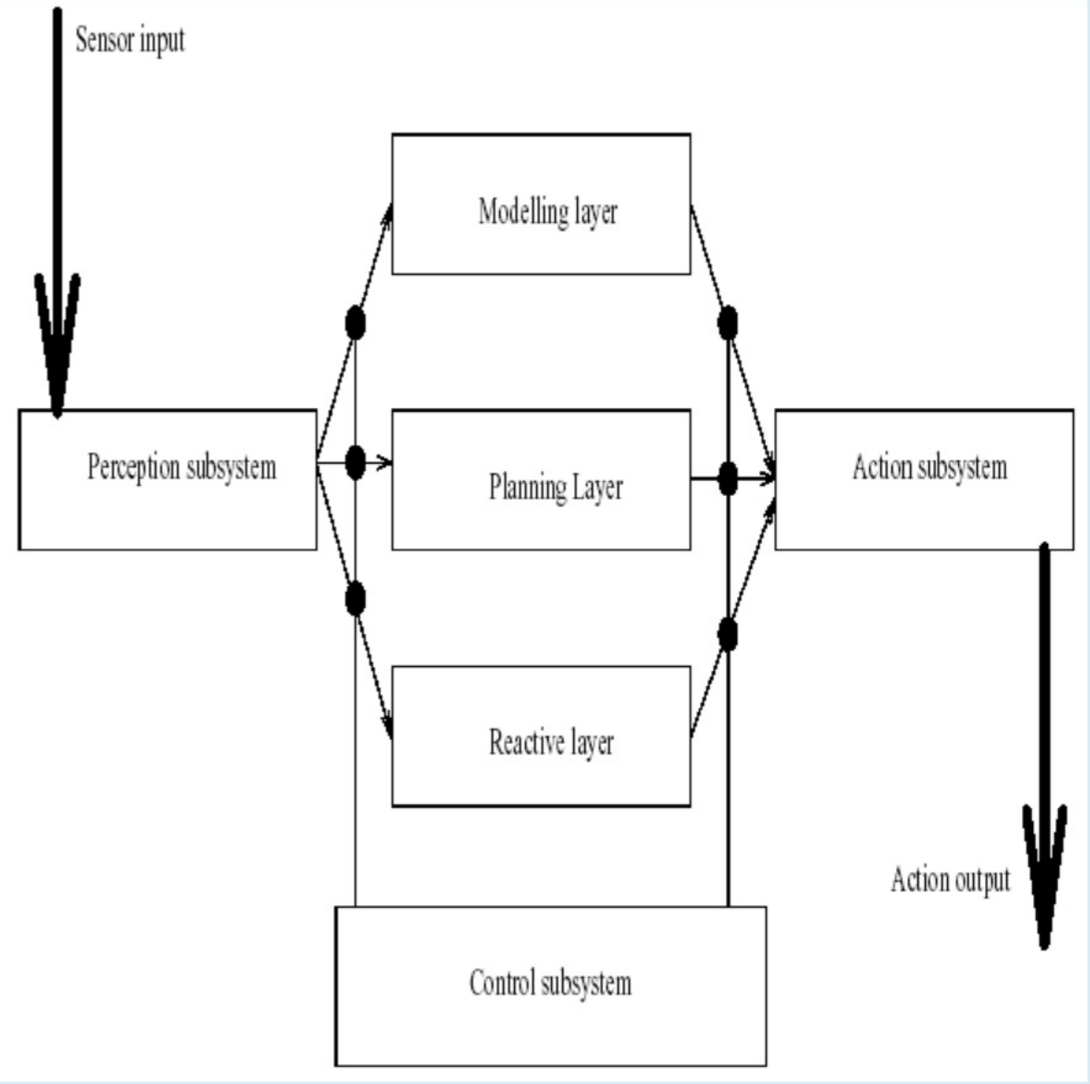
Domanda 4

Completo

Punteggio ottenuto 1,50 su 2,00

Contrassegna domanda

Descrivere i componenti della architettura **TouringMachines** rappresentata nella figura seguente. In particolare, dire se si tratta di una architettura orizzontale o verticale, e spiegare il significato del "control subsystem".



La Touring Machine e' una architettura a stratificazione orizzontale consistente di una struttura con sottosistemi di percezione e di azione che si interfacciano direttamente con l'ambiente e con tre livelli di controllo racchiusi in un framework di controllo che fa da mediatore tra i livelli.

Il reactive layer e' implementato come un set di regole di situazioni-azioni.

Il planning layer costruisce piani e seleziona azioni da eseguire in ordine per completare i goal dell'agente.

Il modelling layer contiene una rappresenazione simbolica del "cognitive state" delle altre entita' dell'ambiente dell'agente.

Tutti e tre i livelli comunicano tra loro e sono racchiusi in un framework di controllo (Control subsystem) che usa regole di controllo.

Domanda 3

Completo

Punteggio ottenuto 2,00 su 2,00

Contrassegna domanda

Commentare la formula che definisce un **persistent goal** secondo Cohen e Levesque.

$$
\begin{aligned}
(\text{P-GOAL } i\ \varphi) \equiv\ & (\text{GOAL } i\ (\text{LATER } \varphi)) \wedge (\text{BEL } i\ \neg\varphi) \wedge \\
& [\text{BEFORE } ((\text{BEL } i\ \varphi) \vee (\text{BEL } i\ \mathbf{G}\ \neg\varphi)) \\
& \neg(\text{GOAL } i\ (\text{LATER } \varphi))]
\end{aligned}
$$

un Goal persistente e' un achievement goal che puo' essere rimosso solo per due motivi: o l'agente lo ha raggiunto, oppure si e' reso conto che non e' mai raggiungibile.

Piu' formalmente possiamo dire: Ha un goal teta che possibilmente diventera' vero e crede che teta non e' attualmente vera.

Prima che il goal vada in "drop" una delle seguenti condizioni deve essere vera:

1) l'agente crede che il goal sia soddisfatto.

2) l'agente crede che il goal non sara' mai soddisfatto.

Domanda 14

Risposta corretta

Punteggio ottenuto 1,00 su 1,00

Contrassegna domanda

L'obiettivo del Knowledge Sharing Effort è di sviluppare tecniche, metodologie e strumenti software per la condivisione e il riuso di conoscenza.

Scegli una risposta:

- Vero ✓
- Falso

La risposta corretta è 'Vero'.

Domanda 12

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

Contrassegna domanda

Si consideri la definizione di practical reasoning, quali delle seguenti affermazioni sono vere?

Scegli una o più alternative:

- [x] a. La fase di means-ends reasoning consiste nel decidere come uno stato di interesse deve essere raggiunto. ✓
- [x] b. La fase di deliberazione consiste nel decidere quale stato di interesse deve essere raggiunto. ✓
- [ ] c. La fase di deliberazione consiste nel decidere come uno stato di interesse deve essere raggiunto.
- [x] d. La fase di deliberazione precede la fase di means-ends reasoning. ✓
- [ ] e. La fase di means-ends reasoning consiste nel decidere quale stato di interesse deve essere raggiunto.

Risposta corretta.

La risposta corretta è: La fase di deliberazione consiste nel decidere quale stato di interesse deve essere raggiunto., La fase di means-ends reasoning consiste nel decidere come uno stato di interesse deve essere raggiunto., La fase di deliberazione precede la fase di means-ends reasoning.

Domanda 13

Parzialmente corretta

Punteggio ottenuto 1,33 su 2,00

Contrassegna domanda

Si consideri il seguente protocollo a commitment:

cfp means create(C(i, p, propose, accept v reject))

accept means none

reject means release(C(p, i, accept, done v failure))

propose means create(C(p, i, accept, done v failure))

refuse means release(C(i, p, propose, accept v reject))

done means none

failure means none

Quali delle seguenti affermazioni sono vere?

Scegli una o più alternative:

- [ ] a. Lo stato sociale vuoto è raggiunto dopo l'esecuzione della sequenza cfp e reject.
- [ ] b. Non è possibile che si verifichi la sequenza cfp e accept.
- [x] c. Lo stato sociale è {C(i, p, T, accept v reject), C(p, i, accept, done v failure)} dopo l'esecuzione della sequenza cfp e propose. ✓
- [ ] d. Lo stato sociale è {C(i, p, T, accept v reject), C(p, i, accept, done v failure)} dopo l'esecuzione dell'azione propose.
- [x] e. Lo stato sociale vuoto è raggiunto dopo l'esecuzione della sequenza cfp, propose, accept, done. ✓

Risposta parzialmente esatta.

Hai selezionato correttamente 2.

La risposta corretta è: Lo stato sociale è {C(i, p, T, accept v reject), C(p, i, accept, done v failure)} dopo l'esecuzione della sequenza cfp e propose., Lo stato sociale vuoto è raggiunto dopo l'esecuzione della sequenza cfp e reject., Lo stato sociale vuoto è raggiunto dopo l'esecuzione della sequenza cfp, propose, accept, done.

Domanda 1

Completo

Punteggio ottenuto 0,00 su 3,00

Contrassegna domanda

Il GOLOG è un linguaggio di programmazione che fa uso del situation calculus implementato con la logica classica. Spiegare il significato dei "Successor state axioms" facendo riferimento al seguente esempio:

$$Poss(a,s) \Rightarrow [broken(x,do(a,s)) \equiv \exists r\{a = drop(r,x) \wedge fragile(x,s)\} \vee broken(x,s) \wedge \neg\exists r\{a = repair(r,x)\}]$$

le azioni in GOLOG, presentato nel 1994, sono specificate dando delle precondizioni e degli effetti.

I Successor state axioms vengono usati per esplicitare gli effetti delle azioni.

Commento:

Domanda 2

Completo

Punteggio ottenuto 1,50 su 2,00

Contrassegna domanda

Nella logica temporale Branching-Time il tempo è modellato con un albero infinito su cui si possono formulare *path formulas* o *state formulas*. Dire se le formule **A**π e **E**π sono *path* o *state formulas* e spiegare il loro significato.

Le state formula sono le seguenti:
1)p
2) alfa OR Beta
3) not alfa
4) A(pi)
5)E(pi)
Dove alfa e' una state formula mentre pi si riferisce alle path formula.

Le path formula sono invece:
1) alfa
2)pi OR p
3) not pi
4) Xpi
5)pi U p
Dove alfa e' uno state formula mentre pi e p sono path formula

A(pi) significa "per tutti i futuri", ovvero viene usato per controllare tutti i percorsi (partendo da uno stato iniziale)
E(pi) significa "per qualche futuro", ovvero viene usato per controllare solo alcuni percorsi (partendo da uno stato iniziale).
A(pi) e E(pi) sono duali.

Commento:

Si consideri il modello di esecuzione dei comportamenti di un agente in JADE. Dire quali delle seguenti affermazioni sono vere.

Scegli una o più alternative:

- [ ] a. La gestione dei compotamenti di un agente è di tipo competitivo.
- [ ] b. Quando un comportamento di un agente è selezionato, il suo metodo action viene completamente eseguito fino al suo return.
- [x] c. Ogni comportamento di un agente definisce un metodo done.
- [x] d. La sincronizzazione tra i comportamenti di un agente è ottenuta mediante le usuali primitive del linguaggio Java.
- [ ] e. La gestione dei comportamenti di un agente è di tipo pre-emptive.

Cohen e Levesque hanno proposto una logica per modellare agenti BDI la cui semantica è basata sui mondi possibili. Spiegare cosa hanno di diverso i modelli di questa logica rispetto ai modelli della logica modale classica.

Si consideri il seguente esempio di programma Concurrent METATEM.

Example. A **'resource producer'** rp:

rp (ask1, ask2) [give1, give2]

1. $\bigcirc$ask1 $\Rightarrow$ $\Diamond$give1
2. $\bigcirc$ask2 $\Rightarrow$ $\Diamond$give2
3. **start** $\Rightarrow$ $\Box\neg$(give1 $\wedge$ give2)

Two **consumers**:

rc1 (give1) [ask1]

4. **start** $\Rightarrow$ ask1
5. $\bigcirc$ask1 $\Rightarrow$ ask1

rc2 (ask1, give2) [ask2]

6. $\bigcirc$(ask1 $\wedge$ $\neg$ask2) $\Rightarrow$ ask2

Commentare brevemente le regole da 1 a 6.

Quali delle seguente affermazioni relative ad AgentSpeak(L) sono vere?

Scegli una o più alternative:

- a. AgentSpeak(L) rappresenta achievement goal in un piano ma non test goal.
- b. AgentSpeak(L) ha una funzione di selezione per determinare i piani rilevanti.
- c. AgentSpeak(L) distingue tra eventi interni ed esterni.
- d. AgentSpeak(L) ha una funzione di selezione per determinare gli eventi rilevanti.
- e. AgentSpeak(L) è implentato in linguaggio Java.
- f. In AgentSpeak(L) sono rappresentati i belief e i desire ma non le intention.
- g. AgentSpeak(L) è un linguaggio dichiarativo.
- h. AgentSpeak(L) ha una funzione di selezione per determinare i belief da adottare.

Si consideri l'attività di *Practical Reasoning*, quali delle seguenti affermazioni sono vere?

Scegli una o più alternative:

- a. L'attività di means-ends reasoning produce come risultato le intezioni dell'agente.
- b. L'attività di deliberation stabilisce come raggiungere uno stato.
- c. L'attività di means-ends reasoning stabilisce quale stato raggiungere.

Domanda **14**

Risposta non data

Punteggio max.: 2,50

Contrassegna domanda

Cosa si intende per **strategia pura** e **strategia mista** nella teoria dei giochi.

Domanda **12**

Parzialmente corretta

Punteggio ottenuto 1,20 su 1,50

Contrassegna domanda

Si consideri l'attività di *Practical Reasoning*, quali delle seguenti affermazioni sono vere?

Scegli una o più alternative:

- a. L'attività di means-ends reasoning stabilisce quale stato raggiungere.
- b. L'attività di means-ends reasoning produce come risultato le intezioni dell'agente.
- c. L'attività di deliberation stabilisce quale stato raggiungere. ✓
- d. L'attività di deliberation stabilisce come raggiungere uno stato.
- e. L'attività di means-ends reasoning verifica l'effettiva eseguibilità di un piano per soddisfare le intenzioni. ✗
- f. L'attività di deliberation produce come risultato le intenzioni dell'agente. ✓
- g. L'attività di means-ends reasoning stabilisce come raggiungere uno stato. ✓
- h. L'attività di deliberation delibera la sequenza di azioni che l'agente deve eseguire per raggiungere uno stato.

Risposta parzialmente esatta.

Hai selezionato troppe opzioni.
La risposta corretta è: L'attività di deliberation stabilisce quale stato raggiungere., L'attività di deliberation produce come risultato le intenzioni dell'agente., L'attività di means-ends reasoning stabilisce come raggiungere uno stato.

Domanda **13**

Risposta errata

Punteggio ottenuto 0,00 su 2,00

Contrassegna domanda

Quali delle seguente affermazioni relative ad AgentSpeak(L) sono vere?

Scegli una o più alternative:

- a. AgentSpeak(L) è implentato in linguaggio Java.
- b. AgentSpeak(L) rappresenta achievement goal in un piano ma non test goal.
- c. AgentSpeak(L) ha una funzione di selezione per determinare gli eventi rilevanti. ✗
- d. In AgentSpeak(L) sono rappresentati i belief e i desire ma non le intention. ✗
- e. AgentSpeak(L) ha una funzione di selezione per determinare i piani rilevanti.
- f. AgentSpeak(L) è un linguaggio dichiarativo. ✓
- g. AgentSpeak(L) distingue tra eventi interni ed esterni.
- h. AgentSpeak(L) ha una funzione di selezione per determinare i belief da adottare. ✗

Risposta errata.

La risposta corretta è: AgentSpeak(L) è un linguaggio dichiarativo., AgentSpeak(L) distingue tra eventi interni ed esterni.

Domanda **11**

Risposta non data

Punteggio max.: 2,50

Contrassegna domanda

Si consideri il seguente esempio di programma Concurrent METATEM.

Example. A '**resource producer**' rp:

rp (ask1, ask2) [give1, give2]

1. $\bigcirc$ask1 $\Rightarrow$ $\Diamond$give1
2. $\bigcirc$ask2 $\Rightarrow$ $\Diamond$give2
3. **start** $\Rightarrow$ $\Box\neg$(give1 $\wedge$ give2)

Two **consumers**:

rc1 (give1) [ask1]

4. **start** $\Rightarrow$ ask1
5. $\bigcirc$ask1 $\Rightarrow$ ask1

rc2 (ask1, give2) [ask2]

6. $\bigcirc$(ask1 $\wedge$ $\neg$ask2) $\Rightarrow$ ask2

Commentare brevemente le regole da 1 a 6.

e. L'evento +!start3 ha come output:
[test] Start3 X: 3
[test] Start3 (body) Y: 3 ✗

f. L'evento +!start2 ha come output:
[test] Start2 (body) Y: 3

g. L'evento +!start1 ha come output:
[test] Start1 X: 1
[test] Start1 X: 2
[test] Start1 X: 3
[test] Start1 (body) X: 3 ✓

h.
L'evento +!start1 ha come output:
[test] Start1 X: 1
[test] Start1 X: 2
[test] Start1 X: 3
[test] Start1 (body) X: 1
[test] Start1 (body) X: 2

[test] Start1 (body) X: 3

Risposta parzialmente esatta.

Hai selezionato correttamente 2.
La risposta corretta è: L'evento +!start1 ha come output:
[test] Start1 X: 1
[test] Start1 X: 2
[test] Start1 X: 3
[test] Start1 (body) X: 3,
L'evento +!start2 ha come output:
[test] Start2 (body) Y: 1, L'evento +!start3 ha come output:
[test] Start3 X: 1
[test] Start3 X: 2
[test] Start3 X: 3
[test] Start3 (body) Y: 1
[test] No failure

Domanda **10**

Parzialmente corretta

Punteggio ottenuto 1,40 su 3,00

Contrassegna domanda

Si consideri il seguente frammento di codice in AgentSpeak(l):

```
a(1).
a(2).
a(3).
a(4).
a(5).

+!start1 : a(X) & .print("Start1 X: ", X) & X=3 <- .print("Start1 (body) X: ", X) .

+!start2 : true <- ?a(Y); .print("Start2 (body) Y: ", Y) .

+!start3 : a(X) & .print("Start3 X: ", X) & X=3 <- ?a(Y); .print("Start3 (body) Y: ", Y); Y=3 .
```

Quali delle seguenti affermazioni sono vere?

Scegli una o più alternative:

- [ ] a.
  L'evento +!start2 ha come output:
  [test] Start2 (body) Y: 1

  [test] Start2 (body) Y: 2

  [test] Start2 (body) Y: 3
- [x] b.
  L'evento +!start2 ha come output:
  [test] Start2 (body) Y: 1 ✓
- [ ] c. L'evento +!start3 ha come output:
  [test] Start3 X: 1
  [test] Start3 X: 2
  [test] Start3 X: 3
  [test] Start3 (body) Y: 1
  [test] No failure
- [ ] d. L'evento +!start3 ha come output:
  [test] Start3 X: 1
  [test] Start3 X: 2
  [test] Start3 X: 3
  [test] Start3 (body) Y: 1
  [test] Start3 (body) Y: 2

  [test] Start3 (body) Y: 3
- [x] e. L'evento +!start3 ha come output:
  [test] Start3 X: 3
  [test] Start3 (body) Y: 3 ✗

Domanda **8**

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

Contrassegna domanda

Due giocatori A e B partecipano a una asta di Wickrey. A e B fanno una offerta maggiore di tutti gli altri partecipanti. A offre 100, mentre B offre 80. Quali delle seguenti affermazioni sono vere:

Scegli una o più alternative:

- a. B è il vincitore e paga 100.
- b. B è il vincitore e paga 80.
- c. A è il vincitore e paga 80. ✓
- d. A è il vincitore e paga 100.
- e. Una asta di Wickrey è una asta in busta chiusa. ✓

Risposta corretta.

La risposta corretta è: Una asta di Wickrey è una asta in busta chiusa., A è il vincitore e paga 80.

Domanda **9**

Completo

Punteggio ottenuto 2,00 su 3,00

Contrassegna domanda

Data la seguente matrice di payoff, spiegare in cosa consiste il *dilemma del prigioniero*.

	Alice: testify	Alice: refuse
Bob: testify	A = -5, B = -5	A = -10, B = 0
Bob: refuse	A = 0, B = -10	A = -1, B = -1

Il dilemma del prigioniero e' uno degli esempio piu' popolari della Teoria dei giochi.
In questo caso Alice e Bob sono due prigionieri arrestati dalla polizia, la quale interrogandoli separatamente propone loro un patteggiamento. Come visibile dalla matrice di payoff le condizioni sono queste:
Se entrambi non confessassero riceverebbero 1 anno di reclusione a testa
Se entrambi confessassero riceverebbero 5 anni di reclusione a testa
Se uno dei due confessasse e l'altro no, il testimone sarebbe totalmente libero mentre l'altro prigioniero sconterebbe una pena di 10 anni.
Dato che i due prigionieri tendono a preservarsi egoisticamente il punto di equilibrio lo si individua nella situazione in cui sia Bob che Alice confessino per evitare di essere traditi dal proprio complice.

Commento:

Domanda **5**

Completo

Punteggio ottenuto 1,00 su 2,00

Contrassegna domanda

Cohen e Levesque hanno proposto una logica per modellare agenti BDI la cui semantica è basata sui mondi possibili. Spiegare cosa hanno di diverso i modelli di questa logica rispetto ai modelli della logica modale classica.

La logica di Cohen-Levesque e' una delle due logiche utili per modellare gli agenti BDI. Essa si bassa sulla logica modale e sulla logica temporale lineare.
A differenza della logica classica permette di avere agromentazioni non funzionali e considera degli operatori come il BEL, INTENT, COMMITMENT

Commento:

Domanda **6**

Completo

Punteggio ottenuto 0,00 su 2,50

Contrassegna domanda

Logica modale: definire l'**assioma K** e la regola di **necessitation**.

assioma K: $K(\phi \rightarrow \psi) = K\phi \rightarrow K\psi$

regola necessitation: $[]\ K\phi \rightarrow KK\phi$

Commento:

Domanda **7**

Risposta errata

Punteggio ottenuto 0,00 su 1,50

Contrassegna domanda

Con il termine performativa, nella Speech Act Theory si intende:

Scegli un'alternativa:

- a. L'azione risultante dall'atto fisico di chi parla. ✗
- b. L'atto fisico di chi parla.
- c. Il significato attribuito da chi parla.

Risposta errata.

La risposta corretta è: Il significato attribuito da chi parla.

Domanda **3**

Risposta errata

Punteggio ottenuto 0,00 su 1,50

Contrassegna domanda

Si consideri il commitment sociale C(x, y, p, q) secondo la definizione data da Munindar Singh. Quali dele seguenti affermazioni sono vere?

Scegli una o più alternative:

- a. Se q diventa falsa, il commitment è violato.
- b. y ha la aspettativa che q diventi vera nel caso che p sia vera.
- c. Se p diventa falsa, il commitment è violato.
- d. x ha la aspettativa che q diventi vera nel caso che p sia vera. ✗
- e. x è il debitore del commitment.
- f. Se p è vera e successivamente q diventa falsa, il commitment è violato. ✓
- g. y è il debitore del commitment. ✗
- h. Se p diventa vera, il commitment è soddisfatto.

Risposta errata.

La risposta corretta è: x è il debitore del commitment., y ha la aspettativa che q diventi vera nel caso che p sia vera., Se p è vera e successivamente q diventa falsa, il commitment è violato.

Domanda **4**

Parzialmente corretta

Punteggio ottenuto 1,67 su 2,50

Contrassegna domanda

Si consideri il modello di esecuzione dei comportamenti di un agente in JADE. Dire quali delle seguenti affermazioni sono vere.

Scegli una o più alternative:

- a. La gestione dei compotamenti di un agente è di tipo competitivo.
- b. La gestione dei comportamenti di un agente è di tipo pre-emptive.
- c. Quando un comportamento di un agente è selezionato, il suo metodo action viene completamente eseguito fino al suo return. ✓
- d. Ogni comportamento di un agente definisce un metodo done. ✓
- e. La sincronizzazione tra i comportamenti di un agente è ottenuta mediante le usuali primitive del linguaggio Java. ✗

Risposta parzialmente esatta.

Hai selezionato troppe opzioni.
La risposta corretta è: Quando un comportamento di un agente è selezionato, il suo metodo action viene completamente eseguito fino al suo return., Ogni comportamento di un agente definisce un metodo done.

Domanda **15**

Parzialmente corretta

Punteggio ottenuto 2,00 su 3,00

Contrassegna domanda

Si consideri il seguente frammento di codice di un metodo action di un behaviour Jade.

```
public void action() {
    ...
    MessageTemplate tpl = MessageTemplate.MatchOntology("Test-Ontology");
    ACLMessage msg = myAgent.receive(tpl);
    if (msg != null) {
        // Process the message
    }
    else {
        block();
    }
    ...
}
```

Quali delle seguenti affermazioni sono vere?

Scegli una o più alternative:

- [x] a. La coda dei messaggi è comune a tutti i comportamenti di uno stesso agente. ✓
- [x] b. Se un messaggio non soddisfa il template tpl non viene prelevato dalla coda dei messaggi. ✓
- [x] c. Il processo dell'agente risulta in wait all'esecuzione dell'istruzione block() sino alla ricezione del prossimo messaggio. ✗
- [ ] d. Alla ricezione di un messaggio l'esecuzione riprende dal punto successivo l'istruzione block().
- [ ] e. L'esecuzione dell'istruzione block() ferma immediatamente l'esecuzione del comportamento che la contiene.

Risposta parzialmente esatta.

Hai selezionato troppe opzioni.
La risposta corretta è: Se un messaggio non soddisfa il template tpl non viene prelevato dalla coda dei messaggi., La coda dei messaggi è comune a tutti i comportamenti di uno stesso agente.

Domanda 1

Parzialmente corretta

Punteggio ottenuto 0,25 su 1,50

Contrassegna domanda

Si considerino le intenzioni, quali delle seguenti affermazioni sono vere?

Scegli una o più alternative:

- a. Le intenzioni sono piani, sequenze di azioni che un agente esegue per raggiungere un suo goal.
- b. Le intenzioni sono la sorgente della proattività di un agente. ✓
- c. Le intezioni sono dei vincoli che un agente deve rispettare nella scelta delle azioni da eseguire.
- d. Le intezioni di un agente sono soggette ad un processo di revisione dopo ogni azione da esso eseguita. ✗
- e. Le intezioni di un agente non sono mai soggette ad un processo di revisione.

Risposta parzialmente esatta.

Hai selezionato correttamente 1.
La risposta corretta è: Le intezioni sono dei vincoli che un agente deve rispettare nella scelta delle azioni da eseguire., Le intenzioni sono la sorgente della proattività di un agente.

Domanda 2

Completo

Punteggio ottenuto 0,50 su 2,00

Contrassegna domanda

Spiegare che cosa si intende per **Model checking**.

Model checking e' una tecnica usata per controllare se una data formula ø e' conseguenza logica di un dato modello M.
Si basa sulla logica temporale lineare.

Commento:

Commento:

Domanda 9
Completo
Punteggio ottenuto 3,00 su 3,00
Contrassegna domanda

Scrivere l'Agent Control Loop per l'architettura BDI nella versione senza Commitment Strategy (Versione 3), spiegando la notazione utilizzata.

```
B = B0
I = I0
whlie true do
 get next percept p
 B = brf (B,p)
 D = options (B,I)
 I = filter (B,D,I)
 π = plan (B,I)
 execute (π)
end while
```

l'agente percepisce l'ambiente ed aggiorna le proprie beliefs con la funzione brf, dopo produce i piani eseguibili in base ai propri desideri con la funzione options, successivamente con la funzione filter sceglie un intenzione da eseguire ed infine con la funzione plan viene prodotto un piano di azioni che viene poi eseguito dall'execute.

```
B := B0;
I := I0;
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);
end while
```

(a) {C(m,c,T,goods), C(c,m,goods,pay)}

(b) {C(m,c,pay,receipt)}

(c) {}

Commento:

Domanda 8

Completo

Punteggio ottenuto 0,00 su 3,00

Contrassegna domanda

Spiegare il significato della seguente "commitment rule" del linguaggio Agent-0.

```
1 (COMMIT
2   (?agent REQUEST (DO ?time ?action))
3   (AND (B ?now (Friend ?agent)) (CAN myself ?action))
4   (myself (DO ?time ?action)))
```

I commitment rule di Agent-0 definiscono il comportamento dell'agente. In questo caso l'agente richiede un'azione da un agente amico.

Commento:

Domanda 9

Completo

Punteggio ottenuto 3,00 su 3,00

Contrassegna domanda

Scrivere l'Agent Control Loop per l'architettura BDI nella versione senza Commitment Strategy (Versione 3), spiegando la notazione utilizzata.

```
B = B0
I = I0
whlie true do
 get next percept p
 B = brf (B,p)
 D = options (B,I)
 I = filter (B,D,I)
 π = plan (B,I)
 execute (π)
```

Risposta corretta.

La risposta corretta è: Il practical reasoning è il ragionamento che riguarda le azioni (actions)., Il theoretical reasoning è il ragionamento che riguarda le credenze (beliefs).

Domanda 7

Completo

Punteggio ottenuto 0,00 su 2,00

Contrassegna domanda

Si consideri il seguente protocollo a commitment:

request means none
quote means create(C(m, c, accept, goods))
accept means create(C(c, m, goods, pay))
goods means create(C(m, c, pay, receipt))
pay means none
receipt means none

Scrivere l'insieme di commitment nelle seguenti situazioni:

(a) sono state eseguite le azioni (in sequenza): request, quote, accept;

(b) sono state eseguite le azioni (in sequenza): request, quote, goods;

(d) sono state eseguite le azioni (in sequenza): quote, accept, goods, pay, receipt.

A

C(c,accept), C(m,goods), C(c,pay)

B

C(c,accept), C(m,goods), C(c,pay), C(m,receipt)

D

C(c,accept), C(m,goods), C(c,pay), C(m,receipt)

(a) {C(m,c,T,goods), C(c,m,goods,pay)}

(b) {C(m,c,pay,receipt)}

(c) {}

Commento:

Domanda 5
Risposta errata
Punteggio ottenuto 0,00 su 2,00
Contrassegna domanda

Si consideri la formula "true U α" della logica temporale lineare (LTL). Qual è il significato della formula?

Scegli una o più alternative:
- [x] a. α sarà sempre vera. ✗
- [ ] b. Prima o poi α sarà vera.

Risposta errata.
La risposta corretta è: Prima o poi α sarà vera.

Domanda 6
Risposta corretta
Punteggio ottenuto 1,00 su 1,00
Contrassegna domanda

Quale o quali delle seguenti affermazioni sono vere?

Scegli una o più alternative:
- [ ] a. Il practical reasoning è un esempio di ragionamento epistemico.
- [x] b. Il theoretical reasoning è il ragionamento che riguarda le credenze (beliefs). ✓
- [x] c. Il practical reasoning è il ragionamento che riguarda le azioni (actions). ✓
- [ ] d. Il practical reasoning è il ragionamento che riguarda le credenze (beliefs).

Risposta corretta.
La risposta corretta è: Il practical reasoning è il ragionamento che riguarda le azioni (actions)., Il theoretical reasoning è il ragionamento che riguarda le credenze (beliefs).

Domanda 7

Si consideri il seguente protocollo a commitment:

14
15
16
17
18
19
20

Adding first behaviour
Adding second behaviour
Adding third behaviour
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20

Commento:

[6 - Mul... AgInt-1... Esame ... Esame ... [rma@... [sniffer... [/home... [/home... [Termin... Scarica... cpu mem swap 1 giorno 0,0 k 25 giu, 17:05


```
            if ((counter % 2) == 1)
                System.out.println(counter);
        }
        public boolean done() {
          return counter >= 20;
        }
      } );
      System.out.println("Adding third behaviour");
      addBehaviour(new Behaviour() {
        public void action() {
          counter++;
        }
        public boolean done() {
          return counter >= 20;
        }
      } );

    }
    protected void takeDown() {
      System.out.println("Bye...");
    }
}
```

Dire cosa viene stampato a console durante l'esecuzione.

Adding first behaviour

Adding second behaviour

Adding third behaviour

0

1

2

3

4

5

6

7

intelligente può essere generato senza ragionamento simbolico.

Domanda 4

Completo

Punteggio ottenuto 3,00 su 3,00

Contrassegna domanda

Si consideri il seguente programma Jade:

```
import jade.core.Agent;
import jade.core.behaviours.Behaviour;

public class Esame20190618 extends Agent {
   private int counter = 0;
   protected void setup() {
      System.out.println("Adding first behaviour");
      addBehaviour(new Behaviour() {
         public void action() {
            if ((counter % 2) = = 0)
               System.out.println(counter);
         }
         public boolean done() {
           return counter > = 20;
         }
      } );
      System.out.println("Adding second behaviour");
      addBehaviour(new Behaviour() {
         public void action() {
            if ((counter % 2) = = 1)
               System.out.println(counter);
         }
         public boolean done() {
           return counter > = 20;
         }
      } );
      System.out.println("Adding third behaviour");
      addBehaviour(new Behaviour() {
         public void action() {
            counter + + ;
         }
         public boolean done() {
           return counter > = 20;
         }
      } );
```

Scegli una o più alternative:

- [ ] a. (*Alice*: *a1*, *Bob*: *a1*) è *Pareto optimal*.
- [x] b. (*Alice*: *a2*, *Bob*: *a1*) è un *equilibrio di Nash*. ✓
- [ ] c. (*Alice*: *a2*, *Bob*: *a2*) è un *equilibrio di Nash*.

Risposta corretta.

La risposta corretta è: (*Alice*: *a2*, *Bob*: *a1*) è un *equilibrio di Nash*.

Domanda 3
Parzialmente corretta
Punteggio ottenuto 0,50 su 1,00
Contrassegna domanda

Quali tesi sostiene Brooks?

Scegli una o più alternative:

- [x] a. Un comportamento intelligente può essere generato senza rappresentazione esplicita della conoscenza. ✓
- [ ] b. Un comportamento intelligente può essere generato senza ragionamento simbolico.
- [ ] c. Un comportamento intelligente non dipende dall'ambiente.

Risposta parzialmente esatta.

Hai selezionato correttamente 1.
La risposta corretta è: Un comportamento intelligente può essere generato senza rappresentazione esplicita della conoscenza., Un comportamento intelligente può essere generato senza ragionamento simbolico.

Domanda 4
Completo
Punteggio ottenuto 3,00 su 3,00

Si consideri il seguente programma Jade:

```
import jade.core.Agent;
import jade.core.behaviours.Behaviour;

public class Esame20190618 extends Agent {
    private int counter = 0;
    protected void setup() {
```

La risposta corretta è: F1 è un assioma della logica epistemica e F2 non lo è.

Domanda 2
Risposta corretta
Punteggio ottenuto 3,00 su 3,00
Contrassegna domanda

Si consideri la seguente **matrice di payoff**, dove sia *Alice* che *Bob* possono eseguire o l'azione *a1* o *a2*.

	Alice: a1	Alice: a2
Bob: a1	A = 0, B = 0	A = 1, B = 3
Bob: a2	A = 3, B = 1	A = 2, B = 2

Scegli una o più alternative:

- [ ] a. (*Alice*: *a1*, *Bob*: *a1*) è *Pareto optimal*.
- [x] b. (*Alice*: *a2*, *Bob*: *a1*) è un *equilibrio di Nash*. ✓
- [ ] c. (*Alice*: *a2*, *Bob*: *a2*) è un *equilibrio di Nash*.

Risposta corretta.
La risposta corretta è: (*Alice*: *a2*, *Bob*: *a1*) è un *equilibrio di Nash*.

Domanda 3

Quali tesi sostiene Brooks?

Domanda 15
Completo
Punteggio ottenuto 0,50 su 3,00
Contrassegna domanda

Si consideri il seguente frammento di codice AgentSpeak(L):

```
adjacent(a,b).
adjacent(b,c).
adjacent(c,d).
location(robot,b).
location(waste,c).
location(car,d).
location(bin,a).

(P1)
+location(waste, X): location(robot,X) & location(bin,Y)
  <- pick(waste); !location(robot, Y); drop(waste).
(P2)
+!location(robot,X): location(robot,X) <- true.
(P3)
+!location(robot,X): location(robot,Y) & (not (X = Y)) & adjacent(Y,Z) & (not (location(car,Z)))
  <- move(Y,Z); !location(robot,X).
```

ed il goal: +!location(robot,c).

Scrivere (a) i piani rilevanti e l'unificatore rilevante (b) il piano applicabile e l'unificatore applicabile (c) l'intended means risultante dall'applicazione dell'unificatore applicabile.

inserendo il goal il piano rilevante unificato è p3 ed è anche applicabile in quanto le precondizioni unificano, l'intended means produce la move dell'agente e al'aggiunta del goal della nuova posizione. Quest'ultimo unifica con p2 ed è applicabile unificando le precondizioni, l'intended means aggiunge alla base di conoscenza la nuova posizione

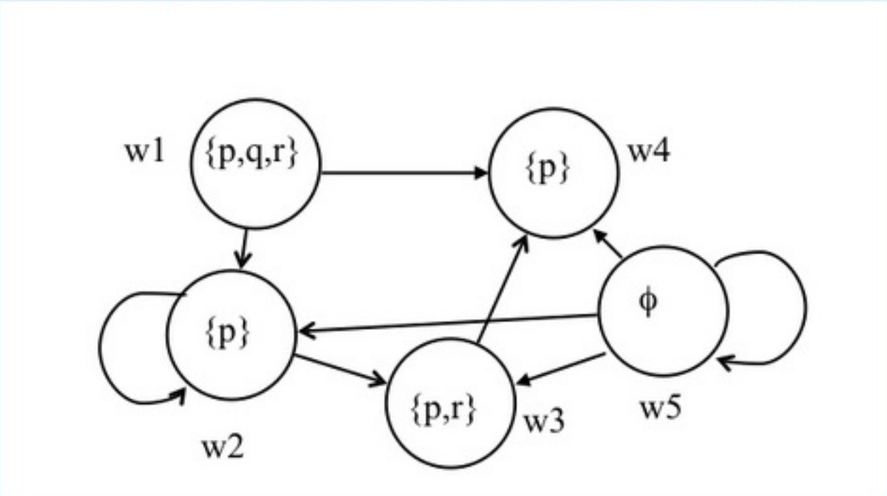
(a) θ = {X/c} P2 e P3 piani rilevanti.

(b) $\sigma\theta$ = {X/c, Y/b, Z/c} P3 applicabile.

(c) move(b,c); !location(robot,c).

Domanda 14
Parzialmente corretta
Punteggio ottenuto 2,00 su 3,00
Contrassegna domanda

Si consideri il seguente modello. Dire quali delle seguenti affermazioni sono vere.



Scegli una o più alternative:

- [x] a. <>r è vero nel mondo w5. ✓
- [ ] b. []p è vero nel mondo w5.
- [x] c. []p è vero nel mondo w2. ✓
- [ ] d. []r è vero nel mondo w4.
- [ ] e. <>r è vero nel mondo w4.

Risposta parzialmente esatta.
Hai selezionato correttamente 2.
La risposta corretta è: []p è vero nel mondo w2., <>r è vero nel mondo w5., []r è vero nel mondo w4.

Domanda 15
Completo

Si consideri il seguente frammento di codice AgentSpeak(L):

Contrassegna domanda

Il significato inteso da parte dell'oratore che vuole rendere noto all'ascoltatore. Illocuzione ✓

Risposta corretta.

La risposta corretta è: La produzione fisica di enunciati da parte di un oratore. – Locuzione, Il risultato, le conseguenze provocate dall'atto comunicativo. – Perlocuzione, Il significato inteso da parte dell'oratore che vuole rendere noto all'ascoltatore. – Illocuzione

Domanda 13

Parzialmente corretta

Punteggio ottenuto 0,25 su 1,00

Contrassegna domanda

Quale/quali delle seguenti affermazioni sono vere?

Scegli una o più alternative:

- [ ] a. Le intenzioni sono persistenti.
- [ ] b. Le intenzioni sono il risultato della fase deliberativa del practical reasoning.
- [ ] c. Un agente può non credere che le sue intenzioni siano raggiungibili.
- [x] d. L'insieme dei desire di un agente può non essere consistente. ✓
- [ ] e. Le intenzioni sono il risultato della fase means-ends reasoning del practical reasoning.
- [x] f. L'insieme delle intenzioni di un agente può non essere consistente. ✗
- [ ] g. Un'insieme di intenzioni di un agente non influenza la sua attività di practical reasoning.
- [x] h. Un agente crede che le sue intenzioni siano raggiungibili. ✓

Risposta parzialmente esatta.

Hai selezionato correttamente 2.

La risposta corretta è: Le intenzioni sono il risultato della fase deliberativa del practical reasoning., Le intenzioni sono persistenti., Un agente crede che le sue intenzioni siano raggiungibili., L'insieme dei desire di un agente può non essere consistente.

Domanda 14

Parzialmente corretta

Si consideri il seguente modello. Dire quali delle seguenti affermazioni sono vere.

Contrassegna domanda

Risposta errata.

La risposta corretta è: Un equilibrio con strategie dominanti è anche un equilibrio di Nash.

Domanda 12

Risposta corretta

Punteggio ottenuto 1,00 su 1,00

Contrassegna domanda

Associare il corretto significato.

La produzione fisica di enunciati da parte di un oratore.	Locuzione ✓
Il risultato, le conseguenze provocate dall'atto comunicativo.	Perlocuzione ✓
Il significato inteso da parte dell'oratore che vuole rendere noto all'ascoltatore.	Illocuzione ✓

Risposta corretta.

La risposta corretta è: La produzione fisica di enunciati da parte di un oratore. – Locuzione, Il risultato, le conseguenze provocate dall'atto comunicativo. – Perlocuzione, Il significato inteso da parte dell'oratore che vuole rendere noto all'ascoltatore. – Illocuzione

Domanda 13

Parzialmente corretta

Punteggio ottenuto 0,25 su 1,00

Contrassegna

Quale/quali delle seguenti affermazioni sono vere?

Scegli una o più alternative:

- [ ] a. Le intenzioni sono persistenti.
- [ ] b. Le intenzioni sono il risultato della fase deliberativa del practical reasoning.
- [ ] c. Un agente può non credere che le sue intenzioni siano raggiungibili.
- [x] d. L'insieme dei desire di un agente può non essere consistente. ✓
- [ ] e. Le intenzioni sono il risultato della fase means-ends reasoning del practical reasoning.

Domanda 10
Parzialmente corretta
Punteggio ottenuto 0,50 su 1,00
Contrassegna domanda

Associare correttamente i significati.

Abilita la coordinazione tra i comportamenti di un insieme di agenti.	Coordinazione ✗
La proprietà di un sistema di agenti di eseguire attività competitive o self-interested in un ambiente condiviso.	Negozione ✓
La proprietà di un sistema di agenti di eseguire attività non antagoniste in un ambiente condiviso.	Cooperazione ✓
La proprietà di un sistema di agenti di eseguire attività in un ambiente condiviso.	Comunicazione ✗

Risposta parzialmente esatta.

Hai selezionato correttamente 2.
La risposta corretta è: Abilita la coordinazione tra i comportamenti di un insieme di agenti. – Comunicazione, La proprietà di un sistema di agenti di eseguire attività competitive o self-interested in un ambiente condiviso. – Negozione, La proprietà di un sistema di agenti di eseguire attività non antagoniste in un ambiente condiviso. – Cooperazione, La proprietà di un sistema di agenti di eseguire attività in un ambiente condiviso. – Coordinazione

Domanda 11
Risposta errata
Punteggio ottenuto 0,00 su 1,00
Contrassegna domanda

Dire cosa è vero.

Scegli una o più alternative:
- [x] a. Un equilibrio di Nash è anche un equilibrio con strategie dominanti. ✗
- [ ] b. Un equilibrio con strategie dominanti è anche un equilibrio di Nash.

Risposta errata.

La risposta corretta è: Un equilibrio con strategie dominanti è anche un equilibrio di Nash.

[6 - Mul... AgInt-1... Esame ... Esame ... [rma@... [sniffer... [/home... [/home... [Termin... Scarica... cpu mem swap 1 giorno 0,0 k 25 giu, 17:04
Esame del 18 giugno 2019 - Mozilla Firefox
https://informatica.i-learn.unito.it/mod/quiz/review.php?attempt=44571 80%

NAVIGAZIONE QUIZ
1 2 3 4 5 6 7 8 9
10 11 12 13 14 15
Fine revisione

Iniziato	martedì, 18 giugno 2019, 09:18
Stato	Completato
Terminato	martedì, 18 giugno 2019, 10:38
Tempo impiegato	1 ora 20 min.
Valutazione	**16,75** su un massimo di 30,00 (**56**%)

Domanda 1
Risposta corretta
Punteggio ottenuto 2,00 su 2,00
Contrassegna domanda

Si considerino le formule F1: $K\phi \Rightarrow \phi$ e F2: $B\phi \Rightarrow \phi$ in figura, dove K e B sono gli operatori di knowledge e belief. Dire quali delle seguenti affermazioni sono vere

Scegli una o più alternative:
- [x] a. F1 è un assioma della logica epistemica e F2 non lo è. ✓
- [ ] b. F1 e F2 sono assiomi della logica epistemica.
- [ ] c. F2 è un assioma della logica epistemica e F1 non lo è.

Risposta corretta.
La risposta corretta è: F1 è un assioma della logica epistemica e F2 non lo è.

Domanda 2
Risposta corretta
Punteggio ottenuto 3,00 su 3,00
Contrassegna domanda

Si consideri la seguente **matrice di payoff**, dove sia *Alice* che *Bob* possono eseguire o l'azione *a1* o *a2*.

	Alice: a1	Alice: a2
Bob: a1	A = 0, B = 0	A = 1, B = 3

Domanda 3

Risposta corretta

Punteggio ottenuto 3,00 su 3,00

Contrassegna domanda

Dire quali delle seguenti affermazioni relative alle logiche temporali sono vere e quali false.

$F\alpha \equiv true\ \mathbf{U}\ \alpha$ significa che α sarà sempre vera	Falso ✓
$\mathbf{AG}(EFp)$ è una formula CTL	Vero ✓
$\mathbf{G}(p \Rightarrow \mathbf{X}q)$ è una formula LTL che è soddisfatta da tutte le sequenze in cui p è vero infinite volte	Falso ✓
$\mathbf{FG}p$ è una formula CTL	Falso ✓

Risposta corretta.

Domanda 4

Risposta corretta

Punteggio ottenuto 3,00 su 3,00

Contrassegna

Si consideri un gioco in cui sia Alice che Bob possono eseguire o l'azione a1 o l'azione a2.
Dire quali delle seguenti affermazioni sono vere.

Se (Alice:a1, Bob:a1) è un equilibrio di Nash, allora a1 è una strategia dominante sia per Alice che per Bob.	Falso ✓
Se a1 è una strategia dominante sia per Alice che per Bob, allora (Alice:a1, Bob:a1) è un equilibrio di Nash.	Vero ✓

Domanda 2

Parzialmente corretta

Punteggio ottenuto 2,40 su 3,00

Contrassegna domanda

Associare le affermazioni con la corretta descrizione.

Non sarebbe razionale da parte mia adottare l'intenzione ϕ se credessi che ϕ non fosse possibile.	Gli agenti non credono che non porteranno avanti le loro intenzioni
Se ho l'intenzione ϕ, non ci si aspetta che adotti un'intenzione ψ tale che ϕ e si escludono a vicenda.	Le intenzioni forniscono un "filtro" per l'adozione di altre intenzioni, che non devono entrare in conflitto
Se ho l'intenzione ϕ, ci si aspetta che che dedichi risorse per decidere come portare avanti ϕ.	Gli agenti non credono che non porteranno avanti le loro intenzioni
Se credo $\phi \rightarrow \psi$ e intendo che ϕ, non intendo necessariamente anche ψ.	Gli agenti non devono necessariamente avere intenzione su tutti gli effetti colleterali delle loro intenzioni
Normalmente non sarebbe razionale da parte mia credere che avrei realizzato le mie intenzioni; le intenzioni possono fallire. Inoltre, non ha senso che se credo ϕ sia inevitabile che io lo adotti come intenzione.	In determinate circostanze, gli agenti credono che realizzeranno le loro intenzioni

Domanda **1**

Parzialmente corretta

Punteggio ottenuto 2,40 su 3,00

Contrassegna domanda

Quali delle seguenti affermazioni sono vere e quali false?

Il problema della rappresentazione/ragionamento di un agente deliberativo, è il problema concernente il tempo impiegato per tradurre il mondo reale, un ambiente, in una descrizione simbolica accurata.	Falso ✓
Il problema sottostante della trasduzione di un agente deliberativo, risiede nella complessità degli algoritmi di manipolazione.	Vero ✓
Le risorse di calcolo disponibili attualmente hanno permesso di superare il problema della trasduzione di un agente deliberativo.	Falso ✓
Il problema della rappresentazione/ragionamento di un agente deliberativo, è il problema concernente il tempo di ragionare su una rappresentazione simbolica del mondo e il modo con cui questo è rappresentato.	Vero ✓
Il problema sottostante della rappresentazione/ragionamento di un agente deliberativo, risiede nella complessità degli algoritmi di manipolazione.	Falso ✗

Risposta parzialmente esatta.

Hai selezionato correttamente 4.

Domanda 3

Completo

Punteggio ottenuto 4,50 su 4,50

Contrassegna domanda

Commentare la formula che definisce l'intenzione di eseguire una azione α secondo Cohen e Levesque:

(INTEND i α) ≡ (P-GOAL i [DONE i (BEL i (HAPPENS α))? ; α])

La formula esprime che l'agente i intende eseguire alpha (una azione) se ha un goal persistente di raggiungere lo stato in cui riteneva di essere arrivato per eseguire alpha, quindi esegue l'azione alpha. L'idea è quella di far applicare l'azione alpha in modo voluto e non accidentale, assicurandosi che l'agente sia committed per garantire questa cosa.

Commento:

Domanda 4

Completo

Punteggio ottenuto 4,00 su 4,50

Contrassegna domanda

Un agente Jade può eseguire più comportamenti in parallelo, però il "behaviour scheduling" è **non preemptive** ma **cooperativo** e tutto viene eseguito in un singolo Thread.

Si commenti l'affermazione.

Lo scheduling è cooperativo nel senso che si prefissa di eseguire comportamenti in modo da far collaborare diverse azioni nel raggiungere il goal. Il cambio di comportamento avviene solo ed esclusivamente alla fine dell'esecuzione del metodo "action", non quando il comportamento è effettivamente terminato. Questo permette di eseguire diverse azioni di diversi comportamenti, garantendo un interleaving in grado di esplorare le soluzioni in modo migliore.

Commento:
"non quando il comportamento è effettivamente terminato", detto i questo modo non si comprende cosa vuole dire perché appena prima ha detto che il cambio avviene quando è completata l'esecuzione della action.

Preemptive cosa significa? Non scrive, questo è il significato.

Domanda 2

Completo

Punteggio ottenuto 4,50 su 4,50

Contrassegna domanda

Il GOLOG è un linguaggio di programmazione che fa uso del situation calculus implementato con la logica classica. Spiegare il significato dei "Successor state axioms" facendo riferimento al seguente esempio:

$$Poss(a, s) \Rightarrow [broken(x, do(a, s)) \equiv$$

$$\exists r\{a = drop(r, x) \wedge fragile(x, s)\} \vee$$

$$broken(x, s) \wedge \neg \exists r\{a = repair(r, x)\}]$$

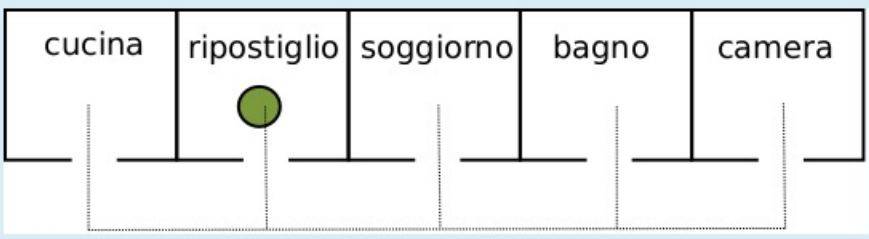
Il successor state axiom è un assioma utilizzato nel calcolo delle situazioni. In particolare, viene utilizzato per formalizzare tutto ciò che dopo l'esecuzione di una azione non è cambiato.

L'esempio riportato esprime che se un oggetto X è "broken", il suo stato è risultato di due possibili azioni: o si è verificata una azione "drop" e l'oggetto era "fragile" oppure l'oggetto era già "broken" e non è stata eseguita alcuna azione "repair"

Commento:

```
/* Descrizione del mondo */

???? (A) ????

direzione(destra).

/* Initial goals */

???? (B) ????

/* Plans */

???? (C) ????

+!pulisci(Stanza) : dentro(Stanza)
   <-   !aspira.

+!pulisci(Stanza) : dentro(AltraStanza) & not (Stanza = AltraStanza)
   <-   !esci(AltraStanza); !vaidifronte(Stanza); !entra(Stanza); !pulisci(Stanza).

+!esci(Stanza) : dentro(Stanza)
   <-   -dentro(Stanza); +difronte(Stanza);
      .print("Esco da ", Stanza).

+!entra(Stanza)   : difronte(Stanza)
   <-   -difronte(Stanza);   +dentro(Stanza);
      .print("Entro in ", Stanza).

+!vaidifronte(Stanza) : difronte(Stanza)
   <-   true.

+!vaidifronte(Stanza) : difronte(AltraStanza) & not (AltraStanza = Stanza)
   <-   !spostaAfianco(AltraStanza);
+!spostaAfianco(Stanza) : direzione(sinistra) & afianco(Stanza, AltraStanza)
   <-   -difronte(Stanza); +difronte(AltraStanza);
      .print(Stanza, " -> ", AltraStanza).

+!spostaAfianco(Stanza) : direzione(sinistra) &   not afianco(Stanza, AltraStanza)
   <-   -direzione(sinistra); +direzione(destra); !spostaAfianco(Stanza).

+!spostaAfianco(Stanza) : direzione(destra) & afianco(AltraStanza, Stanza)
   <-   -difronte(Stanza); +difronte(AltraStanza);
      .print(AltraStanza, " <- ", Stanza).

+!spostaAfianco(Stanza)   : direzione(destra) & not afianco(AltraStanza, Stanza)
   <-   -direzione(destra);   +direzione(sinistra); !spostaAfianco(Stanza).

+!aspira : dentro(Stanza)
   <-   .print("Aspiro : ", Stanza); -sporco(Stanza).
```

Dopo aver studiato il programma, si risponda alle seguenti domande:



(1) Si aggiunga la descrizione del mondo in (A) necessaria per il funzionamento del programma nella situazione della figura, dove l'aspirapolvere si trova dentro il ripostiglio, la cucina è a fianco del ripostiglio, il ripostiglio è a fianco del soggiorno, e così via.

(2) Quale evento goal si potrebbe inserire in (B) affinché l'aspirapolvere vada a pulire la camera?

(3) Si sipponga l'evento di aggiunta del belief sporco(camera), quale piano o piani sarebbe necessario aggiungere in (C) perché l'aspirapolvere vada a pulire la camera? Se l'evento aggiunto fosse sporco([camera, bagno]), quale piano o piani sarebbe necessario aggiungere perché l'aspirapolvere pulisca la camera e poi il bagno?

Punteggio ottenuto 3,50 su 4,50

A

dentro(Ripostiglio)

afianco(cucina, ripostiglio)

afianco(ripostiglio, soggiorno)

afianco(soggiorno, bagno)

afianco(bagno, camera)

B

!pulisci(camera)

C

```
+!pulisci(camera) : dentro(AltraStanza) & not (AltraStanza = camera)
   <-   !esci(AltraStanza); !vaidifronte(camera); !entra(camera); !pulisci(camera).


+!pulisci(camera, bagno) : dentro(AltraStanza) & not (AltraStanza = bagno) & not (AltraStanza = camera)
   <-   !esci(AltraStanza); !vaidifronte(camera); !entra(camera); !pulisci(camera); !esci(camera); !vaidifronte(bagno); !entra(bagno);
!pulisci(bagno).
```

Commento:

3)

```
+sporco(Stanza <- !pulisci(Stanza).


+sporco([])
   <- .print("Finito!").

+sporco([Stanza|AltreStanze])
   <- .print("Pulisco ", Stanza); !pulisci(Stanza); +sporco(AltreStanze).
```

A

```
dentro(Ripostiglio)
afianco(cucina, ripostiglio)
afianco(ripostiglio, soggiorno)
afianco(soggiorno, bagno)
afianco(bagno, camera)
```

B

```
!pulisci(camera)
```

C

```
+!pulisci(camera) : dentro(AltraStanza) & not (AltraStanza = camera)
   <-    !esci(AltraStanza); !vaidifronte(camera); !entra(camera); !pulisci(camera).


+!pulisci(camera, bagno) : dentro(AltraStanza) & not (AltraStanza = bagno) & not (AltraStanza = camera)
   <-    !esci(AltraStanza); !vaidifronte(camera); !entra(camera); !pulisci(camera); !esci(camera); !vaidifronte(bagno); !entra(bagno);
!pulisci(bagno).
```

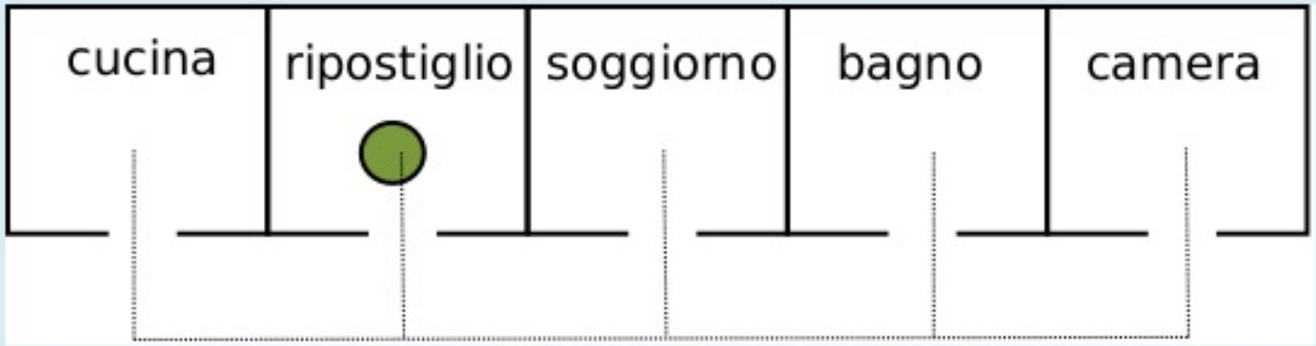
Commento:

3)

```
+sporco(Stanza <- !pulisci(Stanza).


+sporco([])
   <- .print("Finito!").

+sporco([Stanza|AltreStanze])
   <- .print("Pulisco ", Stanza); !pulisci(Stanza); +sporco(AltreStanze).
```

```
+!spostaAfianco(Stanza) : direzione(sinistra) & afianco(Stanza, AltraStanza)
   <-    -difronte(Stanza); +difronte(AltraStanza);
       .print(Stanza, " -> ", AltraStanza).

+!spostaAfianco(Stanza) : direzione(sinistra) &    not afianco(Stanza, AltraStanza)
   <-    -direzione(sinistra); +direzione(destra); !spostaAfianco(Stanza).

+!spostaAfianco(Stanza) : direzione(destra) & afianco(AltraStanza, Stanza)
   <-    -difronte(Stanza); +difronte(AltraStanza);
       .print(AltraStanza, " <- ", Stanza).

+!spostaAfianco(Stanza)    : direzione(destra) & not afianco(AltraStanza, Stanza)
   <-    -direzione(destra);    +direzione(sinistra); !spostaAfianco(Stanza).

+!aspira : dentro(Stanza)
   <-    .print("Aspiro : ", Stanza); -sporco(Stanza).
```

Dopo aver studiato il programma, si risponda alle seguenti domande:



(1) Si aggiunga la descrizione del mondo in (A) necessaria per il funzionamento del programma nella situazione della figura, dove l'aspirapolvere si trova dentro il ripostiglio, la cucina è a fianco del ripostiglio, il ripostiglio è a fianco del soggiorno, e così via.

(2) Quale evento goal si potrebbe inserire in (B) affinché l'aspirapolvere vada a pulire la camera?

(3) Si sipponga l'evento di aggiunta del belief sporco(camera), quale piano o piani sarebbe necessario aggiungere in (C) perché l'aspirapolvere vada a pulire la camera? Se l'evento aggiunto fosse sporco([camera, bagno]), quale piano o piani sarebbe necessario aggiungere perché l'aspirapolvere pulisca la camera e poi il bagno?

Completo

Punteggio ottenuto 3,50 su 4,50

Contrassegna domanda

```
/* Descrizione del mondo */

???? (A) ????

direzione(destra).

/* Initial goals */

???? (B) ????

/* Plans */

???? (C) ????

+!pulisci(Stanza) : dentro(Stanza)
   <-    !aspira.

+!pulisci(Stanza) : dentro(AltraStanza) & not (Stanza = AltraStanza)
   <-    !esci(AltraStanza); !vaidifronte(Stanza); !entra(Stanza); !pulisci(Stanza).

+!esci(Stanza) : dentro(Stanza)
   <-    -dentro(Stanza); +difronte(Stanza);
       .print("Esco da ", Stanza).

+!entra(Stanza)    : difronte(Stanza)
   <-    -difronte(Stanza);    +dentro(Stanza);
       .print("Entro in ", Stanza).

+!vaidifronte(Stanza) : difronte(Stanza)
   <-    true.

+!vaidifronte(Stanza) : difronte(AltraStanza) & not (AltraStanza = Stanza)
   <-    !spostaAfianco(AltraStanza);
```

Domanda 3

Completo

Punteggio ottenuto 3,50 su 4,50

Contrassegna domanda

Si consideri le seguenti classi Java per JADE:

```
import jade.core.behaviours.Behaviour;

public class MyThreeStepBehaviour extends Behaviour {
   private int step = 0;
   public void action() {
      switch (step) {
      case 0:
         System.out.println("Operation X");
         step++;
         break;
      case 1:
         System.out.println("Operation Y");
         step++;
         myAgent.doDelete();
         break;
      case 2:
         System.out.println("Operation Z");
         step++;
         break;
      }
      System.out.println("Completed execution MyThreeStepBehaviour, step: " + step + ".");
   }
   public boolean done() {
      return step == 3;
   }
}
```

```
import jade.core.behaviours.Behaviour;

public class MySimpleThreeStepBehaviour extends Behaviour {
   private int step = 0;
   public void action() {
      System.out.println("Operation W");
```

Domanda 4

Completo

Punteggio ottenuto 4,50 su 4,50

Contrassegna domanda

Logica modale: definire l'**assioma K** e la regola di **necessitation**.

L'assioma K e la regola di necessitation sono due proprietà fondamentali che valgono su qualunque logica modale.

- Assioma K: "box"(alfa implica beta) implica (("box" alfa) implica ("box" beta)).

- Necessitation: è una regola di inferenza. Se fi è valida allora "box" fi è valida.

Si consideri le seguenti classi Java per JADE:

```
import jade.core.behaviours.Behaviour;

public class MyThreeStepBehaviour extends Behaviour {
   private int step = 0;
   public void action() {
      switch (step) {
      case 0:
         System.out.println("Operation X");
         step++;
         break;
      case 1:
         System.out.println("Operation Y");
         step++;
         myAgent.doDelete();
         break;
      case 2:
         System.out.println("Operation Z");
         step++;
         break;
      }
      System.out.println("Completed execution MyThreeStepBehaviour, step: " + step + ".");
   }
   public boolean done() {
      return step == 3;
   }
}
```

Punteggio ottenuto 3,50 su 4,50

```
import jade.core.behaviours.Behaviour;

public class MySimpleThreeStepBehaviour extends Behaviour {
   private int step = 0;
   public void action() {
      System.out.println("Operation W");
      step++;
      System.out.println("Completed execution MySimpleThreeStepBehaviour, step: " + step + ".");
   }
   public boolean done() {
      return step == 3;
   }
}
```

```
import jade.core.behaviours.OneShotBehaviour;

public class MyOneShotBehaviour extends OneShotBehaviour {
   public void action() {
      myAgent.addBehaviour(new MyThreeStepBehaviour());
   }
}
```

```
import jade.core.Agent;
import java.util.Iterator;

public class HelloWorldAgent4 extends Agent {
   protected void setup () {
      System.out.println("Hello World. I'am an agent!");
      System.out.println("My local-name is " + getAID().getLocalName());
      addBehaviour(new MyOneShotBehaviour());
      addBehaviour(new MySimpleThreeStepBehaviour());
   }
   protected void takeDown() {
      System.out.println("Agent " + getAID().getName() + " termination.");
   }
}
```

Il programma è compilato con il comando:

javac -cp /..../JADE-all-4.5.0/JADE-bin-4.5.0/jade/lib/jade.jar:. -d classes HelloWorldAgent4.java

ed eseguito con il comando:

java -cp /..../JADE-all-4.5.0/JADE-bin-4.5.0/jade/lib/jade.jar:classes jade.Boot -agents "peter:HelloWorldAgent4"

Riportare l'output a console per l'esecuzione.

```
Hello World. I am an agent!
My local-name is peter
Operation X
Completed execution MyThreeStepBehaviour, step: 1.
Operation Y
Agent peter termination.
```

Domanda 2

Completo

Punteggio ottenuto 4,50 su 4,50

Contrassegna domanda

Commentare la seguente formula che definisce un **persistent goal** secondo Cohene Levesque.

$$(\text{P-GOAL } i\ \varphi) \equiv (\text{GOAL } i\ (\text{LATER } \varphi)) \wedge (\text{BEL } i\ \neg\varphi) \wedge [\text{BEFORE } ((\text{BEL } i\ \varphi) \vee (\text{BEL } i\ \mathbf{G}\ \neg\varphi)) \neg(\text{GOAL } i\ (\text{LATER } \varphi))]$$

Un goal persistente "fi" è un achievement goal, ovvero un goal che "fi" prima o poi diventerà vero e si crede che attualmente "fi" sia falso. Inoltre prima che l'agente i abbandoni tale persistent goal "fi" deve valere una delle seguenti condizioni:

- l'agente i crede che il goal sia stato soddisfatto;

- l'agente i crede che il goal non sarà mai soddisfatto.

Si scriva lo stato sociale (insieme di commitment) validi dopo l'esecuzione delle seguenti sequenze di azioni nell'ipotesi che lo stato sociale iniziale sia vuoto:

1) propose_inv ; accept_inv ; issue_contract ; withdraw

2) propose_inv ; accept_inv ; issue_contract ; sign_contract ; invest

3) propose_inv ; accept_inv ; issue_contract ; sign_contract ; countersign_contract ; invest

4) propose_inv ; accept_inv ; issue_contract ; sign_contract ; countersign_contract ; invest ; notify

5) propose_inv ; accept_inv ; withdraw

Quali delle sequenze è un run (di successo) del protocollo?

1) Lo stato sociale risulta vuoto. È un run di successo perché non restano commitment pendenti.

2) Alla fine dell'esecuzione resta C(bank, investor, T, countersign_contract ^ notify_inv). Non è un run di successo.

3) Alla fine dell'esecuzione resta C(bank, investor, T, T ^ notify_inv). Non è un run di successo.

4) Lo stato sociale risulta vuoto. È un run di successo perché non restano commitment pendenti.

5) Lo stato sociale risulta vuoto. È un run di successo perché non restano commitment pendenti.

Domanda 1

Completo

Punteggio ottenuto 4,00 su 4,50

Contrassegna domanda

Si cosideri il seguente protocollo di vendita di titoli a commitment:

propose_inv **means** create(C(bank, investor, accept_inv, issue_contract))

accept_inv **means** create(C(investor, bank, issue_contract, sign_contract))

issue_contract **means** create(C(bank, investor, sign_contract ^ invest, countersign_contract ^ notify_inv))

sign_contract **means** create(C(investor, bank, countersign_contract, invest))

counter_sign **means** none

invest **menas** none

notify **means** none

withdraw **means**

release(C(bank, investor, sign_contract ^ invest, countersign_contract ^ notify_inv))

cancel(C(investor, bank, issue_contract, sign_contract))

release(C(bank, investor, accept_inv, issue_contract))

^ rappresenta "and".

Si scriva lo stato sociale (insieme di commitment) validi dopo l'esecuzione delle seguenti sequenze di azioni nell'ipotesi che lo stato sociale iniziale sia vuoto:

1) propose_inv ; accept_inv ; issue_contract ; withdraw

2) propose_inv ; accept_inv ; issue_contract ; sign_contract ; invest

3) propose_inv ; accept_inv ; issue_contract ; sign_contract ; countersign_contract ; invest

4) propose_inv ; accept_inv ; issue_contract ; sign_contract ; countersign_contract ; invest ; notify

5) propose_inv ; accept_inv ; withdraw

Quali delle sequenze è un run (di successo) del protocollo?

1) Lo stato sociale risulta vuoto. È un run di successo perché non restano commitment pendenti.

2) Alla fine dell'esecuzione resta C(bank, investor, T, countersign_contract ^ notify_inv). Non è un run di successo.

3) Alla fine dell'esecuzione resta C(bank, investor, T, T ^ notify_inv). Non è un run di successo.

4) Lo stato sociale risulta vuoto. È un run di successo perché non restano commitment pendenti.

```
/* Initial goals */

....

/* Plans */

(p1)
+!start(X)
   :    true
   <-     ?a(X); ?b(Y); .print("X: ", X, " Y: ", Y).

(p2)
+!start(X)
   :    a(X) & b(Y) & X = Y
   <-     .print("X: ", X, " Y: ", Y).

(p3)

+!start(3)
   :    a(X) & b(Y) & Y = X
   <-    .print("X: ", X, " Y: ", Y).

(p4)
+!start(X, Y)
   :    a(X) & b(Y) & Y = X
   <-    .print("X: ", X, " Y: ", Y).
```

Dire se le seguenti affermazioni sono vere o false.

Il goal !start(1) ha come piani rilevanti solo p1. Falso

Il goal !start(4, 4) non ha piani applicabili. Vero

Il goal !start(X) ha come piani rilevanti p1, p2, p3, p4. Falso

Il goal !start(X) può restituire come output X: 1 Y:2 Vero

Il goal !start(2) ha come piani applicabili p2. Vero

Domanda 3

Completo

Punteggio ottenuto 3,00 su 3,00

Contrassegna domanda

Nella **logica epistemica** si assume che per l'operatore **K** (conoscenza) valgano le seguenti proprietà, che possono essere modellate con assiomi:

p1) La conoscenza è non contraddittoria
p2) Ciò che si conosce è vero
p3) Introspezione positiva
p4) Introspezione negativa

Dire quali delle seguenti affermazioni sono vere e quali false.

L'assioma $K\phi \Rightarrow \neg K \neg \phi$ modella la proprietà p4 — Falso

L'assioma $K\phi \Rightarrow KK\phi$ modella la proprietà p3 — Vero

L'assioma $\neg K\phi \Rightarrow K(\neg K\phi)$ modella la proprietà p1 — Falso

L'assioma $K\phi \Rightarrow \phi$ modella la proprietà p2 — Vero

Domanda 4

Completo

Punteggio ottenuto 3,00 su 3,00

Contrassegna domanda

Si consideri il seguente programma Jason:

```
// Agent Test in project TESTJason.mas2j

/* Initial beliefs and rules */

a(1).
a(2).
a(3).

b(2).
b(3).
b(4).

/* Initial goals */

....
```

```
        addBehaviour(new MySimpleThreeStepBehaviour());
    }
    protected void takeDown() {
        System.out.println("Agent " + getAID().getName() + " termination.");
    }
}
```

Il programma è compilato con il comando:

javac -cp /..../JADE-all-4.5.0/JADE-bin-4.5.0/jade/lib/jade.jar:. -d classes HelloWorldAgent4.java

ed eseguito con il comando:

java -cp /..../JADE-all-4.5.0/JADE-bin-4.5.0/jade/lib/jade.jar:classes jade.Boot -agents "peter:HelloWorldAgent4"

Riportare l'output a console per l'esecuzione.

```
Hello World. I am an agent!
My local-name is peter
Operation X
Completed execution MyThreeStepBehaviour, step: 1.
Operation Y
Agent peter termination.
```

```
import jade.core.behaviours.Behaviour;

public class MySimpleThreeStepBehaviour extends Behaviour {
   private int step = 0;
   public void action() {
      System.out.println("Operation W");
      step++;
      System.out.println("Completed execution MySimpleThreeStepBehaviour, step: " + step + ".");
   }
   public boolean done() {
      return step == 3;
   }
}
```

```
import jade.core.behaviours.OneShotBehaviour;

public class MyOneShotBehaviour extends OneShotBehaviour {
   public void action() {
      myAgent.addBehaviour(new MyThreeStepBehaviour());
   }
}
```

```
import jade.core.Agent;
import java.util.Iterator;

public class HelloWorldAgent4 extends Agent {
   protected void setup () {
      System.out.println("Hello World. I'am an agent!");
      System.out.println("My local-name is " + getAID().getLocalName());
      addBehaviour(new MyOneShotBehaviour());
      addBehaviour(new MySimpleThreeStepBehaviour());
   }
   protected void takeDown() {
      System.out.println("Agent " + getAID().getName() + " termination.");
   }
}
```

Domanda 1

Completo

Punteggio ottenuto 3,00 su 3,00

Contrassegna domanda

Dire se le seguenti affermazioni sono vere o false.

L'intenzione è lo stato di cose che un agente ha scelto e al quale si impegna a raggiungere.	Vero
Le intenzioni non vincolano il futuro practical reasoning.	Falso
Se un agente fallisce un primo tentativo per ottenere l'intenzione a, esso tenterà di eseguire un piano alternativo per raggiungere la propria intenzione.	Vero
Gli agenti non credono che le proprie intenzioni siano possibili fino a che non le raggiungono.	Falso
Le intenzioni sono chiuse rispetto l'implicazione.	Falso

Domanda 2

Completo

Punteggio ottenuto 3,00 su 3,00

Contrassegna domanda

Due giocatori A e B partecipano a una asta di Wickrey. A e B fanno una offerta maggiore di tutti gli altri partecipanti. A offre 100, mentre B offre 80.
Quali delle seguenti affermazioni sono vere e quali false?

A è il vincitore e paga 100	Falso
B è il vincitore e paga 80	Falso
B è il vincitore e paga 100	Falso
A è il vincitore e paga 80	Vero
Una asta di Wickrey è una asta in busta chiusa	Vero

Domanda 3

Completo

Punteggio ottenuto 3,00 su

Nella **logica epistemica** si assume che per l'operatore **K** (conoscenza) valgano le seguenti proprietà, che possono essere modellate con assiomi:

p1) La conoscenza è non contraddittoria
p2) Ciò che si conosce è vero

Domanda 3

Completo

Punteggio ottenuto 3,00 su 3,00

Contrassegna domanda

Due giocatori A e B partecipano a una asta di Wickrey. A e B fanno una offerta maggiore di tutti gli altri partecipanti. A offre 100, mentre B offre 80.
Quali delle seguenti affermazioni sono vere e quali false?

A è il vincitore e paga 100	Falso
Una asta di Wickrey è una asta in busta chiusa	Vero
A è il vincitore e paga 80	Vero
B è il vincitore e paga 80	Falso
B è il vincitore e paga 100	Falso

Domanda 4

Completo

Punteggio ottenuto 3,00 su 3,00

Contrassegna domanda

Nella **logica epistemica** si assume che per l'operatore **K** (conoscenza) valgano le seguenti proprietà, che possono essere modellate con assiomi:

p1) La conoscenza è non contraddittoria
p2) Ciò che si conosce è vero
p3) Introspezione positiva
p4) Introspezione negativa

Dire quali delle seguenti affermazioni sono vere e quali false.

L'assioma $K\varphi \Rightarrow \neg K \neg \varphi$ modella la proprietà p4	Falso
L'assioma $K\phi \Rightarrow KK\phi$ modella la proprietà p3	Vero
L'assioma $\neg K\phi \Rightarrow K(\neg K\phi)$ modella la proprietà p1	Falso
L'assioma $K\varphi \Rightarrow \varphi$ modella la proprietà p2	Vero

```
(p1)
+!start(X)
   :   true
   <-    ?a(X); ?b(Y); .print("X: ", X, " Y: ", Y).
(p2)
+!start(X)
   :   a(X) & b(Y) & X = Y
   <-    .print("X: ", X, " Y: ", Y).
(p3)
+!start(3)
   :   a(X) & b(Y) & Y = X
   <-   .print("X: ", X, " Y: ", Y).
(p4)
+!start(X, Y)
   :   a(X) & b(Y) & Y = X
   <-   .print("X: ", X, " Y: ", Y).
```

Dire se le seguenti affermazioni sono vere o false.

Il goal !start(X) ha come piani rilevanti solo p1 e p2.	Vero
Il goal !start(3) ha come piani rilevanti p3.	Vero
Il goal !start(3, 3) ha un solo piano rilevante e un solo piano applicabile.	Vero
Il goal !start(3) può restituire come output X: 3 Y:2.	Falso
Il goal !start(X) ha come piani applicabili p1, p2, p3.	Falso

Domanda **3**

Completo

Due giocatori A e B partecipano a una asta di Wickrey. A e B fanno una offerta maggiore di tutti gli altri partecipanti. A offre 100, mentre B offre 80.

To exit full screen, swipe down from top of screen or press F11

Domanda 2

Completo

Punteggio ottenuto 1,20 su 3,00

Contrassegna domanda

Si consideri il seguente programma Jason:

```
// Agent Test in project TESTJason.mas2j

/* Initial beliefs and rules */

a(1).
a(2).
a(3).

b(2).
b(3).
b(4).

/* Initial goals */

....

/* Plans */

(p1)
+!start(X)
   :   true
   <-    ?a(X); ?b(Y); .print("X: ", X, " Y: ", Y).

(p2)
+!start(X)
   :   a(X) & b(Y) & X = Y
   <-    .print("X: ", X, " Y: ", Y).

(p3)
+!start(3)
   :   a(X) & b(Y) & Y = X
   <-   .print("X: ", X, " Y: ", Y).

(p4)
+!start(X, Y)
   :   a(X) & b(Y) & Y = X
```

Fine revisione

Tempo impiegato	15 min. 7 secondi
Valutazione	**9,60** su un massimo di 12,00 (**80%**)

Domanda 1

Completo

Punteggio ottenuto 2,40 su 3,00

Contrassegna domanda

Dire se le seguenti affermazioni sono vere o false.

Se un agente ha l'intezione **a**, ci aspettiamo che l'agente adotti una intenzione **b** tale che **a** e **b** siano mutuamente esclusive.	Falso
Gli agenti credono che le proprie intenzioni siano possibili.	Vero
Le intenzioni non sono chiuse rispetto l'implicazione.	Vero
Le intenzioni sono persistenti.	Vero
L'intenzione è lo stato di cose che un agente desidera.	Vero

Domanda 2

Completo

Punteggio ottenuto 1,20 su 3,00

Contrassegna domanda

Si consideri il seguente programma Jason:

```
// Agent Test in project TESTJason.mas2j

/* Initial beliefs and rules */

a(1).
a(2).
a(3).

b(2).
b(3).
b(4).

/* Initial goals */

....
```

2) stato sociale C(investor, bank,T,invest)

3) stato sociale vuoto

4) stato sociale vuoto

5) stato sociale vuoto

un run di successo: 1, 3, 4, 5

Domanda **4**

Completo

Punteggio ottenuto 4,00 su 4,50

Contrassegna domanda

Cosa si intende per **strategia pura** e **strategia mista** nella teoria dei giochi.

una strategia pura è una sequenza di mosse adottata da un giocatore in un gioco.

una strategia mista per un giocatore è una distribuzione di probabilità sull'insieme di strategie pure che il giocatore ha a disposizione.

Inoltre una strategia pura può essere vista come una strategia mista avente come probabilità 1

Fine revisione

Servizio realizzato presso il Dipartimento di Informatica - Servizi ICT - Page Served By Apollo1
W3C EasyReading Font

Contrassegna domanda

sign_contract **means** create(C(investor, bank, countersign_contract, invest))

counter_sign **means** none

invest **menas** none

notify **means** none

withdraw **means**

release(C(bank, investor, sign_contract ^ invest, countersign_contract ^ notify_inv))

cancel(C(investor, bank, issue_contract, sign_contract))

release(C(bank, investor, accept_inv, issue_contract))

^ rappresenta "and".

Si scriva lo stato sociale (insieme di commitment) validi dopo l'esecuzione delle seguenti sequenze di azioni nell'ipotesi che lo stato sociale iniziale sia vuoto:

1) propose_inv ; accept_inv ; issue_contract ; withdraw

2) propose_inv ; accept_inv ; issue_contract ; sign_contract ; countersign_contract

3) propose_inv ; accept_inv ; issue_contract ; sign_contract ; invest ; notify

4) propose_inv ; accept_inv ; issue_contract ; sign_contract ; invest ; notify ; countersign_contract

5) propose_inv ; accept_inv ; withdraw

Quali delle sequenze è un run (di successo) del protocollo?

1) stato sociale vuoto

2) stato sociale C(investor, bank,T,invest)

3) stato sociale vuoto

4) stato sociale vuoto

5) stato sociale vuoto

un run di successo: 1, 3, 4, 5

To exit full screen, swipe down from top of screen or press F11

Domanda 3

Completo

Punteggio ottenuto 1,50 su 4,50

Contrassegna domanda

Si consideri il seguente protocollo di vendita di titoli a commitment:

propose_inv **means** create(C(bank, investor, accept_inv, issue_contract))

accept_inv **means** create(C(investor, bank, issue_contract, sign_contract))

issue_contract **means** create(C(bank, investor, sign_contract ^ invest, countersign_contract ^ notify_inv))

sign_contract **means** create(C(investor, bank, countersign_contract, invest))

counter_sign **means** none

invest **menas** none

notify **means** none

withdraw **means**

release(C(bank, investor, sign_contract ^ invest, countersign_contract ^ notify_inv))

cancel(C(investor, bank, issue_contract, sign_contract))

release(C(bank, investor, accept_inv, issue_contract))

^ rappresenta "and".

Si scriva lo stato sociale (insieme di commitment) validi dopo l'esecuzione delle seguenti sequenze di azioni nell'ipotesi che lo stato sociale iniziale sia vuoto:

1) propose_inv ; accept_inv ; issue_contract ; withdraw

2) propose_inv ; accept_inv ; issue_contract ; sign_contract ; countersign_contract

3) propose_inv ; accept_inv ; issue_contract ; sign_contract ; invest ; notify

4) propose_inv ; accept_inv ; issue_contract ; sign_contract ; invest ; notify ; countersign_contract

5) propose_inv ; accept_inv ; withdraw

Quali delle sequenze è un run (di successo) del protocollo?

1) stato sociale vuoto

2) stato sociale C(investor, bank,T,invest)

3) stato sociale vuoto

Il programma è compilato con il comando:

javac -cp /..../JADE-all-4.5.0/JADE-bin-4.5.0/jade/lib/jade.jar:. -d classes HelloWorldAgent4.java

ed eseguito con il comando:

java -cp /..../JADE-all-4.5.0/JADE-bin-4.5.0/jade/lib/jade.jar:classes jade.Boot -agents "peter:HelloWorldAgent4"

Riportare l'output a console per l'esecuzione.

```
Hello World. I'am an agent!
My local-name is 1
Operation W
Completed execution MySimpleThreeStepBehaviour, step: 1.
Operation X
Completed execution MyThreeStepBehaviour, step: 1.
Operation Y
Completed execution MyThreeStepBehaviour, step: 2.
Operation Z
Completed execution MyThreeStepBehaviour, step: 3.
Agent HelloWorldAgent4 termination.
```

Domanda 3

Si consideri il seguente protocollo di vendita di titoli a commitment:

```
-----------------------------------------------------------------------
import jade.core.behaviours.Behaviour;

public class MySimpleThreeStepBehaviour extends Behaviour {
   private int step = 0;
   public void action() {
      if (step == 1) myAgent.doDelete();
      System.out.println("Operation W");
      step++;
      System.out.println("Completed execution MySimpleThreeStepBehaviour, step: " + step + ".");
   }
   public boolean done() {
      return step == 3;
   }
}
-----------------------------------------------------------------------
import jade.core.behaviours.OneShotBehaviour;

public class MyOneShotBehaviour extends OneShotBehaviour {
   public void action() {
      myAgent.addBehaviour(new MySimpleThreeStepBehaviour());
   }
}
-----------------------------------------------------------------------
import jade.core.Agent;
import java.util.Iterator;

public class HelloWorldAgent4 extends Agent {
   protected void setup () {
      System.out.println("Hello World. I'am an agent!");
      System.out.println("My local-name is " + getAID().getLocalName());
      addBehaviour(new MyOneShotBehaviour());
      addBehaviour(new MyThreeStepBehaviour());
   }
   protected void takeDown() {
```

Domanda 2

Completo

Punteggio ottenuto 3,00 su 4,50

Contrassegna domanda

Si consideri le seguenti classi Java per JADE:

```
import jade.core.behaviours.Behaviour;

public class MyThreeStepBehaviour extends Behaviour {
   private int step = 0;
   public void action() {
      switch (step) {
      case 0:
         System.out.println("Operation X");
         step++;
         break;
      case 1:
         System.out.println("Operation Y");
         step++;
         break;
      case 2:
         System.out.println("Operation Z");
         step++;
         break;
      }
      System.out.println("Completed execution MyThreeStepBehaviour, step: " + step + ".");
   }
   public boolean done() {
      return step == 3;
   }
}
```

```
import jade.core.behaviours.Behaviour;

public class MySimpleThreeStepBehaviour extends Behaviour {
   private int step = 0;
   public void action() {
      if (step == 1) myAgent.doDelete();
      System.out.println("Operation W");
```

Domanda 1

Completo

Punteggio ottenuto 4,00 su 4,50

Contrassegna domanda

Commentare la seguente formula che definisce un **persistent goal** secondo Cohene Levesque.

$$(\text{P-GOAL } i\ \varphi) \equiv (\text{GOAL } i\ (\text{LATER } \varphi)) \wedge (\text{BEL } i\ \neg\varphi) \wedge [\text{BEFORE } ((\text{BEL } i\ \varphi) \vee (\text{BEL } i\ \mathbf{G}\ \neg\varphi)) \neg(\text{GOAL } i\ (\text{LATER } \varphi))]$$

un persistent goal è un achievement goal, ma prima di abbandonare il goal:

- l'agente crede che il goal sia soddisfatto, o
- l'agente crede che il goal non sarà mai soddisfatto

In particolare se phi è un achievement goal (prima riga), prima che venga abbandonato devono essere verificate 1 delle 2 condizioni sopra

Possiamo dire che un persistent goal è ciò che possibilmente sarà vero e l'agente crede che phi non è vera in questo momento.

Domanda 2

Completo

Punteggio ottenuto 3,00 su 4,50

Contrassegna domanda

Si consideri le seguenti classi Java per JADE:

```
import jade.core.behaviours.Behaviour;

public class MyThreeStepBehaviour extends Behaviour {
    private int step = 0;
    public void action() {
        switch (step) {
        case 0:
            System.out.println("Operation X");
            step++;
```

Domanda 1

Risposta non ancora data

Punteggio max.: 3,00

Contrassegna domanda

Si consideri il seguente frammento di codice in AgentSpeak(L):

```
a(1).
a(2).
a(3).
a(4).
a(5).

+!start1 : a(X) & .print("Start1 X: ", X) & X=3 <- .print("Start1 (body) X: ", X) .

+!start2 : true <- ?a(Y); .print("Start2 (body) Y: ", Y) .

+!start3 : a(X) & .print("Start3 X: ", X) & X=3 <- ?a(Y); .print("Start3 (body) Y: ", Y); Y=3 .
```

Dire se le seguenti affermazioni sono vere o false.

L'evento +!start3 ha come output:

[test] Start3 X: 1
[test] Start3 X: 2
[test] Start3 X: 3
[test] Start3 (body) Y: 1
[test] No failure

Scegli...

L'evento +!start2 ha come output:

[test] Start2 (body) Y: 3

Scegli...

L'evento +!start2 ha come output:

[test] Start2 (body) Y: 1

Scegli...

L'evento +!start1 ha come output:

L'evento +!start1 ha come output:

[test] Start1 X: 1
[test] Start1 X: 2
[test] Start1 X: 3
[test] Start1 (body) X: 3

Scegli...

L'evento +!start1 ha come output:

[test] Start1 X: 1
[test] Start1 X: 2
[test] Start1 X: 3
[test] Start1 (body) X: 1
[test] Start1 (body) X: 2
[test]Start1 (body) X: 3

Scegli...

L'evento +!start3 ha come output:

[test] Start3 X: 3
[test] Start3 (body) Y: 3

Scegli...

Domanda 2

Risposta non ancora data

Punteggio max.: 3,00

Contrassegna domanda

Per modellare con la logica modale diverse proprietà degli agenti può essere necessario modificare la logica aggiungendo degli **assiomi** o, in alternativa, restringere la classe dei modelli formulando delle **proprietà** dei frame. In diversi casi la formulazione con assiomi **corrisponde** alle proprietà dei frame.

Dati i seguenti assiomi e proprietà:

(a1) $[]\varphi \Rightarrow \varphi$

(a2) $[]\varphi \Rightarrow <>\varphi$

(p1) $\forall w \in W. \exists w' \in W. (w, w') \in R$

(p2) $\forall w \in W. (w, w) \in R$

dire se le seguenti affermazioni sono vere o false.

L'assioma (a2) corrisponde alla proprietà (p1) Scegli...

L'assioma (a1) corrisponde alla proprietà (p1) Scegli...

La proprietà (p2) modella la riflessività Scegli...

L'assioma (a1) corrisponde alla proprietà (p2) Scegli...

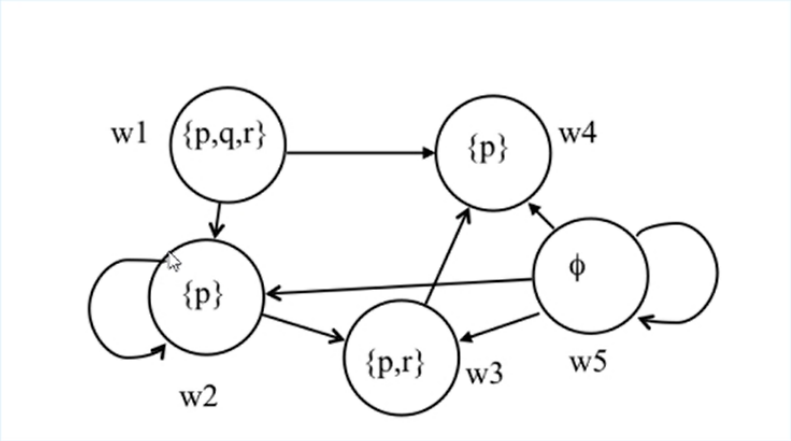
Domanda 3

Risposta non ancora data

Punteggio max.: 3,00

Contrassegna domanda

Si consideri il seguente modello.



Dire se le seguenti affermazioni sono vere o false.

[ ]p è vero nel mondo w2 Scegli...

[ ]p è vero nel mondo w5 Scegli...

< >r è vero nel mondo w4 Scegli...

< >r è vero nel mondo w5 Scegli...

Domanda 4
Risposta non ancora data
Punteggio max.: 3,00
Contrassegna domanda

Si consideri lo scheduling dei comportamenti di un agente Jade. Dire se le seguenti affermazioni sono vere o false.

Il metodo action() è eseguito parzialmente sino ad una condizione di sincronizzazione con altro comportamento presente pool dei comportamenti attivi dell'agente	Scegli...
Il metodo action() è eseguito completamente ogni volta che il comportamento è selezionato dal pool dei comportamenti attivi dell'agente	Scegli...
La funzione di setup() è invocata una sola volta	Scegli...
Il valore restituito dal metodo done() determina se un comportamento deve essere o no inserito nel pool dei comportamenti attivi dell'agente	Scegli...
Un comportamento risiede nel pool dei comportamenti attivi fino all'esecuzione del metodo takeDown()	Scegli...
Un comportamento ciclico non valuta mai il risultato dell'esecuzione del metodo done()	Scegli...

Domanda 1
Risposta non ancora data
Punteggio max.: 4,50
Contrassegna domanda

Si consideri il seguente protocollo a commitment:

1. cfp means create(C(i, p, propose, accept v reject))
2. accept means none
3. reject means release(C(p, i, accept, done v failure))
4. propose means create(C(p, i, accept, done v failure))
5. refuse means release(C(i, p, propose, accept v reject))
6. done means none
7. failure means none

Scrivere l'insieme di commitment presenti nello stato sociale dopo l'esecuzione delle sequenze di azioni tenendo conto che lo stato sociale iniziale è vuoto:

1. cfp; propose.
2. propose; accept.
3. cfp; refuse.

Domanda 2
Risposta non ancora data
Punteggio max.: 4,50
Contrassegna domanda

Data la seguente matrice di payoff, spiegare in cosa consiste il dilemma del prigioniero.

	Alice: testify	Alice: refuse
Bob: testify	A = -5, B = -5	A = -10, B = 0
Bob: refuse	A = 0, B = -10	A = -1, B = -1

Domanda 3
Risposta non ancora data

Logica modale: definire l'assioma K e la regola di necessitation.

Domanda 4
Risposta non ancora data
Punteggio max.: 4,50

Scrivere l'Agent Control Loop per l'architettura BDI nella versione senza Commitment Strategy (Versione 3), spiegando la notazione utilizzata.

Domanda 1

Completo

Punteggio ottenuto 0,00 su 3,00

Contrassegna domanda

Il GOLOG è un linguaggio di programmazione che fa uso del situation calculus implementato con la logica classica. Spiegare il significato dei "Successor state axioms" facendo riferimento al seguente esempio:

$$Poss(a,s) \Rightarrow [broken(x,do(a,s)) \equiv \exists r\{a = drop(r,x) \wedge fragile(x,s)\} \vee broken(x,s) \wedge \neg \exists r\{a = repair(r,x)\}]$$

le azioni in GOLOG, presentato nel 1994, sono specificate dando delle precondizioni e degli effetti.

I Successor state axioms vengono usati per esplicitare gli effetti delle azioni.

Commento:

Domanda 2

Completo

Punteggio ottenuto 1,50 su 2,00

Contrassegna domanda

Nella logica temporale Branching-Time il tempo è modellato con un albero infinito su cui si possono formulare *path formulas* o *state formulas*. Dire se le formule **A**π e **E**π sono *path* o *state formulas* e spiegare il loro significato.

Le state formula sono le seguenti:
1)p
2) alfa OR Beta

3) not alfa

4) A(pi)

5)E(pi)

Dove alfa e' una state formula mentre pi si riferisce alle path formula.

Le path formula sono invece:

1) alfa

2)pi OR p

3) not pi

4) Xpi

5)pi U p

Dove alfa e' uno state formula mentre pi e p sono path formula

A(pi) significa "per tutti i futuri", ovvero viene usato per controllare tutti i percorsi (partendo da uno stato iniziale)

E(pi) significa "per qualche futuro", ovvero viene usato per controllare solo alcuni percorsi (partendo da uno stato iniziale).

A(pi) e E(pi) sono duali.

Commento:

Domanda 3

Completo

Punteggio ottenuto 2,00 su 2,00

Contrassegna domanda

Commentare la formula che definisce un **persistent goal** secondo Cohen e Levesque.

$$(\text{P-GOAL } i\ \varphi) \equiv (\text{GOAL } i\ (\text{LATER } \varphi)) \wedge (\text{BEL } i\ \neg\varphi) \wedge [\text{BEFORE } ((\text{BEL } i\ \varphi) \vee (\text{BEL } i\ \mathbf{G}\ \neg\varphi)) \neg(\text{GOAL } i\ (\text{LATER } \varphi))]$$

un Goal persistente e' un achievement goal che puo' essere rimosso solo per due motivi: o l'agente lo ha raggiunto, oppure si e' reso conto che non e' mai raggiungibile.

Piu' formalmente possiamo dire: Ha un goal teta che possibilmente diventera' vero e crede che teta non e' attualmente vera.

Prima che il goal vada in "drop" una delle seguenti condizioni deve essere vera:

1) l'agente crede che il goal sia soddisfatto.

2) l'agente crede che il goal non sara' mai soddisfatto.

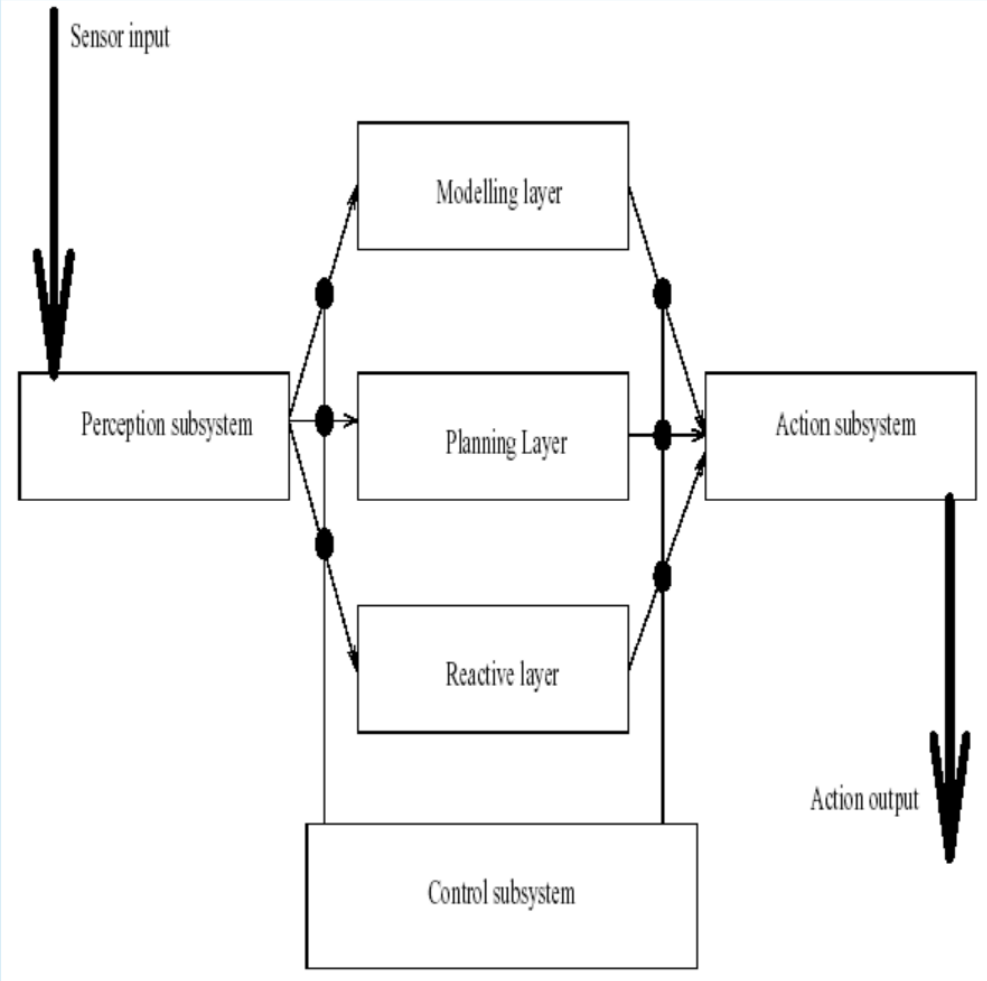
Domanda 4

Completo

Punteggio ottenuto 1,50 su 2,00

Contrassegna domanda

Descrivere i componenti della architettura **TouringMachines** rappresentata nella figura seguente. In particolare, dire se si tratta di una architettura orizzontale o verticale, e spiegare il significato del "control subsystem".



La Touring Machine e' una architettura a stratificazione orizzontale consistente di una struttura con sottosistemi di percezione e di azione che si interfacciano direttamente con l'ambiente e con tre livelli di controllo racchiusi in un framework di controllo che fa da mediatore tra i livelli.

Il reactive layer e' implementato come un set di regole di situazioni-azioni.

Il planning layer costruisce piani e seleziona azioni da eseguire in ordine per completare i goal dell'agente.

Il modelling layer contiene una rappresenazione simbolica del "cognitive state" delle altre entita' dell'ambiente dell'agente.

Tutti e tre i livelli comunicano tra loro e sono racchiusi in un framework di controllo (Control subsystem) che usa regole di controllo.

Domanda 5

Parzialmente corretta

Punteggio ottenuto 2,00 su 4,00

Contrassegna domanda

La seguente matrice di payoff si riferisce al "game of chicken" in cui i due giocatori **I** e **J** guidano ciascuno la propria auto verso un precipizio. Ogni giocatore può sterzare o andare diritto.

	I : diritto	I : sterza
J : diritto	I = 0, J = 0	I = 1, J = 3
J : sterza	I = 3, J = 1	I = 2, J = 2

Dire quali delle seguenti affermazioni sono vere.

Scegli una o più alternative:

- a. [I : diritto, J : diritto] è un *equilibrio di Nash*.
- b. [I : diritto, J : sterza] è un *equilibrio di Nash*.
- c. "sterza" è una *strategia dominante* per I.
- d. [I : sterza, J : sterza] è *Pareto optimal*. ✓

Risposta parzialmente esatta.

Hai selezionato correttamente 1.
La risposta corretta è: [I : diritto, J : sterza] è un *equilibrio di Nash.*, [I : sterza, J : sterza] è *Pareto optimal*.

Domanda 6

Parzialmente corretta

Punteggio ottenuto 2,00 su 3,00

Contrassegna domanda

Per modellare con la logica modale diverse proprietà degli agenti può essere necessario modificare la logica aggiungendo degli **assiomi** o, in alternativa, restringere la classe dei modelli formulando delle **proprietà** dei frame. In diversi casi la formulazione con assiomi **corrisponde** alle proprietà dei frame.

Dati i seguenti assiomi e proprietà:

(a1) $[\,]\varphi \Rightarrow \varphi$

(a2) $[\,]\varphi \Rightarrow \Diamond\varphi$

(p1) $\forall w \in W \;.\; \exists w' \in W \;.\; (w,w') \in R$

(p2) $\forall w \in W \;.\; (w,w) \in R$

dire quali delle seguenti affermazioni sono vere.

Scegli una o più alternative:

- [ ] a.
  L'assioma (a2) corrisponde alla proprietà (p2).
- [x] b.
  L'assioma (a1) corrisponde alla proprietà (p2). ✓
- [ ] c.
  L'assioma (a1) corrisponde alla proprietà (p1).
- [ ] d.
  La proprietà (p2) modella la serialità.
- [x] e.
  L'assioma (a2) corrisponde alla proprietà (p1). ✓
- [ ] f.
  La proprietà (p2) modella la riflessività.

Risposta parzialmente esatta.

Hai selezionato correttamente 2.
La risposta corretta è:
La proprietà (p2) modella la riflessività.,

L'assioma (a1) corrisponde alla proprietà (p2).,

L'assioma (a2) corrisponde alla proprietà (p1).

Domanda 8

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

Contrassegna domanda

Riordinare le seguenti istruzioni di un Agent Control Loop.

1. B := B0; I := I0; ✓
2. while true do ✓
3. get next percept ✓
4. B := brf(B, ρ) ✓
5. D := options(B, I) ✓
6. I := filter(B, D, I) ✓
7. π := plan(B, I) ✓
8. execute(π) ✓
9. end while ✓

Risposta corretta.

La risposta corretta è: 1. – B := B0; I := I0;, 2. – while true do, 3. – get next percept, 4. – B := brf(B, ρ), 5. – D := options(B, I), 6. – I := filter(B, D, I), 7. – π := plan(B, I), 8. – execute(π), 9. – end while

Domanda 9

Parzialmente corretta

Punteggio ottenuto 0,33 su 2,00

Contrassegna domanda

Si consideri lo scheduling dei comportamenti di un agente Jade. Quali delle seguenti affermazioni sono vere?

Scegli una o più alternative:

- [x] a. La funzione di setup() è invocata una sola volta. ✓
- [ ] b. Il metodo action() è eseguito completamente ogni volta che il comportamento è selezionato dal pool dei comportamenti attivi dell'agente.
- [ ] c. Un comportamento ha tanti metodi action() quanti sono i task indipendenti che tale comportamento deve raggiungere.
- [ ] d. Il metodo action() è eseguito parzialmente sino ad una condizione di sincronizzazione con altro comportamento presente pool dei comportamenti attivi dell'agente.
- [x] e. Un comportamento ciclico non valuta mai il risultato dell'esecuzione del metodo done(). ✗
- [x] f. Un comportamento risiede nel pool dei comportamenti attivi fino all'esecuzione del metodo takeDown(). ✗
- [x] g. Il valore restituito dal metodo done() determina se un comportamento deve o no essere inserito nel pool dei comportamenti attivi dell'agente. ✓

Risposta parzialmente esatta.

Hai selezionato troppe opzioni.
La risposta corretta è: La funzione di setup() è invocata una sola volta., Il metodo action() è eseguito completamente ogni volta che il comportamento è selezionato dal pool dei comportamenti attivi dell'agente., Il valore restituito dal metodo done() determina se un comportamento deve o no essere inserito nel pool dei comportamenti attivi dell'agente.

Domanda 10

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

Contrassegna domanda

Si consideri il seguente frammento di codice Jade:

```
public class MyThreeStepBehaviour extends Behaviour {
    private int step = 0;
    public void action() {
        // istruzione1
        switch (step) {
        case 0: // perform operation X
            step++;
            break;
        case 1: // perform operation Y
            step++;
            break;
        case 2: // perform operation Z
            step++;
            break;
        }
        // istruzione2
    }
    public boolean done() {
        return step == 3;
    }
}
```

Quali delle seguenti affermazioni sono vere?

Scegli una o più alternative:

- a. istruizione1 viene eseguita 1 volta.
- b. istruzione1 e istruzione2 vengono eseguite 3 volte. ✓
- c. istruzione2 non viene mai eseguita.
- d. istruizone1 e istruzione2 vengono eseguite 1 volta.
- e. istruzione1 non viene mai eseguita.

Risposta corretta.

La risposta corretta è: istruzione1 e istruzione2 vengono eseguite 3 volte.

Domanda 11

Risposta non data

Punteggio max.: 3,00

Contrassegna domanda

Si consideri il seguente frammento di codice Jason:

```
a(1).
a(2).
b(2).
h(X) : a(X) & b(Y) & X = Y <- body1. (p1)
h(3) : a(X) & b(Y) & X = Y <- body2, (p2)
h(X) : a(X) & b(Y) & X = Y <- ?b(1), body3. (p3)
```

Quali sono le affermazioni vere?

Scegli una o più alternative:

- a. (p1) è il solo piano rilevante per l'evento +!h(2).
- b. (p1) e (p3) sono piani rilevanti per l'evento +!h(2).
- c. (p1) è il solo piano applicabile per l'evento +!h(2).
- d. (p1) e (p3) sono piani rilevanti per l'evento +!h(1).
- e. Nessun piano rilevante per l'evento +!h(3).
- f. Nessun piano applicabile per l'evento +!h(3).
- g. (p2) è il solo piano rilevante per l'evento +!h(3).
- h. (p1) e (p2) sono piani rilevanti per l'evento +!h(2).
- i. (p1) e (p3) sono piani applicabili per l'evento +!h(2).
- j. (p1), (p2) e (p3) sono piani rilevanti per l'evento +!h(1).
- k. Non esiste un piano applicabile per l'evento +!h(1).
- l. (p1) è un piano rilevante per l'evento +!h(1).
- m. (p2) è il solo applicabile per l'evento +!h(3).
- n. (p1) è un piano applicabile per l'evento +!h(1).

Risposta errata.

La risposta corretta è: (p1) e (p3) sono piani rilevanti per l'evento +!h(1)., Non esiste un piano applicabile per l'evento +!h(1)., (p1) e (p3) sono piani rilevanti per l'evento +!h(2)., (p1) e (p3) sono piani applicabili per l'evento +!h(2)., (p2) è il solo piano rilevante per l'evento +!h(3)., (p2) è il solo applicabile per l'evento +!h(3).

Domanda 12

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

Contrassegna domanda

Si consideri la definizione di practical reasoning, quali delle seguenti affermazioni sono vere?

Scegli una o più alternative:

- [x] a. La fase di means-ends reasoning consiste nel decidere come uno stato di interesse deve essere raggiunto. ✓
- [x] b. La fase di deliberazione consiste nel decidere quale stato di interesse deve essere raggiunto. ✓
- [ ] c. La fase di deliberazione consiste nel decidere come uno stato di interesse deve essere raggiunto.
- [x] d. La fase di deliberazione precede la fase di means-ends reasoning. ✓
- [ ] e. La fase di means-ends reasoning consiste nel decidere quale stato di interesse deve essere raggiunto.

Risposta corretta.

La risposta corretta è: La fase di deliberazione consiste nel decidere quale stato di interesse deve essere raggiunto., La fase di means-ends reasoning consiste nel decidere come uno stato di interesse deve essere raggiunto., La fase di deliberazione precede la fase di means-ends reasoning.

Domanda 13

Parzialmente corretta

Punteggio ottenuto 1,33 su 2,00

Contrassegna domanda

Si consideri il seguente protocollo a commitment:

cfp means create(C(i, p, propose, accept v reject))

accept means none

reject means release(C(p, i, accept, done v failure))

propose means create(C(p, i, accept, done v failure))

refuse means release(C(i, p, propose, accept v reject))

done means none

failure means none

Quali delle seguenti affermazioni sono vere?

Scegli una o più alternative:

- [ ] a. Lo stato sociale vuoto è raggiunto dopo l'esecuzione della sequenza cfp e reject.
- [ ] b. Non è possibile che si verifichi la sequenza cfp e accept.
- [x] c. Lo stato sociale è {C(i, p, T, accept v reject), C(p, i, accept, done v failure)} dopo l'esecuzione della sequenza cfp e propose. ✓
- [ ] d. Lo stato sociale è {C(i, p, T, accept v reject), C(p, i, accept, done v failure)} dopo l'esecuzione dell'azione propose.
- [x] e. Lo stato sociale vuoto è raggiunto dopo l'esecuzione della sequenza cfp, propose, accept, done. ✓

Risposta parzialmente esatta.

Hai selezionato correttamente 2.

La risposta corretta è: Lo stato sociale è {C(i, p, T, accept v reject), C(p, i, accept, done v failure)} dopo l'esecuzione della sequenza cfp e propose., Lo stato sociale vuoto è raggiunto dopo l'esecuzione della sequenza cfp e reject., Lo stato sociale vuoto è raggiunto dopo l'esecuzione della sequenza cfp, propose, accept, done.

Domanda 14

Risposta corretta

Punteggio ottenuto 1,00 su 1,00

Contrassegna domanda

L'obiettivo del Knowledge Sharing Effort è di sviluppare tecniche, metodologie e strumenti software per la condivisione e il riuso di conoscenza.

Scegli una risposta:

- Vero ✓
- Falso

La risposta corretta è 'Vero'.

Domanda 11

Risposta non data

Punteggio max.: 3,00

Contrassegna domanda

Si consideri il seguente frammento di codice Jason:

```
a(1).
a(2).
b(2).
h(X) : a(X) & b(Y) & X = Y <- body1. (p1)
h(3) : a(X) & b(Y) & X = Y <- body2, (p2)
h(X) : a(X) & b(Y) & X = Y <- ?b(1), body3. (p3)
```

Quali sono le affermazioni vere?

Scegli una o più alternative:

- a. (p1) è il solo piano rilevante per l'evento +!h(2).
- b. (p1) e (p3) sono piani rilevanti per l'evento +!h(2).
- c. (p1) è il solo piano applicabile per l'evento +!h(2).
- d. (p1) e (p3) sono piani rilevanti per l'evento +!h(1).
- e. Nessun piano rilevante per l'evento +!h(3).
- f. Nessun piano applicabile per l'evento +!h(3).
- g. (p2) è il solo piano rilevante per l'evento +!h(3).
- h. (p1) e (p2) sono piani rilevanti per l'evento +!h(2).
- i. (p1) e (p3) sono piani applicabili per l'evento +!h(2).
- j. (p1), (p2) e (p3) sono piani rilevanti per l'evento +!h(1).
- k. Non esiste un piano applicabile per l'evento +!h(1).
- l. (p1) è un piano rilevante per l'evento +!h(1).
- m. (p2) è il solo applicabile per l'evento +!h(3).
- n. (p1) è un piano applicabile per l'evento +!h(1).

Risposta errata.

La risposta corretta è: (p1) e (p3) sono piani rilevanti per l'evento +!h(1)., Non esiste un piano applicabile per l'evento +!h(1)., (p1) e (p3) sono piani rilevanti per l'evento +!h(2)., (p1) e (p3) sono piani applicabili per l'evento +!h(2)., (p2) è il solo piano rilevante per l'evento +!h(3)., (p2) è il solo applicabile per l'evento +!h(3).

Domanda 10

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

Contrassegna domanda

Si consideri il seguente frammento di codice Jade:

```
public class MyThreeStepBehaviour extends Behaviour {
   private int step = 0;
   public void action() {
      // istruzione1
      switch (step) {
      case 0: // perform operation X
         step++;
         break;
      case 1: // perform operation Y
         step++;
         break;
      case 2: // perform operation Z
         step++;
         break;
      }
      // istruzione2
   }
   public boolean done() {
      return step == 3;
   }
}
```

Quali delle seguenti affermazioni sono vere?

Scegli una o più alternative:

- [ ] a. istruizione1 viene eseguita 1 volta.
- [x] b. istruzione1 e istruzione2 vengono eseguite 3 volte. ✓
- [ ] c. istruzione2 non viene mai eseguita.
- [ ] d. istruizone1 e istruzione2 vengono eseguite 1 volta.
- [ ] e. istruzione1 non viene mai eseguita.

Risposta corretta.

La risposta corretta è: istruzione1 e istruzione2 vengono eseguite 3 volte.

Domanda 9

Parzialmente corretta

Punteggio ottenuto 0,33 su 2,00

Contrassegna domanda

Si consideri lo scheduling dei comportamenti di un agente Jade. Quali delle seguenti affermazioni sono vere?

Scegli una o più alternative:

- [x] a. La funzione di setup() è invocata una sola volta. ✓
- [ ] b. Il metodo action() è eseguito completamente ogni volta che il comportamento è selezionato dal pool dei comportamenti attivi dell'agente.
- [ ] c. Un comportamento ha tanti metodi action() quanti sono i task indipendenti che tale comportamento deve raggiungere.
- [ ] d. Il metodo action() è eseguito parzialmente sino ad una condizione di sincronizzazione con altro comportamento presente pool dei comportamenti attivi dell'agente.
- [x] e. Un comportamento ciclico non valuta mai il risultato dell'esecuzione del metodo done(). ✗
- [x] f. Un comportamento risiede nel pool dei comportamenti attivi fino all'esecuzione del metodo takeDown(). ✗
- [x] g. Il valore restituito dal metodo done() determina se un comportamento deve o no essere inserito nel pool dei comportamenti attivi dell'agente. ✓

Risposta parzialmente esatta.

Hai selezionato troppe opzioni.
La risposta corretta è: La funzione di setup() è invocata una sola volta., Il metodo action() è eseguito completamente ogni volta che il comportamento è selezionato dal pool dei comportamenti attivi dell'agente., Il valore restituito dal metodo done() determina se un comportamento deve o no essere inserito nel pool dei comportamenti attivi dell'agente.

Domanda 8

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

Contrassegna domanda

Riordinare le seguenti istruzioni di un Agent Control Loop.

1. B := B0; I := I0; ✓
2. while true do ✓
3. get next percept ✓
4. B := brf(B, ρ) ✓
5. D := options(B, I) ✓
6. I := filter(B, D, I) ✓
7. π := plan(B, I) ✓
8. execute(π) ✓
9. end while ✓

Risposta corretta.

La risposta corretta è: 1. – B := B0; I := I0;, 2. – while true do, 3. – get next percept, 4. – B := brf(B, ρ), 5. – D := options(B, I), 6. – I := filter(B, D, I), 7. – π := plan(B, I), 8. – execute(π), 9. – end while

Domanda 6

Parzialmente corretta

Punteggio ottenuto 2,00 su 3,00

Contrassegna domanda

Per modellare con la logica modale diverse proprietà degli agenti può essere necessario modificare la logica aggiungendo degli **assiomi** o, in alternativa, restringere la classe dei modelli formulando delle **proprietà** dei frame. In diversi casi la formulazione con assiomi **corrisponde** alle proprietà dei frame.

Dati i seguenti assiomi e proprietà:

(a1) $[\,]\varphi \Rightarrow \varphi$

(a2) $[\,]\varphi \Rightarrow \Diamond\varphi$

(p1) $\forall w \in W \;.\; \exists w' \in W \;.\; (w,w') \in R$

(p2) $\forall w \in W \;.\; (w,w) \in R$

dire quali delle seguenti affermazioni sono vere.

Scegli una o più alternative:

- [ ] a.
  L'assioma (a2) corrisponde alla proprietà (p2).
- [x] b.
  L'assioma (a1) corrisponde alla proprietà (p2). ✓
- [ ] c.
  L'assioma (a1) corrisponde alla proprietà (p1).
- [ ] d.
  La proprietà (p2) modella la serialità.
- [x] e.
  L'assioma (a2) corrisponde alla proprietà (p1). ✓
- [ ] f.
  La proprietà (p2) modella la riflessività.

Risposta parzialmente esatta.

Hai selezionato correttamente 2.
La risposta corretta è:
La proprietà (p2) modella la riflessività.,

L'assioma (a1) corrisponde alla proprietà (p2).,

L'assioma (a2) corrisponde alla proprietà (p1).

Domanda 5

Parzialmente corretta

Punteggio ottenuto 2,00 su 4,00

Contrassegna domanda

La seguente matrice di payoff si riferisce al "game of chicken" in cui i due giocatori **I** e **J** guidano ciascuno la propria auto verso un precipizio. Ogni giocatore può sterzare o andare diritto.

	I : diritto	I : sterza
J : diritto	I = 0, J = 0	I = 1, J = 3
J : sterza	I = 3, J = 1	I = 2, J = 2

Dire quali delle seguenti affermazioni sono vere.

Scegli una o più alternative:

- a. [I : diritto, J : diritto] è un *equilibrio di Nash*.
- b. [I : diritto, J : sterza] è un *equilibrio di Nash*.
- c. "sterza" è una *strategia dominante* per I.
- d. [I : sterza, J : sterza] è *Pareto optimal*. ✓

Risposta parzialmente esatta.

Hai selezionato correttamente 1.

La risposta corretta è: [I : diritto, J : sterza] è un *equilibrio di Nash.*, [I : sterza, J : sterza] è *Pareto optimal*.

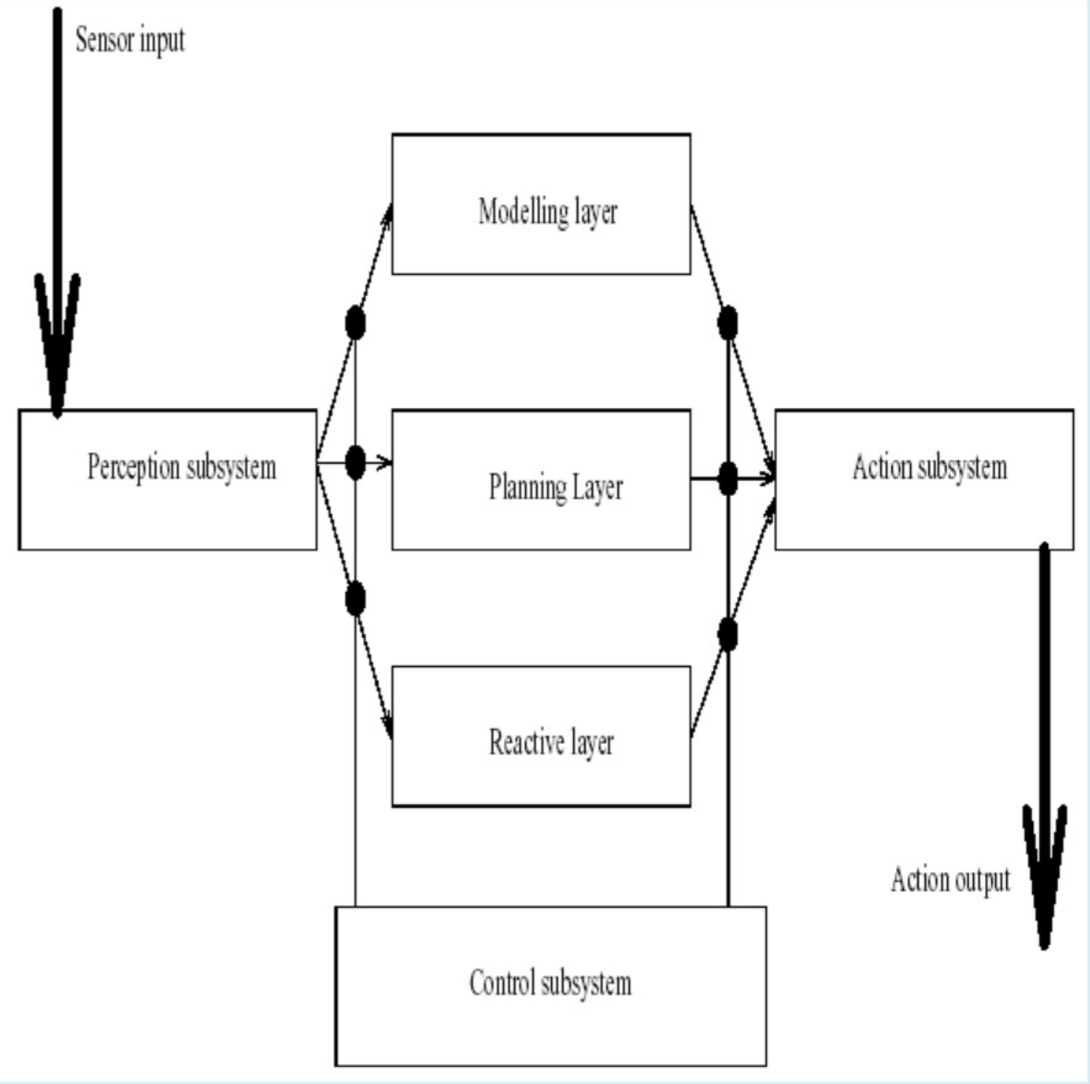
Domanda 4

Completo

Punteggio ottenuto 1,50 su 2,00

Contrassegna domanda

Descrivere i componenti della architettura **TouringMachines** rappresentata nella figura seguente. In particolare, dire se si tratta di una architettura orizzontale o verticale, e spiegare il significato del "control subsystem".



La Touring Machine e' una architettura a stratificazione orizzontale consistente di una struttura con sottosistemi di percezione e di azione che si interfacciano direttamente con l'ambiente e con tre livelli di controllo racchiusi in un framework di controllo che fa da mediatore tra i livelli.

Il reactive layer e' implementato come un set di regole di situazioni-azioni.

Il planning layer costruisce piani e seleziona azioni da eseguire in ordine per completare i goal dell'agente.

Il modelling layer contiene una rappresenazione simbolica del "cognitive state" delle altre entita' dell'ambiente dell'agente.

Tutti e tre i livelli comunicano tra loro e sono racchiusi in un framework di controllo (Control subsystem) che usa regole di controllo.

Domanda 3

Completo

Punteggio ottenuto 2,00 su 2,00

Contrassegna domanda

Commentare la formula che definisce un **persistent goal** secondo Cohen e Levesque.

$$
\begin{aligned}
(\text{P-GOAL } i\ \varphi) \equiv\ & (\text{GOAL } i\ (\text{LATER } \varphi)) \wedge (\text{BEL } i\ \neg\varphi) \wedge \\
& [\text{BEFORE } ((\text{BEL } i\ \varphi) \vee (\text{BEL } i\ \mathbf{G}\ \neg\varphi)) \\
& \neg(\text{GOAL } i\ (\text{LATER } \varphi))]
\end{aligned}
$$

un Goal persistente e' un achievement goal che puo' essere rimosso solo per due motivi: o l'agente lo ha raggiunto, oppure si e' reso conto che non e' mai raggiungibile.

Piu' formalmente possiamo dire: Ha un goal teta che possibilmente diventera' vero e crede che teta non e' attualmente vera.

Prima che il goal vada in "drop" una delle seguenti condizioni deve essere vera:

1) l'agente crede che il goal sia soddisfatto.

2) l'agente crede che il goal non sara' mai soddisfatto.

Domanda 14

Risposta corretta

Punteggio ottenuto 1,00 su 1,00

Contrassegna domanda

L'obiettivo del Knowledge Sharing Effort è di sviluppare tecniche, metodologie e strumenti software per la condivisione e il riuso di conoscenza.

Scegli una risposta:

- Vero ✓
- Falso

La risposta corretta è 'Vero'.

Domanda 12

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

Contrassegna domanda

Si consideri la definizione di practical reasoning, quali delle seguenti affermazioni sono vere?

Scegli una o più alternative:

- [x] a. La fase di means-ends reasoning consiste nel decidere come uno stato di interesse deve essere raggiunto. ✓
- [x] b. La fase di deliberazione consiste nel decidere quale stato di interesse deve essere raggiunto. ✓
- [ ] c. La fase di deliberazione consiste nel decidere come uno stato di interesse deve essere raggiunto.
- [x] d. La fase di deliberazione precede la fase di means-ends reasoning. ✓
- [ ] e. La fase di means-ends reasoning consiste nel decidere quale stato di interesse deve essere raggiunto.

Risposta corretta.

La risposta corretta è: La fase di deliberazione consiste nel decidere quale stato di interesse deve essere raggiunto., La fase di means-ends reasoning consiste nel decidere come uno stato di interesse deve essere raggiunto., La fase di deliberazione precede la fase di means-ends reasoning.

Domanda 13

Parzialmente corretta

Punteggio ottenuto 1,33 su 2,00

Contrassegna domanda

Si consideri il seguente protocollo a commitment:

cfp means create(C(i, p, propose, accept v reject))

accept means none

reject means release(C(p, i, accept, done v failure))

propose means create(C(p, i, accept, done v failure))

refuse means release(C(i, p, propose, accept v reject))

done means none

failure means none

Quali delle seguenti affermazioni sono vere?

Scegli una o più alternative:

- [ ] a. Lo stato sociale vuoto è raggiunto dopo l'esecuzione della sequenza cfp e reject.
- [ ] b. Non è possibile che si verifichi la sequenza cfp e accept.
- [x] c. Lo stato sociale è {C(i, p, T, accept v reject), C(p, i, accept, done v failure)} dopo l'esecuzione della sequenza cfp e propose. ✓
- [ ] d. Lo stato sociale è {C(i, p, T, accept v reject), C(p, i, accept, done v failure)} dopo l'esecuzione dell'azione propose.
- [x] e. Lo stato sociale vuoto è raggiunto dopo l'esecuzione della sequenza cfp, propose, accept, done. ✓

Risposta parzialmente esatta.

Hai selezionato correttamente 2.

La risposta corretta è: Lo stato sociale è {C(i, p, T, accept v reject), C(p, i, accept, done v failure)} dopo l'esecuzione della sequenza cfp e propose., Lo stato sociale vuoto è raggiunto dopo l'esecuzione della sequenza cfp e reject., Lo stato sociale vuoto è raggiunto dopo l'esecuzione della sequenza cfp, propose, accept, done.

Domanda 1

Completo

Punteggio ottenuto 0,00 su 3,00

Contrassegna domanda

Il GOLOG è un linguaggio di programmazione che fa uso del situation calculus implementato con la logica classica. Spiegare il significato dei "Successor state axioms" facendo riferimento al seguente esempio:

$$Poss(a,s) \Rightarrow [broken(x,do(a,s)) \equiv \exists r\{a = drop(r,x) \wedge fragile(x,s)\} \vee broken(x,s) \wedge \neg\exists r\{a = repair(r,x)\}]$$

le azioni in GOLOG, presentato nel 1994, sono specificate dando delle precondizioni e degli effetti.

I Successor state axioms vengono usati per esplicitare gli effetti delle azioni.

Commento:

Domanda 2

Completo

Punteggio ottenuto 1,50 su 2,00

Contrassegna domanda

Nella logica temporale Branching-Time il tempo è modellato con un albero infinito su cui si possono formulare *path formulas* o *state formulas*. Dire se le formule **A**π e **E**π sono *path* o *state formulas* e spiegare il loro significato.

Le state formula sono le seguenti:
1)p
2) alfa OR Beta
3) not alfa
4) A(pi)
5)E(pi)
Dove alfa e' una state formula mentre pi si riferisce alle path formula.

Le path formula sono invece:
1) alfa
2)pi OR p
3) not pi
4) Xpi
5)pi U p
Dove alfa e' uno state formula mentre pi e p sono path formula

A(pi) significa "per tutti i futuri", ovvero viene usato per controllare tutti i percorsi (partendo da uno stato iniziale)
E(pi) significa "per qualche futuro", ovvero viene usato per controllare solo alcuni percorsi (partendo da uno stato iniziale).
A(pi) e E(pi) sono duali.

Commento:

Si consideri il modello di esecuzione dei comportamenti di un agente in JADE. Dire quali delle seguenti affermazioni sono vere.

Scegli una o più alternative:

- [ ] a. La gestione dei compotamenti di un agente è di tipo competitivo.
- [ ] b. Quando un comportamento di un agente è selezionato, il suo metodo action viene completamente eseguito fino al suo return.
- [x] c. Ogni comportamento di un agente definisce un metodo done.
- [x] d. La sincronizzazione tra i comportamenti di un agente è ottenuta mediante le usuali primitive del linguaggio Java.
- [ ] e. La gestione dei comportamenti di un agente è di tipo pre-emptive.

Cohen e Levesque hanno proposto una logica per modellare agenti BDI la cui semantica è basata sui mondi possibili. Spiegare cosa hanno di diverso i modelli di questa logica rispetto ai modelli della logica modale classica.

Si consideri il seguente esempio di programma Concurrent METATEM.

Example. A **'resource producer'** rp:

rp (ask1, ask2) [give1, give2]

1. $\bigcirc$ask1 $\Rightarrow$ $\Diamond$give1
2. $\bigcirc$ask2 $\Rightarrow$ $\Diamond$give2
3. **start** $\Rightarrow$ $\Box\neg$(give1 $\wedge$ give2)

Two **consumers**:

rc1 (give1) [ask1]

4. **start** $\Rightarrow$ ask1
5. $\bigcirc$ask1 $\Rightarrow$ ask1

rc2 (ask1, give2) [ask2]

6. $\bigcirc$(ask1 $\wedge$ $\neg$ask2) $\Rightarrow$ ask2

Commentare brevemente le regole da 1 a 6.

Quali delle seguente affermazioni relative ad AgentSpeak(L) sono vere?

Scegli una o più alternative:

- a. AgentSpeak(L) rappresenta achievement goal in un piano ma non test goal.
- b. AgentSpeak(L) ha una funzione di selezione per determinare i piani rilevanti.
- c. AgentSpeak(L) distingue tra eventi interni ed esterni.
- d. AgentSpeak(L) ha una funzione di selezione per determinare gli eventi rilevanti.
- e. AgentSpeak(L) è implentato in linguaggio Java.
- f. In AgentSpeak(L) sono rappresentati i belief e i desire ma non le intention.
- g. AgentSpeak(L) è un linguaggio dichiarativo.
- h. AgentSpeak(L) ha una funzione di selezione per determinare i belief da adottare.

Si consideri l'attività di *Practical Reasoning*, quali delle seguenti affermazioni sono vere?

Scegli una o più alternative:

- a. L'attività di means-ends reasoning produce come risultato le intezioni dell'agente.
- b. L'attività di deliberation stabilisce come raggiungere uno stato.
- c. L'attività di means-ends reasoning stabilisce quale stato raggiungere.

Domanda **1**

Parzialmente corretta

Punteggio ottenuto 0,25 su 1,50

Contrassegna domanda

Si considerino le intenzioni, quali delle seguenti affermazioni sono vere?

Scegli una o più alternative:

- a. Le intenzioni sono piani, sequenze di azioni che un agente esegue per raggiungere un suo goal.
- b. Le intenzioni sono la sorgente della proattività di un agente. ✓
- c. Le intezioni sono dei vincoli che un agente deve rispettare nella scelta delle azioni da eseguire.
- d. Le intezioni di un agente sono soggette ad un processo di revisione dopo ogni azione da esso eseguita. ✗
- e. Le intezioni di un agente non sono mai soggette ad un processo di revisione.

Risposta parzialmente esatta.

Hai selezionato correttamente 1.
La risposta corretta è: Le intezioni sono dei vincoli che un agente deve rispettare nella scelta delle azioni da eseguire., Le intenzioni sono la sorgente della proattività di un agente.

Domanda **2**

Completo

Punteggio ottenuto 0,50 su 2,00

Contrassegna domanda

Spiegare che cosa si intende per **Model checking**.

Model checking e' una tecnica usata per controllare se una data formula ø e' conseguenza logica di un dato modello M.
Si basa sulla logica temporale lineare.

Commento:

Domanda **3**

Risposta errata

Punteggio ottenuto 0,00 su 1,50

Contrassegna domanda

Si consideri il commitment sociale C(x, y, p, q) secondo la definizione data da Munindar Singh. Quali dele seguenti affermazioni sono vere?

Scegli una o più alternative:

- a. Se q diventa falsa, il commitment è violato.
- b. y ha la aspettativa che q diventi vera nel caso che p sia vera.
- c. Se p diventa falsa, il commitment è violato.
- d. x ha la aspettativa che q diventi vera nel caso che p sia vera. ✗
- e. x è il debitore del commitment.
- f. Se p è vera e successivamente q diventa falsa, il commitment è violato. ✓
- g. y è il debitore del commitment. ✗
- h. Se p diventa vera, il commitment è soddisfatto.

Risposta errata.

La risposta corretta è: x è il debitore del commitment., y ha la aspettativa che q diventi vera nel caso che p sia vera., Se p è vera e successivamente q diventa falsa, il commitment è violato.

Domanda **4**

Parzialmente corretta

Punteggio ottenuto 1,67 su 2,50

Contrassegna domanda

Si consideri il modello di esecuzione dei comportamenti di un agente in JADE. Dire quali delle seguenti affermazioni sono vere.

Scegli una o più alternative:

- a. La gestione dei compotamenti di un agente è di tipo competitivo.
- b. La gestione dei comportamenti di un agente è di tipo pre-emptive.
- c. Quando un comportamento di un agente è selezionato, il suo metodo action viene completamente eseguito fino al suo return. ✓
- d. Ogni comportamento di un agente definisce un metodo done. ✓
- e. La sincronizzazione tra i comportamenti di un agente è ottenuta mediante le usuali primitive del linguaggio Java. ✗

Risposta parzialmente esatta.

Hai selezionato troppe opzioni.
La risposta corretta è: Quando un comportamento di un agente è selezionato, il suo metodo action viene completamente eseguito fino al suo return., Ogni comportamento di un agente definisce un metodo done.

Domanda **5**

Completo

Punteggio ottenuto 1,00 su 2,00

Contrassegna domanda

Cohen e Levesque hanno proposto una logica per modellare agenti BDI la cui semantica è basata sui mondi possibili. Spiegare cosa hanno di diverso i modelli di questa logica rispetto ai modelli della logica modale classica.

La logica di Cohen-Levesque e' una delle due logiche utili per modellare gli agenti BDI. Essa si bassa sulla logica modale e sulla logica temporale lineare.
A differenza della logica classica permette di avere agromentazioni non funzionali e considera degli operatori come il BEL, INTENT, COMMITMENT

Commento:

Domanda **6**

Completo

Punteggio ottenuto 0,00 su 2,50

Contrassegna domanda

Logica modale: definire l'**assioma K** e la regola di **necessitation**.

assioma K: $K(\phi \rightarrow \psi) = K\phi \rightarrow K\psi$

regola necessitation: $[\,]\, K\phi \rightarrow KK\phi$

Commento:

Domanda **7**

Risposta errata

Punteggio ottenuto 0,00 su 1,50

Contrassegna domanda

Con il termine performativa, nella Speech Act Theory si intende:

Scegli un'alternativa:

- a. L'azione risultante dall'atto fisico di chi parla. ✗
- b. L'atto fisico di chi parla.
- c. Il significato attribuito da chi parla.

Risposta errata.

La risposta corretta è: Il significato attribuito da chi parla.

Domanda **8**

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

Contrassegna domanda

Due giocatori A e B partecipano a una asta di Wickrey. A e B fanno una offerta maggiore di tutti gli altri partecipanti. A offre 100, mentre B offre 80. Quali delle seguenti affermazioni sono vere:

Scegli una o più alternative:

- a. B è il vincitore e paga 100.
- b. B è il vincitore e paga 80.
- c. A è il vincitore e paga 80. ✓
- d. A è il vincitore e paga 100.
- e. Una asta di Wickrey è una asta in busta chiusa. ✓

Risposta corretta.

La risposta corretta è: Una asta di Wickrey è una asta in busta chiusa., A è il vincitore e paga 80.

Domanda **9**

Completo

Punteggio ottenuto 2,00 su 3,00

Contrassegna domanda

Data la seguente matrice di payoff, spiegare in cosa consiste il *dilemma del prigioniero*.

	Alice: testify	Alice: refuse
Bob: testify	A = -5, B = -5	A = -10, B = 0
Bob: refuse	A = 0, B = -10	A = -1, B = -1

Il dilemma del prigioniero e' uno degli esempio piu' popolari della Teoria dei giochi.
In questo caso Alice e Bob sono due prigionieri arrestati dalla polizia, la quale interrogandoli separatamente propone loro un patteggiamento. Come visibile dalla matrice di payoff le condizioni sono queste:
Se entrambi non confessassero riceverebbero 1 anno di reclusione a testa
Se entrambi confessassero riceverebbero 5 anni di reclusione a testa
Se uno dei due confessasse e l'altro no, il testimone sarebbe totalmente libero mentre l'altro prigioniero sconterebbe una pena di 10 anni.
Dato che i due prigionieri tendono a preservarsi egoisticamente il punto di equilibrio lo si individua nella situazione in cui sia Bob che Alice confessino per evitare di essere traditi dal proprio complice.

Commento:

Domanda **10**

Parzialmente corretta

Punteggio ottenuto 1,40 su 3,00

Contrassegna domanda

Si consideri il seguente frammento di codice in AgentSpeak(l):

```
a(1).
a(2).
a(3).
a(4).
a(5).

+!start1 : a(X) & .print("Start1 X: ", X) & X=3 <- .print("Start1 (body) X: ", X) .

+!start2 : true <- ?a(Y); .print("Start2 (body) Y: ", Y) .

+!start3 : a(X) & .print("Start3 X: ", X) & X=3 <- ?a(Y); .print("Start3 (body) Y: ", Y); Y=3 .
```

Quali delle seguenti affermazioni sono vere?

Scegli una o più alternative:

- [ ] a.
  L'evento +!start2 ha come output:
  [test] Start2 (body) Y: 1

  [test] Start2 (body) Y: 2

  [test] Start2 (body) Y: 3
- [x] b.
  L'evento +!start2 ha come output:
  [test] Start2 (body) Y: 1 ✓
- [ ] c. L'evento +!start3 ha come output:
  [test] Start3 X: 1
  [test] Start3 X: 2
  [test] Start3 X: 3
  [test] Start3 (body) Y: 1
  [test] No failure
- [ ] d. L'evento +!start3 ha come output:
  [test] Start3 X: 1
  [test] Start3 X: 2
  [test] Start3 X: 3
  [test] Start3 (body) Y: 1
  [test] Start3 (body) Y: 2

  [test] Start3 (body) Y: 3
- [x] e. L'evento +!start3 ha come output:
  [test] Start3 X: 3
  [test] Start3 (body) Y: 3 ✗

e. L'evento +!start3 ha come output:
[test] Start3 X: 3
[test] Start3 (body) Y: 3 ✗

f. L'evento +!start2 ha come output:
[test] Start2 (body) Y: 3

g. L'evento +!start1 ha come output:
[test] Start1 X: 1
[test] Start1 X: 2
[test] Start1 X: 3
[test] Start1 (body) X: 3 ✓

h.
L'evento +!start1 ha come output:
[test] Start1 X: 1
[test] Start1 X: 2
[test] Start1 X: 3
[test] Start1 (body) X: 1
[test] Start1 (body) X: 2

[test] Start1 (body) X: 3

Risposta parzialmente esatta.

Hai selezionato correttamente 2.
La risposta corretta è: L'evento +!start1 ha come output:
[test] Start1 X: 1
[test] Start1 X: 2
[test] Start1 X: 3
[test] Start1 (body) X: 3,
L'evento +!start2 ha come output:
[test] Start2 (body) Y: 1, L'evento +!start3 ha come output:
[test] Start3 X: 1
[test] Start3 X: 2
[test] Start3 X: 3
[test] Start3 (body) Y: 1
[test] No failure

Domanda **11**

Risposta non data

Punteggio max.: 2,50

Contrassegna domanda

Si consideri il seguente esempio di programma Concurrent METATEM.

Example. A **'resource producer'** rp:

rp (ask1, ask2) [give1, give2]

1. $\bigcirc$ask1 $\Rightarrow$ $\Diamond$give1
2. $\bigcirc$ask2 $\Rightarrow$ $\Diamond$give2
3. **start** $\Rightarrow$ $\Box\neg$(give1 $\wedge$ give2)

Two **consumers**:

rc1 (give1) [ask1]

4. **start** $\Rightarrow$ ask1
5. $\bigcirc$ask1 $\Rightarrow$ ask1

rc2 (ask1, give2) [ask2]

6. $\bigcirc$(ask1 $\wedge$ $\neg$ask2) $\Rightarrow$ ask2

Commentare brevemente le regole da 1 a 6.

Domanda **12**

Parzialmente corretta

Punteggio ottenuto 1,20 su 1,50

Contrassegna domanda

Si consideri l'attività di *Practical Reasoning*, quali delle seguenti affermazioni sono vere?

Scegli una o più alternative:

- a. L'attività di means-ends reasoning stabilisce quale stato raggiungere.
- b. L'attività di means-ends reasoning produce come risultato le intezioni dell'agente.
- c. L'attività di deliberation stabilisce quale stato raggiungere. ✓
- d. L'attività di deliberation stabilisce come raggiungere uno stato.
- e. L'attività di means-ends reasoning verifica l'effettiva eseguibilità di un piano per soddisfare le intenzioni. ✗
- f. L'attività di deliberation produce come risultato le intenzioni dell'agente. ✓
- g. L'attività di means-ends reasoning stabilisce come raggiungere uno stato. ✓
- h. L'attività di deliberation delibera la sequenza di azioni che l'agente deve eseguire per raggiungere uno stato.

Risposta parzialmente esatta.

Hai selezionato troppe opzioni.
La risposta corretta è: L'attività di deliberation stabilisce quale stato raggiungere., L'attività di deliberation produce come risultato le intenzioni dell'agente., L'attività di means-ends reasoning stabilisce come raggiungere uno stato.

Domanda **13**

Risposta errata

Punteggio ottenuto 0,00 su 2,00

Contrassegna domanda

Quali delle seguente affermazioni relative ad AgentSpeak(L) sono vere?

Scegli una o più alternative:

- a. AgentSpeak(L) è implentato in linguaggio Java.
- b. AgentSpeak(L) rappresenta achievement goal in un piano ma non test goal.
- c. AgentSpeak(L) ha una funzione di selezione per determinare gli eventi rilevanti. ✗
- d. In AgentSpeak(L) sono rappresentati i belief e i desire ma non le intention. ✗
- e. AgentSpeak(L) ha una funzione di selezione per determinare i piani rilevanti.
- f. AgentSpeak(L) è un linguaggio dichiarativo. ✓
- g. AgentSpeak(L) distingue tra eventi interni ed esterni.
- h. AgentSpeak(L) ha una funzione di selezione per determinare i belief da adottare. ✗

Risposta errata.

La risposta corretta è: AgentSpeak(L) è un linguaggio dichiarativo., AgentSpeak(L) distingue tra eventi interni ed esterni.

Domanda **14**

Risposta non data

Punteggio max.: 2,50

Contrassegna domanda

Cosa si intende per **strategia pura** e **strategia mista** nella teoria dei giochi.

Domanda **15**

Parzialmente corretta

Punteggio ottenuto 2,00 su 3,00

Contrassegna domanda

Si consideri il seguente frammento di codice di un metodo action di un behaviour Jade.

```
public void action() {
    ...
    MessageTemplate tpl = MessageTemplate.MatchOntology("Test-Ontology");
    ACLMessage msg = myAgent.receive(tpl);
    if (msg != null) {
        // Process the message
    }
    else {
        block();
    }
    ...
}
```

Quali delle seguenti affermazioni sono vere?

Scegli una o più alternative:

- [x] a. La coda dei messaggi è comune a tutti i comportamenti di uno stesso agente. ✓
- [x] b. Se un messaggio non soddisfa il template tpl non viene prelevato dalla coda dei messaggi. ✓
- [x] c. Il processo dell'agente risulta in wait all'esecuzione dell'istruzione block() sino alla ricezione del prossimo messaggio. ✗
- [ ] d. Alla ricezione di un messaggio l'esecuzione riprende dal punto successivo l'istruzione block().
- [ ] e. L'esecuzione dell'istruzione block() ferma immediatamente l'esecuzione del comportamento che la contiene.

Risposta parzialmente esatta.

Hai selezionato troppe opzioni.
La risposta corretta è: Se un messaggio non soddisfa il template tpl non viene prelevato dalla coda dei messaggi., La coda dei messaggi è comune a tutti i comportamenti di uno stesso agente.

Domanda **14**

Risposta non data

Punteggio max.: 2,50

Contrassegna domanda

Cosa si intende per **strategia pura** e **strategia mista** nella teoria dei giochi.

Domanda **12**

Parzialmente corretta

Punteggio ottenuto 1,20 su 1,50

Contrassegna domanda

Si consideri l'attività di *Practical Reasoning*, quali delle seguenti affermazioni sono vere?

Scegli una o più alternative:

- a. L'attività di means-ends reasoning stabilisce quale stato raggiungere.
- b. L'attività di means-ends reasoning produce come risultato le intezioni dell'agente.
- c. L'attività di deliberation stabilisce quale stato raggiungere. ✓
- d. L'attività di deliberation stabilisce come raggiungere uno stato.
- e. L'attività di means-ends reasoning verifica l'effettiva eseguibilità di un piano per soddisfare le intenzioni. ✗
- f. L'attività di deliberation produce come risultato le intenzioni dell'agente. ✓
- g. L'attività di means-ends reasoning stabilisce come raggiungere uno stato. ✓
- h. L'attività di deliberation delibera la sequenza di azioni che l'agente deve eseguire per raggiungere uno stato.

Risposta parzialmente esatta.

Hai selezionato troppe opzioni.
La risposta corretta è: L'attività di deliberation stabilisce quale stato raggiungere., L'attività di deliberation produce come risultato le intenzioni dell'agente., L'attività di means-ends reasoning stabilisce come raggiungere uno stato.

Domanda **13**

Risposta errata

Punteggio ottenuto 0,00 su 2,00

Contrassegna domanda

Quali delle seguente affermazioni relative ad AgentSpeak(L) sono vere?

Scegli una o più alternative:

- a. AgentSpeak(L) è implentato in linguaggio Java.
- b. AgentSpeak(L) rappresenta achievement goal in un piano ma non test goal.
- c. AgentSpeak(L) ha una funzione di selezione per determinare gli eventi rilevanti. ✗
- d. In AgentSpeak(L) sono rappresentati i belief e i desire ma non le intention. ✗
- e. AgentSpeak(L) ha una funzione di selezione per determinare i piani rilevanti.
- f. AgentSpeak(L) è un linguaggio dichiarativo. ✓
- g. AgentSpeak(L) distingue tra eventi interni ed esterni.
- h. AgentSpeak(L) ha una funzione di selezione per determinare i belief da adottare. ✗

Risposta errata.

La risposta corretta è: AgentSpeak(L) è un linguaggio dichiarativo., AgentSpeak(L) distingue tra eventi interni ed esterni.

Domanda **11**

Risposta non data

Punteggio max.: 2,50

Contrassegna domanda

Si consideri il seguente esempio di programma Concurrent METATEM.

Example. A '**resource producer**' rp:

rp (ask1, ask2) [give1, give2]

1. $\bigcirc$ask1 $\Rightarrow$ $\Diamond$give1
2. $\bigcirc$ask2 $\Rightarrow$ $\Diamond$give2
3. **start** $\Rightarrow$ $\Box\neg$(give1 $\wedge$ give2)

Two **consumers**:

rc1 (give1) [ask1]

4. **start** $\Rightarrow$ ask1
5. $\bigcirc$ask1 $\Rightarrow$ ask1

rc2 (ask1, give2) [ask2]

6. $\bigcirc$(ask1 $\wedge$ $\neg$ask2) $\Rightarrow$ ask2

Commentare brevemente le regole da 1 a 6.

e. L'evento +!start3 ha come output:
[test] Start3 X: 3
[test] Start3 (body) Y: 3 ✗

f. L'evento +!start2 ha come output:
[test] Start2 (body) Y: 3

g. L'evento +!start1 ha come output:
[test] Start1 X: 1
[test] Start1 X: 2
[test] Start1 X: 3
[test] Start1 (body) X: 3 ✓

h.
L'evento +!start1 ha come output:
[test] Start1 X: 1
[test] Start1 X: 2
[test] Start1 X: 3
[test] Start1 (body) X: 1
[test] Start1 (body) X: 2

[test] Start1 (body) X: 3

Risposta parzialmente esatta.

Hai selezionato correttamente 2.
La risposta corretta è: L'evento +!start1 ha come output:
[test] Start1 X: 1
[test] Start1 X: 2
[test] Start1 X: 3
[test] Start1 (body) X: 3,
L'evento +!start2 ha come output:
[test] Start2 (body) Y: 1, L'evento +!start3 ha come output:
[test] Start3 X: 1
[test] Start3 X: 2
[test] Start3 X: 3
[test] Start3 (body) Y: 1
[test] No failure

Domanda **10**

Parzialmente corretta

Punteggio ottenuto 1,40 su 3,00

Contrassegna domanda

Si consideri il seguente frammento di codice in AgentSpeak(l):

```
a(1).
a(2).
a(3).
a(4).
a(5).

+!start1 : a(X) & .print("Start1 X: ", X) & X=3 <- .print("Start1 (body) X: ", X) .

+!start2 : true <- ?a(Y); .print("Start2 (body) Y: ", Y) .

+!start3 : a(X) & .print("Start3 X: ", X) & X=3 <- ?a(Y); .print("Start3 (body) Y: ", Y); Y=3 .
```

Quali delle seguenti affermazioni sono vere?

Scegli una o più alternative:

- [ ] a.
  L'evento +!start2 ha come output:
  [test] Start2 (body) Y: 1

  [test] Start2 (body) Y: 2

  [test] Start2 (body) Y: 3
- [x] b.
  L'evento +!start2 ha come output:
  [test] Start2 (body) Y: 1 ✓
- [ ] c. L'evento +!start3 ha come output:
  [test] Start3 X: 1
  [test] Start3 X: 2
  [test] Start3 X: 3
  [test] Start3 (body) Y: 1
  [test] No failure
- [ ] d. L'evento +!start3 ha come output:
  [test] Start3 X: 1
  [test] Start3 X: 2
  [test] Start3 X: 3
  [test] Start3 (body) Y: 1
  [test] Start3 (body) Y: 2

  [test] Start3 (body) Y: 3
- [x] e. L'evento +!start3 ha come output:
  [test] Start3 X: 3
  [test] Start3 (body) Y: 3 ✗

Domanda **8**

Risposta corretta

Punteggio ottenuto 2,00 su 2,00

Contrassegna domanda

Due giocatori A e B partecipano a una asta di Wickrey. A e B fanno una offerta maggiore di tutti gli altri partecipanti. A offre 100, mentre B offre 80. Quali delle seguenti affermazioni sono vere:

Scegli una o più alternative:

- a. B è il vincitore e paga 100.
- b. B è il vincitore e paga 80.
- c. A è il vincitore e paga 80. ✓
- d. A è il vincitore e paga 100.
- e. Una asta di Wickrey è una asta in busta chiusa. ✓

Risposta corretta.

La risposta corretta è: Una asta di Wickrey è una asta in busta chiusa., A è il vincitore e paga 80.

Domanda **9**

Completo

Punteggio ottenuto 2,00 su 3,00

Contrassegna domanda

Data la seguente matrice di payoff, spiegare in cosa consiste il *dilemma del prigioniero*.

	Alice: testify	Alice: refuse
Bob: testify	A = -5, B = -5	A = -10, B = 0
Bob: refuse	A = 0, B = -10	A = -1, B = -1

Il dilemma del prigioniero e' uno degli esempio piu' popolari della Teoria dei giochi.
In questo caso Alice e Bob sono due prigionieri arrestati dalla polizia, la quale interrogandoli separatamente propone loro un patteggiamento. Come visibile dalla matrice di payoff le condizioni sono queste:
Se entrambi non confessassero riceverebbero 1 anno di reclusione a testa
Se entrambi confessassero riceverebbero 5 anni di reclusione a testa
Se uno dei due confessasse e l'altro no, il testimone sarebbe totalmente libero mentre l'altro prigioniero sconterebbe una pena di 10 anni.
Dato che i due prigionieri tendono a preservarsi egoisticamente il punto di equilibrio lo si individua nella situazione in cui sia Bob che Alice confessino per evitare di essere traditi dal proprio complice.

Commento:

Domanda **5**

Completo

Punteggio ottenuto 1,00 su 2,00

Contrassegna domanda

Cohen e Levesque hanno proposto una logica per modellare agenti BDI la cui semantica è basata sui mondi possibili. Spiegare cosa hanno di diverso i modelli di questa logica rispetto ai modelli della logica modale classica.

La logica di Cohen-Levesque e' una delle due logiche utili per modellare gli agenti BDI. Essa si bassa sulla logica modale e sulla logica temporale lineare.
A differenza della logica classica permette di avere agromentazioni non funzionali e considera degli operatori come il BEL, INTENT, COMMITMENT

Commento:

Domanda **6**

Completo

Punteggio ottenuto 0,00 su 2,50

Contrassegna domanda

Logica modale: definire l'**assioma K** e la regola di **necessitation**.

assioma K: $K(\phi \rightarrow \psi) = K\phi \rightarrow K\psi$

regola necessitation: $[]\ K\phi \rightarrow KK\phi$

Commento:

Domanda **7**

Risposta errata

Punteggio ottenuto 0,00 su 1,50

Contrassegna domanda

Con il termine performativa, nella Speech Act Theory si intende:

Scegli un'alternativa:

- a. L'azione risultante dall'atto fisico di chi parla. ✗
- b. L'atto fisico di chi parla.
- c. Il significato attribuito da chi parla.

Risposta errata.

La risposta corretta è: Il significato attribuito da chi parla.

Domanda **3**

Risposta errata

Punteggio ottenuto 0,00 su 1,50

Contrassegna domanda

Si consideri il commitment sociale C(x, y, p, q) secondo la definizione data da Munindar Singh. Quali dele seguenti affermazioni sono vere?

Scegli una o più alternative:

- a. Se q diventa falsa, il commitment è violato.
- b. y ha la aspettativa che q diventi vera nel caso che p sia vera.
- c. Se p diventa falsa, il commitment è violato.
- d. x ha la aspettativa che q diventi vera nel caso che p sia vera. ✗
- e. x è il debitore del commitment.
- f. Se p è vera e successivamente q diventa falsa, il commitment è violato. ✓
- g. y è il debitore del commitment. ✗
- h. Se p diventa vera, il commitment è soddisfatto.

Risposta errata.

La risposta corretta è: x è il debitore del commitment., y ha la aspettativa che q diventi vera nel caso che p sia vera., Se p è vera e successivamente q diventa falsa, il commitment è violato.

Domanda **4**

Parzialmente corretta

Punteggio ottenuto 1,67 su 2,50

Contrassegna domanda

Si consideri il modello di esecuzione dei comportamenti di un agente in JADE. Dire quali delle seguenti affermazioni sono vere.

Scegli una o più alternative:

- a. La gestione dei compotamenti di un agente è di tipo competitivo.
- b. La gestione dei comportamenti di un agente è di tipo pre-emptive.
- c. Quando un comportamento di un agente è selezionato, il suo metodo action viene completamente eseguito fino al suo return. ✓
- d. Ogni comportamento di un agente definisce un metodo done. ✓
- e. La sincronizzazione tra i comportamenti di un agente è ottenuta mediante le usuali primitive del linguaggio Java. ✗

Risposta parzialmente esatta.

Hai selezionato troppe opzioni.
La risposta corretta è: Quando un comportamento di un agente è selezionato, il suo metodo action viene completamente eseguito fino al suo return., Ogni comportamento di un agente definisce un metodo done.

Domanda **15**

Parzialmente corretta

Punteggio ottenuto 2,00 su 3,00

Contrassegna domanda

Si consideri il seguente frammento di codice di un metodo action di un behaviour Jade.

```
public void action() {
    ...
    MessageTemplate tpl = MessageTemplate.MatchOntology("Test-Ontology");
    ACLMessage msg = myAgent.receive(tpl);
    if (msg != null) {
        // Process the message
    }
    else {
        block();
    }
    ...
}
```

Quali delle seguenti affermazioni sono vere?

Scegli una o più alternative:

- [x] a. La coda dei messaggi è comune a tutti i comportamenti di uno stesso agente. ✓
- [x] b. Se un messaggio non soddisfa il template tpl non viene prelevato dalla coda dei messaggi. ✓
- [x] c. Il processo dell'agente risulta in wait all'esecuzione dell'istruzione block() sino alla ricezione del prossimo messaggio. ✗
- [ ] d. Alla ricezione di un messaggio l'esecuzione riprende dal punto successivo l'istruzione block().
- [ ] e. L'esecuzione dell'istruzione block() ferma immediatamente l'esecuzione del comportamento che la contiene.

Risposta parzialmente esatta.

Hai selezionato troppe opzioni.
La risposta corretta è: Se un messaggio non soddisfa il template tpl non viene prelevato dalla coda dei messaggi., La coda dei messaggi è comune a tutti i comportamenti di uno stesso agente.

Domanda 1

Parzialmente corretta

Punteggio ottenuto 0,25 su 1,50

Contrassegna domanda

Si considerino le intenzioni, quali delle seguenti affermazioni sono vere?

Scegli una o più alternative:

- a. Le intenzioni sono piani, sequenze di azioni che un agente esegue per raggiungere un suo goal.
- b. Le intenzioni sono la sorgente della proattività di un agente. ✓
- c. Le intezioni sono dei vincoli che un agente deve rispettare nella scelta delle azioni da eseguire.
- d. Le intezioni di un agente sono soggette ad un processo di revisione dopo ogni azione da esso eseguita. ✗
- e. Le intezioni di un agente non sono mai soggette ad un processo di revisione.

Risposta parzialmente esatta.

Hai selezionato correttamente 1.
La risposta corretta è: Le intezioni sono dei vincoli che un agente deve rispettare nella scelta delle azioni da eseguire., Le intenzioni sono la sorgente della proattività di un agente.

Domanda 2

Completo

Punteggio ottenuto 0,50 su 2,00

Contrassegna domanda

Spiegare che cosa si intende per **Model checking**.

Model checking e' una tecnica usata per controllare se una data formula ø e' conseguenza logica di un dato modello M.
Si basa sulla logica temporale lineare.

Commento:

Domanda 9

Completo

Punteggio ottenuto 2,00 su 2,50

Contrassegna domanda

Spiegare la differenza tra coordinazione, cooperazione e negoziazione.

La coordinazione e la cooperazione riguardano la collaborazione tra due sistemi non competitivi, la negoziazione e' un caso di accordo tra due agenti in competizione. Con coordinazione si intende la capacita' di lavorare in un ambiente condiviso, cooperare si intende la collaborazione che hanno due agenti in un ambiente condiviso, la negoziazione permette a degli agenti competitivi di riuscire a lavorare insieme trovando un accordo che lo permetta.

Commento:

Domanda 8

Completo

Punteggio ottenuto 1,66 su 2,50

Contrassegna domanda

Si consideri il seguente protocollo a commitment:

1. cfp means create(C(i, p, propose, accept v reject))
2. accept means none
3. reject means release(C(p, i, accept, done v failure))
4. propose means create(C(p, i, accept, done v failure))
5. refuse means release(C(i, p, propose, accept v reject))
6. done means none
7. failure means none

Scrivere l'insieme di commitment presenti nello stato sociale dopo l'esecuzione delle sequenze di azioni:

1. cfp; propose.
2. propose; accept.
3. cfp; refuse.

1 Lo stato sociale e' {C(i, p, T, accept v reject), C(p, i, accept, done v failure)}

2 Lo stato sociale e' {C(p, i, T, done v failure)}

3 Lo stato sociale e' {C(i, p, propose, accept v reject)}

Commento:

Domanda 7
Completo
Punteggio ottenuto 2,50 su 2,50
Contrassegna domanda

Si considerino l'Agent Control Loop V3, V4 e V5 seguenti. Quali problemi affronta l'Agent Control Loop Version 4 rispetto alla Version 3 e quali problemi affronta l'Agent Control Loop Version 5 rispetto alla Version 4 e come.

```
Agent Control Loop Version 3

B := B0;
I := I0;
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)
end while
```

```
Agent Control Loop Version 4

B := B0;
I := I0;
while true do
    get next percept ρ;
    B := brf(B, ρ);
    D := options(B, I);
    I := filter(B, D, I);
    π := plan(B, I);
    while not empty(π) do
        α := hd(π);
        execute(α);
        π := tail(π);
        get next percept ρ;
        B := brf(B, ρ);
        if not sound(π, I, B) then
            π := plan(B, I)
        end-if
    end-while
end-while
```

```
Agent Control Loop Version 5

B := B0;
I := I0;
while true do
    get next percept ρ;
    B := brf(B, ρ);
    D := options(B, I);
    I := filter(B, D, I);
    π := plan(B, I);
    while not( empty(π)
            or succeeded(I, B)
            or impossible(I, B)) do
        α := hd(π);
        execute(α);
        π := tail(π);
        get next percept ρ;
        B := brf(B, ρ);
        if not sound(π, I, B) then
            π := plan(B, I)
        end-if
    end-while
end-while
```

Nell'ACL v.4 a differenza della v.3 i piani non vengono eseguiti subito tutti ma viene aggiunto un ciclo che ci permette di eseguire e scomporre il piano in azioni (i piano sono gestiti come una lista di azioni), avendo la possibilita' di aggiornare per ogni azione eseguita le credenze in base alla percezione e ponendo anche un controllo effettuato ad ogni azione eseguita, dove si controlla se il piano e' ancora coerente con le credenze e le intenzioni per poter continuarlo ad eseguirlo. Nella v.5 invece a differenza della v. 4 sono stati aggiungi dei controlli al ciclo dell'esecuzione del piano, difatti ora da quel ciclo non si esce piu' solo se le lista dei piani e' vuota ma anche se l'obiettivo e' stato raggiunto prima che possa finire il piano oppure se ci si e' accorti che l'obbiettivo e' impossibile da raggiungere.

Domanda 7
Completo
Punteggio ottenuto 2,50 su 2,50
Contrassegna domanda

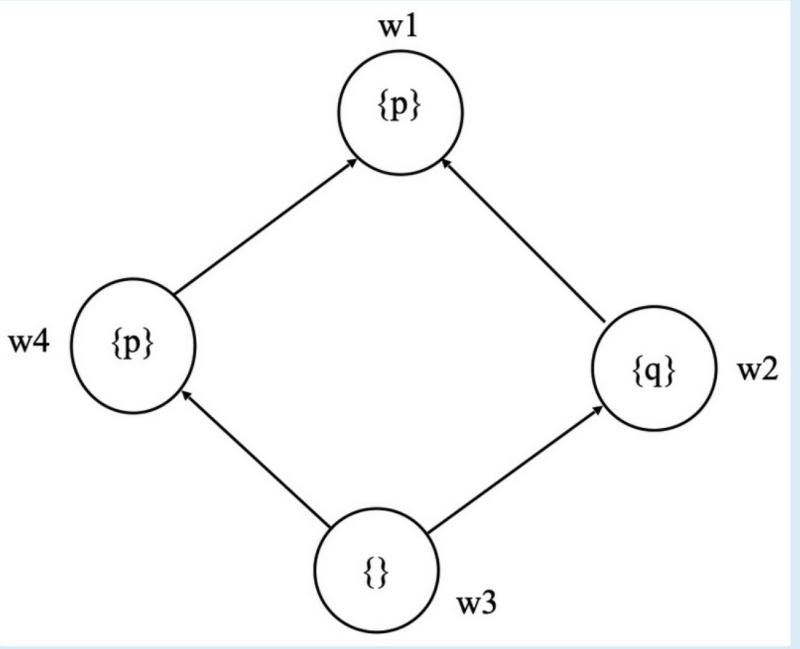
Domanda 6

Parzialmente corretta

Punteggio ottenuto 1,25 su 2,50

Contrassegna domanda

Si consideri il seguente modello di Kripke.



Dire quali delle seguenti affermazioni sono vere.

Scegli una o più alternative:

- [x] a. $[](p \vee q)$ è vero nel mondo w3 ✓
- [ ] b. $[]p$ è vero nel mondo w3
- [ ] c. $([]p \vee []q)$ è vero nel mondo w3
- [ ] d. $[][]p$ è vero nel mondo w3

Risposta parzialmente esatta.

Hai selezionato correttamente 1.

La risposta corretta è: $[](p \vee q)$ è vero nel mondo w3, $[][]p$ è vero nel mondo w3

Domanda 5

Completo

Punteggio ottenuto 2,50 su 2,50

Contrassegna domanda

Nella logica epistemica si assume che per l'operatore **K** (conoscenza) valgano le seguenti proprietà:

1. La conoscenza è non contraddittoria
2. Ciò che si conosce è vero
3. Vale la proprietà della introspezione positiva
4. Vale la proprietà dell'introspezione negativa

Fornire i quattro assiomi che rappresentano queste proprietà.

1. $K\varphi \rightarrow \neg K\neg\varphi$ $\quad B\varphi \rightarrow \neg B\neg\varphi$ Assioma di serialita'
2. $K\varphi \rightarrow \varphi$ Assioma di riflessivita'
3. $K\varphi \rightarrow KK\varphi$ $\quad B\varphi \rightarrow BB\varphi$ Assioma di transitivita'
4. $\neg K\neg\varphi \rightarrow K(\neg K\neg\varphi)$ $\quad \neg B\neg\varphi \rightarrow B(\neg B\neg\varphi)$ Assioma euclideoN

Commento:

Domanda 4

Completo

Punteggio ottenuto 1,00 su 2,50

Contrassegna domanda

Cosa si intende per strategia pura e strategia mista nella teoria dei giochi?

Per strategia mista si intende un tipo di strategia legata alle probabilita', quindi la scelta per il giocatore e' guidata anche dalle probabilita'.

Nella strategia pura il giocatore dovra' scegliere in base alle proprie informazioni la strategia migliore, come ad esempio puo' essere una strategia dominante.

Commento:

Domanda 3

Completo

Punteggio ottenuto 2,00 su 2,50

Contrassegna domanda

Spiegare che cosa si intende per **strategia dominante** nella teoria dei giochi. Si consideri la seguente **matrice di payoff**, dove sia Alice che Bob possono eseguire o l'azione a1 o a2. Spiegare se Alice ha una strategia dominante.

	Alice: a1	Alice: a2
Bob: a1	A = 0, B = 0	A = 1, B = 3
Bob: a2	A = 3, B = 1	A = 2, B = 2

Per strategia dominante si intende la strategia migliore che un giocatore puo' adottare a prescindere da quello che scelga l'altro giocatore. Ad esempio la strategia dominante per Alice viene scelta in base alla matrice di payoff senza contare cosa possa scegliere Bob (non lo si puo' sapere a priori). In questo specifico esempio ne Bob ne Alice hanno una strategia dominante.

Commento:

Domanda 2

Completo

Punteggio ottenuto 2,30 su 2,50

Contrassegna domanda

Spiegare sinteticamente la motivazione e la struttura della subsumption architecture proposta da Brooks.

In questa architettura proposta da Brooks l'agente ha un organizzazione a livelli e il suo comportamento e' determinato tra l'interazione tra qesti livelli. Ogni livello ha un suo compito ed un suo comportamento. I livelli piu' bassi eseguono le azioni piu' semplici e piu' rapide e sono quelli che costituiscono le azioni piu' reattive e immediate dell'agente. In generalu tutti i livelli sono strutturati come regole. Questi livelli sono definiti come automi a stati finiti che possono passare da uno stato all'altro continuando ad interagire fra loro. Il nome e' dovuto al fatto che ogni livello sussume i livelli sottostanti aggiungendo nuove capacita'.

Commento:

Domanda 12

Parzialmente corretta

Punteggio ottenuto 1,17 su 2,50

Contrassegna domanda

Si consideri il seguente frammento di codice in AgentSpeak(l):

```
a(1).
a(2).
a(3).
a(4).
a(5).

+!start1 : a(X) & .print("Start1 X: ", X) & X=3 <- .print("Start1 (body) X: ", X) .

+!start2 : true <- ?a(Y); .print("Start2 (body) Y: ", Y) .

+!start3 : a(X) & .print("Start3 X: ", X) & X=3 <- ?a(Y); .print("Start3 (body) Y: ", Y); Y=3 .
```

Quali delle seguenti affermazioni sono vere?

Scegli una o più alternative:

- [ ] a.
  L'evento +!start1 ha come output:
  [test] Start1 X: 1
  [test] Start1 X: 2
  [test] Start1 X: 3
  [test] Start1 (body) X: 1
  [test] Start1 (body) X: 2

  [test] Start1 (body) X: 3
- [x] b. L'evento +!start1 ha come output:
  [test] Start1 X: 1
  [test] Start1 X: 2
  [test] Start1 X: 3
  [test] Start1 (body) X: 3 ✓
- [x] c. L'evento +!start3 ha come output:
  [test] Start3 X: 1
  [test] Start3 X: 2
  [test] Start3 X: 3
  [test] Start3 (body) Y: 1
  [test] No failure ✓
- [ ] d.
  L'evento +!start2 ha come output:
  [test] Start2 (body) Y: 1
- [x] e.
  L'evento +!start2 ha come output:
  [test] Start2 (body) Y: 1

  [test] Start2 (body) Y: 2

  [test] Start2 (body) Y: 3 ✗
- [ ] f. L'evento +!start3 ha come output:
  [test] Start3 X: 3
  [test] Start3 (body) Y: 3
- [ ] g. L'evento +!start3 ha come output:
  [test] Start3 X: 1
  [test] Start3 X: 2
  [test] Start3 X: 3
  [test] Start3 (body) Y: 1
  [test] Start3 (body) Y: 2

  [test] Start3 (body) Y: 3
- [ ] h. L'evento +!start2 ha come output:
  [test] Start2 (body) Y: 3

Risposta parzialmente esatta.

Hai selezionato correttamente 2.
La risposta corretta è: L'evento +!start1 ha come output:
[test] Start1 X: 1
[test] Start1 X: 2
[test] Start1 X: 3
[test] Start1 (body) X: 3,
L'evento +!start2 ha come output:
[test] Start2 (body) Y: 1, L'evento +!start3 ha come output:
[test] Start3 X: 1
[test] Start3 X: 2
[test] Start3 X: 3
[test] Start3 (body) Y: 1
[test] No failure

Domanda 11

Completo

Punteggio ottenuto 2,50 su 2,50

Contrassegna domanda

Spiegare la differenza tra **practical reasoning** e **theoretical reasoning**.

Il pratical reasoning e' un ragionamento guidato dalle azioni, mentre il theoretical reasoning e' un ragionamento guidato dalla credenze.

Commento:

Domanda 10

Completo

Punteggio ottenuto 2,50 su 2,50

Contrassegna domanda

Si consideri il seguente programma Jade:

```
import jade.core.Agent;
import jade.core.behaviours.Behaviour;

public class Esame20190618 extends Agent {
   private int counter = 0;
   protected void setup() {
      System.out.println("Adding first behaviour");
      addBehaviour(new Behaviour() {
         public void action() {
            if ((counter % 2) == 0)
               System.out.println(counter);
         }
         public boolean done() {
            return counter >= 20;
         }
      } );
      System.out.println("Adding second behaviour");
      addBehaviour(new Behaviour() {
         public void action() {
            if ((counter % 2) == 1)
               System.out.println(counter);
         }
         public boolean done() {
            return counter >= 20;
         }
      } );
      System.out.println("Adding third behaviour");
      addBehaviour(new Behaviour() {
         public void action() {
            counter++;
         }
         public boolean done() {
            return counter >= 20;
         }
      } );

   }
   protected void takeDown() {
      System.out.println("Bye...");
   }
}
```

Dire cosa viene stampato a console durante l'esecuzione.

```
Adding first behaviour
Adding second behaviour
Adding third behaviour
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
```

```
Adding first behaviour
Adding second behaviour
Adding third behaviour
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
```

Domanda 1

Completo

Punteggio ottenuto 2,00 su 2,50

Contrassegna domanda

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)))
```

In questo esempio, un agente (?agent) richiede se al tempo (?time) puo' essere eseguita l'azione (?action), B riceve la richiesta. Le condizioni di messaggio e le condizioni mentali per soddisfare questa richiesta sono le seguenti:

B controlla se ora (?now) e' amico con l'agente (?agent), se puo' eseguire l'azione (?action), se non ha nessun commit e se non sta facendo nessun'altra azione (?any_action) allo stesso tempo (?time), allora puo' eseguire l'azione (?action) al tempo (?time).

Commento: