Briscola in 5: un sistema esperto per una strategia a regole

Mattia Vinci

Università degli studi di Torino
Dipartimento di Informatica

Relatore: Dott. Roberto Micalizio

Venerdì 17 luglio 2015

Indice

Indice

Intelligenza Artificiale e giochi

Ambito dei giochi esplorato sin dagli albori (anni ‘50) dell’Intelligenza Artificiale (scacchi, dama, bridge...)

Giochi forniscono

- ♢ problemi complessi (ampio spazio di ricerca, incertezza)
- ♢ modelli di situazione riproponibili nel mondo reale

Background

Alcune metodologie passate in rassegna:

- Ricerca nello spazio degli stati e algoritmo `minimax`
- Teoria dei giochi
- Sistemi esperti

Indice

Caratteristiche del gioco

La *briscola in 5* è un gioco di carte derivato dalla briscola classica, ma con qualche significativa differenza:

- le squadre non sono simmetriche
  - 2 *soci* (*socio* + *giaguaro*)
  - 3 *villani*
- vi è una fase d'asta iniziale
- la formazione delle squadre non è nota a tutti

Caratteristiche del gioco

La *briscola in 5* è un gioco di carte derivato dalla briscola classica, ma con qualche significativa differenza:

- ♢ le squadre non sono simmetriche
  - ♠ 2 *soci* (*socio* + *giaguaro*)
  - ♠ 3 *villani*
- ♢ vi è una fase d'asta iniziale
- ♢ la formazione delle squadre non è nota a tutti

Caratteristiche del gioco

La *briscola in 5* è un gioco di carte derivato dalla briscola classica, ma con qualche significativa differenza:

- le squadre non sono simmetriche
  - 2 *soci* (*socio* + *giaguaro*)
  - 3 *villani*
- vi è una fase d'asta iniziale
- la formazione delle squadre non è nota a tutti

Caratteristiche del gioco

La *briscola in 5* è un gioco di carte derivato dalla briscola classica, ma con qualche significativa differenza:

- le squadre non sono simmetriche
  - 2 *soci* (*socio* + *giaguaro*)
  - 3 *villani*
- vi è una fase d'asta iniziale
- la formazione delle squadre non è nota a tutti

La briscola in 5 dal punto di vista dell'IA

Informazione incompleta sulle squadre per alcuni giocatori

$\Longrightarrow$

avversari o compagni?

Ambiente della briscola in 5: nè *competitivo* nè *cooperativo*

$\Longrightarrow$

ricerca nello spazio degli stati e *teoria dei giochi* inapplicabili: impossibilità di assegnare *valori di utilità* univoci.

- *valori di utilità* dipendono dalla squadra cui appartiene il giocatore che prende la mano

$\Downarrow$

sistema esperto

La briscola in 5 dal punto di vista dell'IA

Informazione incompleta sulle squadre per alcuni giocatori

$\Longrightarrow$

avversari o compagni?

Ambiente della briscola in 5: nè *competitivo* nè *cooperativo*

$\Longrightarrow$

ricerca nello spazio degli stati e *teoria dei giochi* inapplicabili: impossibilità di assegnare *valori di utilità* univoci.

- *valori di utilità* dipendono dalla squadra cui appartiene il giocatore che prende la mano

$\Downarrow$

sistema esperto

La briscola in 5 dal punto di vista dell'IA

Informazione incompleta sulle squadre per alcuni giocatori

$\Longrightarrow$

avversari o compagni?

Ambiente della briscola in 5: nè *competitivo* nè *cooperativo*

$\Longrightarrow$

ricerca nello spazio degli stati e *teoria dei giochi* inapplicabili: impossibilità di assegnare *valori di utilità* univoci.

- *valori di utilità* dipendono dalla squadra cui appartiene il giocatore che prende la mano

$\Downarrow$

sistema esperto

Il sistema esperto

Il sistema esperto è stato realizzato tramite il *rule engine* Jess.

Il framework prevede due tipi di strategie:

- di **decisione**: selezionano la carta da giocare
- di **analisi**: stimano empiricamente il ruolo degli altri giocatori (*villani* o *soci*)

Il sistema esperto

Il sistema esperto è stato realizzato tramite il *rule engine* JESS.

Il framework prevede due tipi di strategie:

- di **decisione**: selezionano la carta da giocare
- di **analisi**: stimano empiricamente il ruolo degli altri giocatori (*villani* o *soci*)

Il sistema esperto: una regola di decisione

Se sono il *socio* e gioco subito dopo al *chiamante* (o *giaguaro*), prendo per lasciarlo ultimo la mano successiva, soprattutto nelle mani finali.

```
( defrule socio-tiene-giaguaro-ultimo
    ?w <- (calcola-giocata)
    (mio-ruolo socio)
    (giaguaro (player ?g))
    (mio-turno-numero ?n)
    (turno (player ?player&:(= ?player ?g)) (posizione ?pos
        &:(= ?n (mod (+ ?pos 1) 5) ) ) )
    (briscola (card ?b))
    (posso-prendere (card ?c&:(<> ?c ?b)))
=>
    (gioca ?c (- 100 (?c getValue)))
    (assert (ora-di-giocare))
)
```

Il sistema esperto: una regola di analisi

Se un giocatore prende lasciando ultimo il *giaguaro* la mano successiva, probabilmente è il *socio*, soprattutto se nelle ultime mani.

```
( defrule vs-socio-tiene-giaguaro-ultimo
    (not(exists(socio (player ?player))))
    (mano-numero ?mano-numero)
    (giaguaro (player ?g))
    (turno (player ?p&:(= ?p ?g)) (posizione ?pos-giaguaro))
    (giocata (player ?soc) (tipo ?t&:( or ( = ?t "taglio") (
        or (= ?t "strozzino") (= ?t "strozzo") ) ) ))
    (turno (player ?pl&:(= ?pl ?soc)) (posizione ?pos-soc&:(
        = ?pos-soc (mod (+ ?pos-giaguaro 1) 5) )))
=>
    (if (> ?mano-numero 3) then
        (bind ?new-sal 80)
    else
        (bind ?new-sal 40)
    )
    (aumenta-sal-socio ?soc ?new-sal)
)
```

Indice

Piattaforma di gioco

Requisiti:

- distribuita in rete
- giocatori umani e virtuali
- interfaccia utente esperto
- log attività

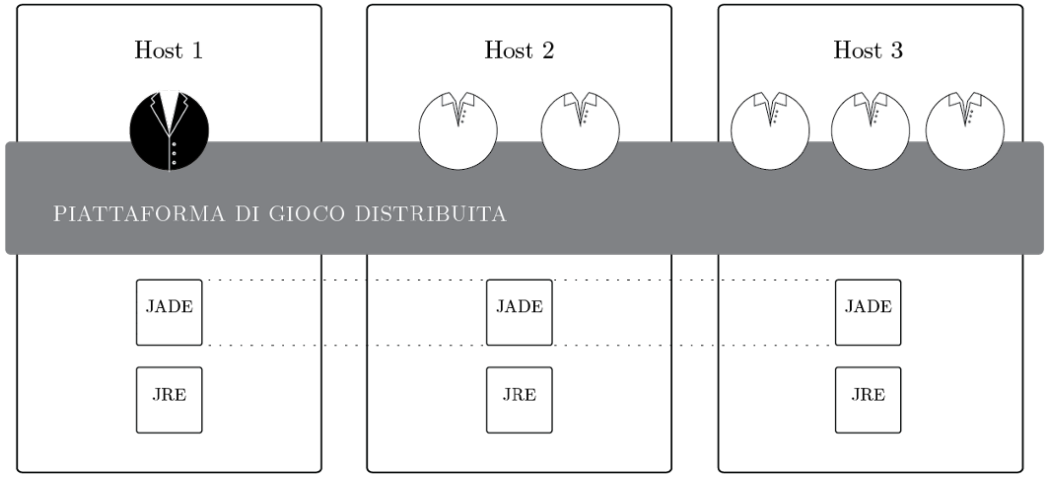
La piattaforma è stata realizzata usando il framework per lo sviluppo di piattaforme multi-agente JADE.

Prevede due tipi di agente: *mazziere* e *giocatore*.

Piattaforma di gioco

Requisiti:

- distribuita in rete
- giocatori umani e virtuali
- interfaccia utente esperto
- log attività



La piattaforma è stata realizzata usando il framework per lo sviluppo di piattaforme multi-agente JADE.

Prevede due tipi di agente: *mazziere* e *giocatore*.

L'agente Mazziere

L'agente mazziere regola e gestisce la partita. Fra i suoi compiti principali:

- "aprire un tavolo"
- farsi carico della comunicazione fra agenti (tranne chat)
- gestire la partita tramite messaggi
- redigere un file di log

L'agente Mazziere

L'agente mazziere regola e gestisce la partita. Fra i suoi compiti principali:

- "aprire un tavolo"
- farsi carico della comunicazione fra agenti (tranne chat)
- gestire la partita tramite messaggi
- redigere un file di log

L'agente Mazziere

L'agente mazziere regola e gestisce la partita. Fra i suoi compiti principali:

- "aprire un tavolo"
- farsi carico della comunicazione fra agenti (tranne chat)
- gestire la partita tramite messaggi
- redigere un file di log

L'agente Mazziere

L'agente mazziere regola e gestisce la partita. Fra i suoi compiti principali:

- "aprire un tavolo"
- farsi carico della comunicazione fra agenti (tranne chat)
- gestire la partita tramite messaggi
- redigere un file di log

L'agente Mazziere

L'agente mazziere regola e gestisce la partita. Fra i suoi compiti principali:

- ♢ "aprire un tavolo"
- ♢ farsi carico della comunicazione fra agenti (tranne chat)
- ♢ gestire la partita tramite messaggi
- ♢ redigere un file di log

Giocatore

L'agente giocatore è quello che prende parte al gioco e s'impegna a fare una mossa legale quando gli è richiesta.

Può avere modalità:

- manuale
- random
- strategia da file
  - Abbiamo creato un file contenente 30 regole di decisione e 15 di analisi, raccolte da siti e forum dedicati alla briscola in 5, per poter essere usato nella fase di sperimentazione.

Giocatore

L'agente giocatore è quello che prende parte al gioco e s'impegna a fare una mossa legale quando gli è richiesta.

Può avere modalità:

- manuale
- random
- strategia da file
  - Abbiamo creato un file contenente 30 regole di decisione e 15 di analisi, raccolte da siti e forum dedicati alla briscola in 5, per poter essere usato nella fase di sperimentazione.

Giocatore

L'agente giocatore è quello che prende parte al gioco e s'impegna a fare una mossa legale quando gli è richiesta.

Può avere modalità:

- manuale
- random
- strategia da file
  - Abbiamo creato un file contenente 30 regole di decisione e 15 di analisi, raccolte da siti e forum dedicati alla briscola in 5, per poter essere usato nella fase di sperimentazione.

Indice

Risultati

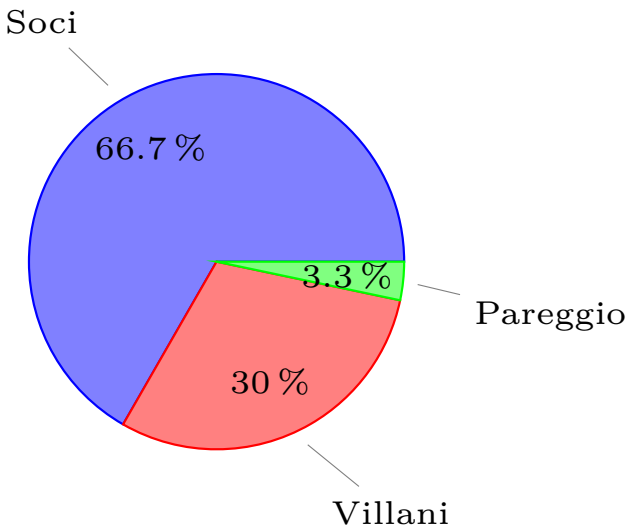
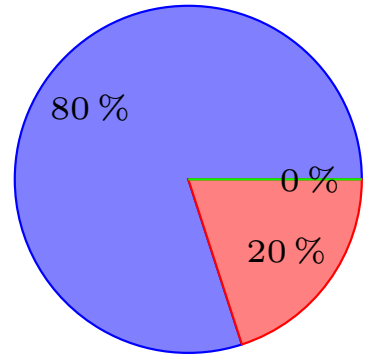
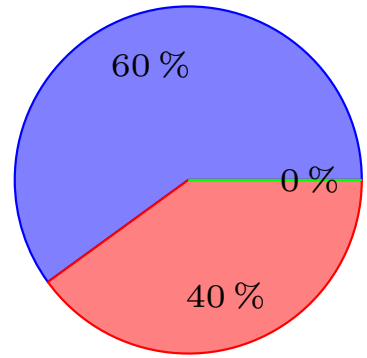


Figura: Random vs Random.
Risultati ottenuti su 30 prove.



(a) Soci a strategia, villani random



(b) villani a strategia, soci random

Figura: Una squadra a strategie contro una random. In blu le vittorie, in rosso le sconfitte.

Conclusioni

Contributi:

- ♢ Studio del gioco della briscola in 5 dal punto di vista dell'Intelligenza Artificiale
- ♢ Realizzazione di un sistema esperto (in JESS) per il giocatore virtuale
- ♢ Realizzazione di una piattaforma multigiocatore (in JADE) che supporta giocatori umani e artificiali
- ♢ Risultati sperimentali incoraggianti

Possibili sviluppi futuri

- ♢ Gestione ad hoc quando le squadre sono note.
- ♢ Utilizzo di metodi di *machine learning* per la scoperta di nuove regole di decisione.
- ♢ Integrazione di un meccanismo di *reputation* alle strategie di analisi.
- ♢ Nuovi criteri di valutazione della validità delle strategie, compreso il confronto con giocatori umani.

Grazie per l'attenzione.

Il sistema esperto: rappresentazione della conoscenza

```
(deftemplate in-mano "carte che posso ancora giocare"
   (slot card)    (slot rank)    (slot suit) )

( deftemplate giocata "info sulle giocate"
    (slot player) (slot card) (slot rank)    (slot suit)
    (slot mano)   (slot turno) (slot tipo) )

( deftemplate posso-prendere "Carte con le quali posso
    prendere la mano"
    (slot card)   (slot rank)    (slot suit)   )

( deftemplate carichi-in-mano "Le carte da punto che ho in
    mano"
    (slot card)    (slot rank)    (slot suit)    (slot
        points) )

( deftemplate giaguaro  (slot player) )
( deftemplate socio (slot player) )
( deftemplate villano   (slot player) )
( deftemplate seme-mano-fact (slot suit) )
( deftemplate prob-socio (slot player) (slot sal) )
```

Il sistema esperto: analisi delle giocate altrui

```
( defrule nuova-giocata "Ricevo una giocata: aggiorno la situa"
  ?w <- (nuova-giocata (player ?p) (card ?c) (rank ?r) (suit ?s))
  (prende (player ?prende-player) (card ?prende-card))
  ?y <- (giocata-numero ?counter-giocata)
  (seme-mano-fact (suit ?seme-mano))
  (mano-numero ?mano-numero)
=>
  (bind ?tipo "liscio")
  (if (= ?counter-giocata -1) then  ;;  Aggiorniamo chi prende
    (assert (prende (player ?p) (card ?c) (suit (?c getSuit)) (rank
        (?c getRank))))
    (assert (seme-mano-fact (suit ?s)))
    (bind ?seme-mano ?s)
  else
    (if (batte ?c ?prende-card ?seme-mano )  then
      (assert (prende (player ?p) (card ?c) (suit (?c getSuit)) (rank
          (?c getRank))))
      (if (= ?seme-mano ?s) then
        (if (< (?r getValue) 10) then  (bind ?tipo "strozzino")
        else  (bind ?tipo "strozzo") )
      else  (bind ?tipo "taglio"))
    else
      (if ( > (?r getValue) 9) then  (bind ?tipo "carico")
      else
        (if (> (?r getValue) 0) then
          (bind ?tipo "carichino")
  ))))
  (bind ?new-counter-giocata (+ ?counter-giocata 1))
  (retract ?w)  (retract ?y)
  (assert (giocata-numero ?new-counter-giocata))
  (assert (giocata (player ?p) (card ?c) (rank ?r) (suit ?s) (turno ?
      new-counter-giocata) (mano ?mano-numero) (tipo ?tipo)))
)
```

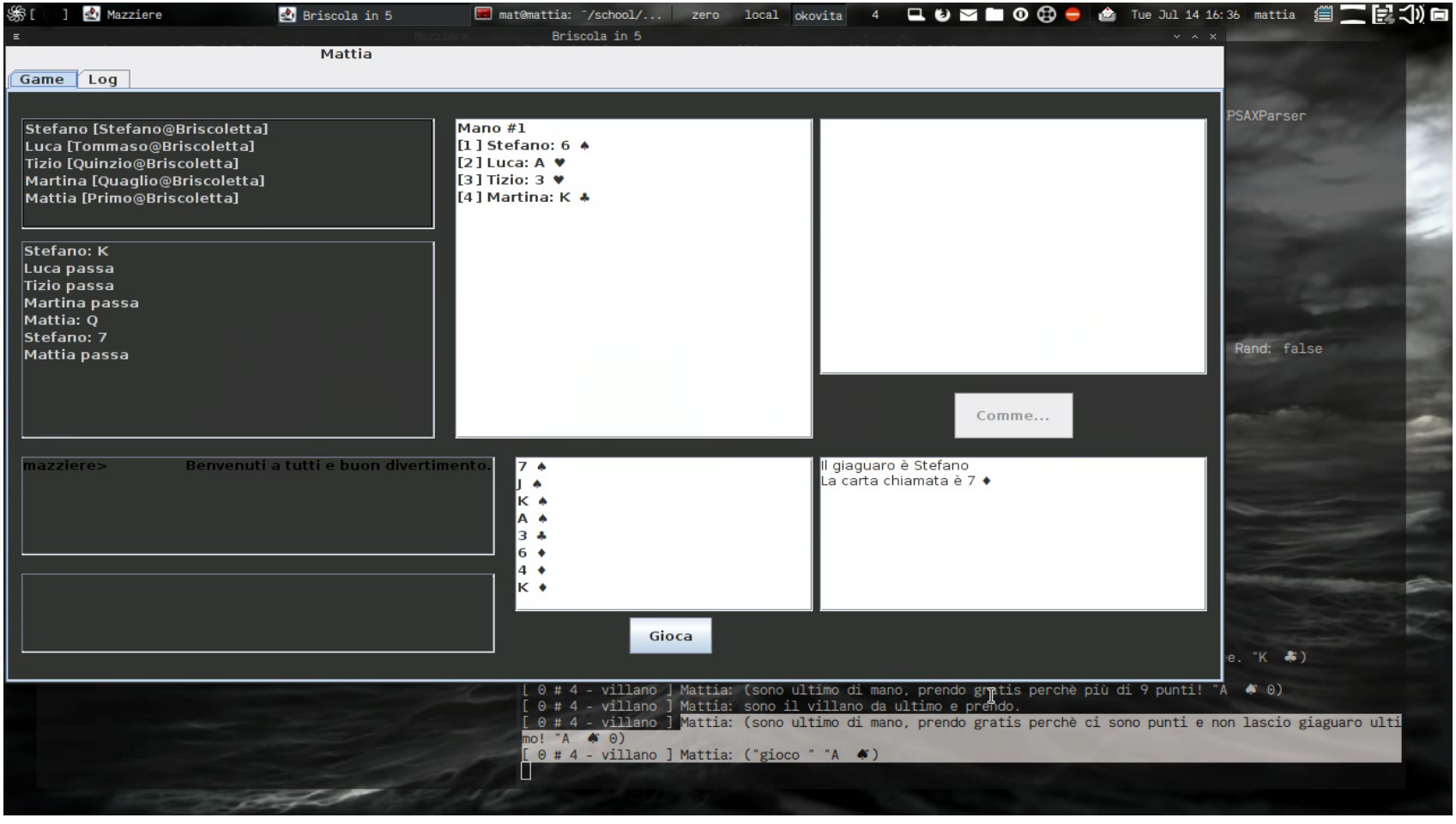
Il momento di giocare: recupero la carta

```
public Card play() throws JessException, InterruptedException {
    Card c;
    switch (strategy) {
        case RANDOM:
            c = getHand().drawRandom();
            break;
        case NORMAL:
            Value v = getRete().fetch("DA-GIOCARE");
            if (null == v) { // se nessuna regola attivata
                say("Giocando a caso");
                c = getHand().drawRandom();
            } else { // recupero la carta selezionata
                c = (Card)
                     v.javaObjectValue(getRete().getGlobalContext());
                getHand().removeCard(c);
            }
            break;
        case MANUAL:
            ((PlayerGUI) gui).beginGiocata();
            say("Aspettando la giocata manuale...");
            while (cartaDaGiocare == null) {
                Thread.sleep(1000);
            }
            c = cartaDaGiocare;
            cartaDaGiocare = null;
            break;
        default:
            say("Strategia sconosciuta", true);
            c = getHand().drawRandom();
            break;
    }
    return c;
}
```

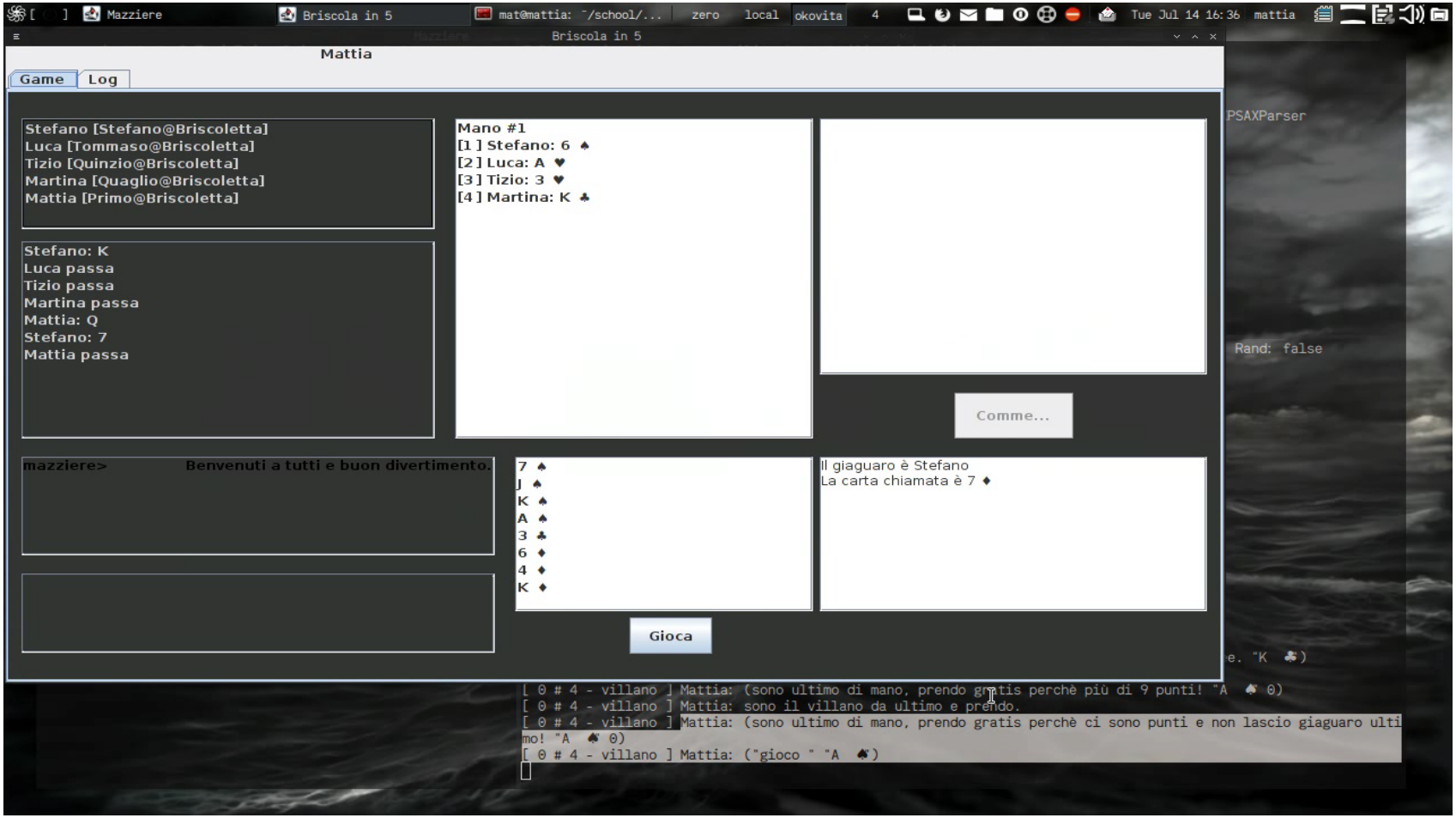
Il momento di giocare: comunico la decisione

```
/** Parte behaviour agente giocatore: seleziona carta da giocare */
class SendGiocata extends OneShotBehaviour {
    @Override
    public void action() {
        ACLMessage gMsg = new ACLMessage(ACL_TELL_GIOCATA);
        try { // do inizio al reasoning
            Fact f = new Fact("mio-turno", agent.getRete());
            agent.getRete().assertFact(f);
            agent.getRete().run();
    // recupero la carta da giocare
            Card c;
            try {
                c = agent.play();
            } catch (InterruptedException ex) {
                ex.printStackTrace();
                c = null;
            } // riazzero la carta da giocare per la prossima mano
            agent.setCardToPlay(null);
    // popolo e spedisco un messaggio di tipo GiocataMessage
            GiocataMessage g = new GiocataMessage(agent.getPlayer(), c,
                                                  status.getMano(),
                                                  status.getCounter());
            gMsg.setContentObject(g); // lo spedisco al mazziere
            gMsg.addReceiver(agent.getMazziereAID());
            myAgent.send(gMsg);
            agent.say("* Gioco" + g);
            ++lastPlayed;
        } catch (IOException | JessException ex) {
            ex.printStackTrace();
        }
        block();
}}
```

Dimostrazione



Dimostrazione



Dimostrazione (player esterno)