Intelligenza Artificiale

Realizzazione di un player per il gioco di carte UNO

Università degli studi di Padova

Anno accademico 2013/2014

Giacomo Quadrio

Matricola: 1061566

Alessandro Zonta  
Matricola:

Indice

1 Introduzione 3

2 La ricerca con avversari 4

X La scelta della libreria Swing 5

x.1 Struttura grafica dell'applicazione 5

x.2 Gestione degli eventi 8

# 1 Introduzione

Nel seguente documento verrà presentato il progetto per il corso di Intelligenza Artificiale svolto dagli studenti Quadrio Giacomo e Zonta Alessandro. Il progetto consiste nella realizzazione di un player per il gioco di carte UNO attraverso l’applicazione degli algoritmi di ricerca con avversari visti a lezione.

# 2 La ricerca con avversari

La ricerca con avversari rientra nella tipologia degli algoritmi di intelligenza artificiale che prevede un ambiente multiagente, ovvero più agenti operano all’interno dello stesso ambiente ed ognuno deve considerare, per le sue azioni, quelle degli altri. Nel progetto gli agenti che operano sono due, Min e Max, il gioco risulta essere a somma zero, ovvero il payoff totale è lo stesso per ogni istanza del gioco, e si costruirà un albero di gioco per decidere come procedere nell’analisi. Scendendo nel dettaglio abbiamo che innanzitutto non dobbiamo trovare la strategia ottima per arrivare all’obiettivo ma dobbiamo scegliere la mossa che conduce alla posizione con valore MinMax più alto ovvero la mossa che ci da il miglior vantaggio sul nostro avversario, che assumiamo giochi in maniera ottima.

# X La scelta della libreria Swing

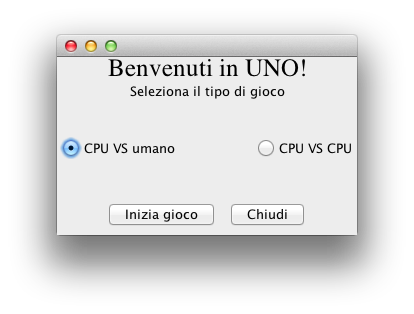
Per il progetto di intelligenza artificiale è stato utilizzato il framework Java denominato Swing per la realizzazione dell'interfaccia grafica. La scelta di utilizzare Swing è derivata dal fatto che queste librerie permettono di creare interfacce grafiche indipendentemente dalla piattaforma su cui viene eseguito il programma finale. Inoltre le diverse parti di essa sono basate su determinate interfacce collegate in modo da consentire facilmente l'aggancio di implementazioni differenti di tali interfacce. Il programmatore può quindi creare un’implementazione personalizzata oppure affidarsi alle versioni di default.

Un altro vantaggio della libreria Swing è la sua leggerezza: essa infatti non richiede di utilizzare i controlli della GUI del sistema operativo nativi, poiché disegna i suoi controlli tramite l'uso delle API 2D di Java. Infine, un componente Swing non deve avere per forza un corrispettivo nell'insieme dei componenti nativi del sistema operativo, ciò significa che è possibile realizzare componenti con libertà, sfruttando tutte le risorse messe a disposizione dalle librerie grafiche Java 2D.

Leggerezza dell'interfaccia, facilità di utilizzo, portabilità sono le caratteristiche che in definitiva ci hanno spinto ad utilizzare Swing per il nostro progetto di intelligenza artificiale.

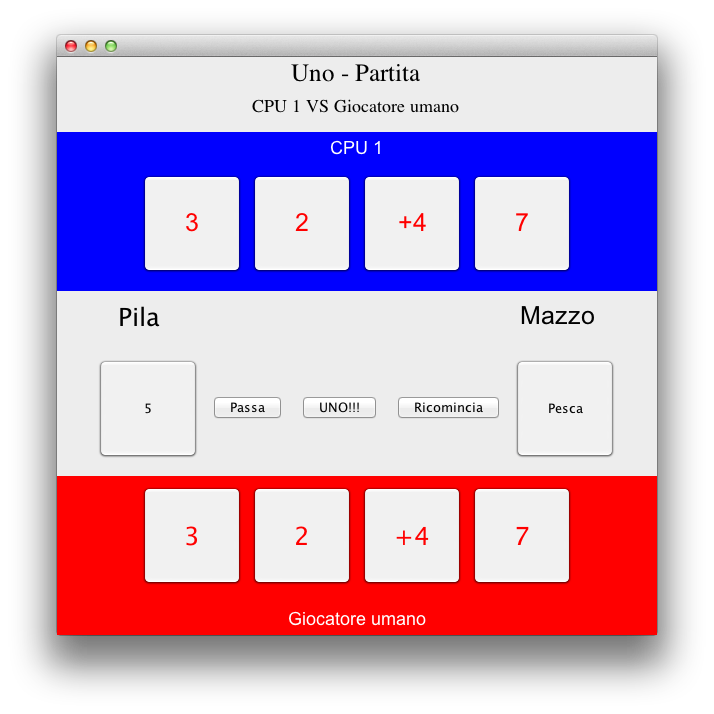
## x.1 Struttura grafica dell'applicazione

Per l'applicazione “UNO” abbiamo utilizzato una struttura grafica molto semplice, basata su di una singola finestra che, a seconda delle esigenze, mostra differenti pulsanti e informazioni per poter procedere con l'esecuzione del gioco.



*Figura x: schermata di selezione del gioco*

Nella prima schermata è possibile selezionare due differenti tipologie di gioco, partita CPU contro giocatore umano e partita CPU contro CPU. Dopo aver effettuato la scelta si può dar inizio ad una partita selezionando il pulsante “Inizia gioco”.



*Tavolo da gioco inizializzato*

Avviato il gioco, la schermata successiva che ci verrà presentata è il tafolo di gioco con presenti su schermo le carte dei due giocatori. Abbiamo deciso di mostrare le carte scoperte di entrambi i player così da offrire una migliore rappresentazione grafica della partita in corso, i due giocatori però non saranno a conoscenza di che carte ha in mano l’avversario. Il gioco è stato quindi inizializzato ed è pronto per essere avviato. Nel caso di un giocatore umano le azioni disponibili saranno quelle visualizzate nell’immagine ovvero ci potrà passare, dichiarare UNO, ricominciare una partita, pescare una carta o selezionare una delle carte che si hanno in mano per poterla lanciare. Se la carta che si seleziona non è compatibile con la carta sulla cima della pila verrà segnalato un messaggio d’errore.

**Principali informazioni sul codice della componente grafica**

Per la realizzazione vera e propria dell'interfaccia grafica si è utilizzato un Frame principale, nel quale venivano di volta in volta modificati i contenuti a seconda delle esigenze. Nel dettaglio si è creata una classe FrameP che estende la classe base Jframe :

class FrameP extends Jframe

In questa classe sono stati innanzitutto inizializzati tutti i vari componenti che verranno utilizzati nelle varie maschere e che saranno disegnati, aggiornati o spostati dalle varie funzioni a seconda dello stato in cui si trova il gioco.

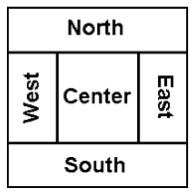
Per il layout generale della finestra si sono utilizzati principalmente tre tipologie di layout: BoxLayout, FlowLayout e BorderLayout. Per poter realizzare schermate più complesse si è inoltre scelto di adottare la tecnica dei layout innestati, di conseguenza si sono inseriti nel container principale altri container, creando quindi una struttura che ci permettesse una gestione più semplice degli elementi presenti a schermo.

Qui di seguito sono elencati brevemente i tre layout in questione:

**BoxLayout:** Permette di inserire gli elementi l’uno di seguito all’altro. Nel dettaglio abbiamo utilizzato una struttura a pila verticale nella quale si aggiungevano di volta in volta gli oggetti grafici uno sotto all’altro. Questo layout è risultato essere molto comodo per la gestione del tavolo di gioco.

**FlowLayout:** Si tratta della tipologia di layout più semplice in quanto con esso si inseriscono gli elementi orizzontalmente uno di fianco all’altro. E’ stato utilizzato per mostrare le carte che i giocatori hanno in mano.

**BorderLayout:** Questo layout è molto particolare in quanto suddivide la finestra in cinque differenti zone salienti. La zona NORD per la parte alta, la zona EAST per la parte centrale destra, la zona WEST per la parte centrale sinistra, la zona CENTER per la parte centrale e la zona SOUTH per la zona bassa. Esso è stato utilizzato per la schermata principale e le varie schermate di transizione tra un’azione e l’altra del giocatore. Di seguito potete vedere un esempio della struttura del BorderLayout.



*Figura x: le aree di BorderLayout*

## x.2 Gestione degli eventi

La gestione degli eventi, infine, è stata realizzata utilizzando le librerie AWT (Abstract Window Toolkit), per la precisione il package relativo alla gestione degli eventi denominato awt.event.

….

start.addActionListener(new StartGame());

….

class StartGame implements ActionListener {

public void actionPerformed(ActionEvent e){

JButton button = (JButton)e.getSource();

if(button.getText()=="Inizia gioco"){

if(game\_1.isSelected()){

InsertName();

start.removeActionListener(this);

}else{

if(game\_2.isSelected()){

g = new Gioco();

g.ciSonoGiocatoriUmani(false);

g.avviaIlGioco();

GameTable();

start.removeActionListener(this);

}

}

}

}

}

Al bottone adibito all'azione è stato aggiunto un ascoltatore per rilevarne la pressione. Una volta eseguita l'azione viene richiamata l'apposita classe che ne contiene le operazioni da eseguire che, in questo caso, riguardano l’avvio di una sessione di gioco tra due CPU. Più in generale, a seconda del tasto premuto l'azione eseguita sarà differente e ciò permette di poter accedere alle varie funzionalità del gioco “UNO”. Gli ascoltatori sono stati utilizzati per poter permettere varie azioni all’interno del gioco come l’avvio di una partita, la selezione della carta da giocare, il riavvio della partita ecc.