

Università degli Studi di Padova

DIPARTIMENTO DI MATEMATICA "TULLIO LEVI-CIVITA"

CORSO DI LAUREA MAGISTRALE IN INFORMATICA



Un'intelligenza artificiale per risolvere  
IQ Puzzler

*Studenti*

Mauro Carlin 1184446

Mattia Bottaro 1179480

---

ANNO ACCADEMICO 2017-2018



# Indice

<b>1</b>	<b>Introduzione</b>	<b>1</b>
1.1	Descrizione IQ Puzzler . . . . .	1
1.1.1	Griglia di gioco . . . . .	1
1.1.2	Forme di gioco . . . . .	1
1.2	Scopo del progetto . . . . .	2
1.3	PEAS . . . . .	2
1.3.1	Performance measure . . . . .	2
1.3.2	Environment . . . . .	2
1.3.3	Actuators . . . . .	2
1.3.4	Sensors . . . . .	2
1.4	Formalizzazione del problema . . . . .	2
1.5	Strumenti utilizzati . . . . .	3
1.6	Utilizzo del programma . . . . .	3
<b>2</b>	<b>Metodi di risoluzione</b>	<b>5</b>
2.1	DFS . . . . .	5
2.1.1	Descrizione . . . . .	5
2.1.2	Utilizzo . . . . .	5
2.2	CSP . . . . .	5
2.2.1	Descrizione . . . . .	5
2.2.2	Utilizzo . . . . .	5
2.3	Nostri miglioramenti . . . . .	5
<b>3</b>	<b>Performance</b>	<b>7</b>
3.1	Confronto dei vari metodi . . . . .	7
3.2	Possibili miglioramenti . . . . .	7
<b>4</b>	<b>Conclusioni</b>	<b>9</b>
<b>I</b>	<b>title</b>	<b>11</b>

## Elenco delle figure

1.1	IQ Puzzler . . . . .	<a href="#">1</a>
1.2	Esempio di esecuzione del programma . . . . .	<a href="#">3</a>

## Elenco delle tabelle

# Capitolo 1

## Introduzione

### 1.1 Descrizione IQ Puzzler

Lo scopo di IQ Puzzler è quello di riempire la griglia di gioco con tutte le varie forme a disposizione. Queste forme sono di varia dimensione, colore e struttura, e possono essere combinate in diversi modi per completare la griglia.

Il gioco comprende anche 100 configurazioni iniziali, dove alcune forme sono già posizionate all'interno della griglia e solo le rimanenti devono essere piazzate. All'aumentare delle forme rimanenti, aumenta notevolmente la complessità del gioco.

#### 1.1.1 Griglia di gioco

La griglia è formata da 11 righe con 4 o 5 celle, dove ogni cella è collegata solo con quelle nelle sue 2 diagonali. In totale la griglia è formata da 50 celle.

#### 1.1.2 Forme di gioco

Le forme totali sono 11, ognuna con una propria struttura diversa dalle altre, e sono formate da dei pallini collegati in modo da essere inseriti nelle celle della griglia.



**Figura 1.1:** IQ Puzzler

## 1.2 Scopo del progetto

Lo scopo del nostro progetto è stato quello di ricreare questo gioco ed implementare alcuni dei metodi risolutivi visti durante il corso di Intelligenza Artificiale per risolverlo. In un secondo momento abbiamo confrontato le performance dei vari algoritmi in modo da valutare la loro efficienza rispetto a questo problema.

## 1.3 PEAS

### 1.3.1 Performance measure

La misura di performance è il tempo impiegato da un algoritmo per risolvere il problema a partire da una delle configurazioni iniziali. Oltre al tempo abbiamo anche considerato il numero di azioni necessarie per raggiungere una soluzione.

### 1.3.2 Environment

Il gioco analizzato presenta un ambiente totalmente osservabile, deterministico, sequenziale, statico, discreto e single-agent. L'ambiente è completamente determinato dallo stato corrente e dall'agente che può effettuare una sola mossa alla volta. L'azione dell'agente determina le azioni future dato che il possibile cammino si riduce ad ogni passo.

### 1.3.3 Actuators

Il programma rappresenta l'esecuzione di un'azione attraverso la colorazione delle celle assegnate ad una determinata forma (passo di computazione).

### 1.3.4 Sensors

Ad ogni passo di computazione l'input della funzione agente è rappresentato dalla griglia e dalle forme rimanenti da inserire.

## 1.4 Formalizzazione del problema

- \* **Stato iniziale:** configurazione iniziale (possibilmente anche vuota) della griglia e relative forme mancanti da inserire;
- \* **Azioni possibili:** tutti i possibili modi di inserire una forma non ancora presente nella griglia;
- \* **Transition model:** colorazione, a seconda del colore della forma, della posizione scelta nella griglia;
- \* **Test obiettivo:** controllare se tutte le forme sono state inserite nella griglia, in modo da riempirla completamente;
- \* **Path cost:** tempo necessario per raggiungere la soluzione.

## 1.5 Strumenti utilizzati

Il programma è stato completamente sviluppato in Python 3.6 con l'utilizzo della libreria *pygame* per quanto riguarda la parte grafica, e la libreria *python-constraint* per la risoluzione del problema formulato come CSP.

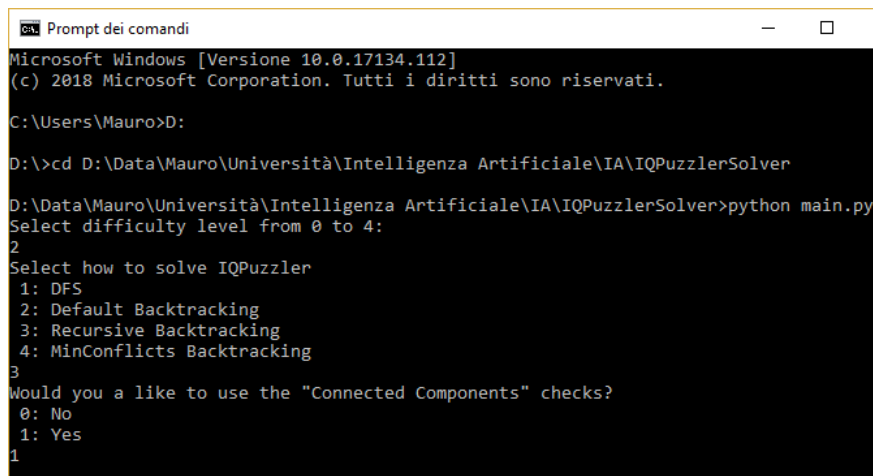
La libreria *python-constraint* è stata inoltre modificata e inclusa nella cartella *python-Constraint* per due motivi:

- \* apportare delle migliorie agli algoritmi presenti, che verranno esposte successivamente nel documento;
- \* permettere di visualizzare ogni assegnamento parziale durante l'esecuzione dell'algoritmo.

## 1.6 Utilizzo del programma

Per poter utilizzare il programma è necessario:

- \* aver installato nel computer Python 3;
- \* spostarsi con il terminale alla cartella radice (IQPuzzlerSoler) contenente il programma;
- \* eseguire uno di questi due comandi:
  - `python main.py`  
inizia l'esecuzione del programma, dando all'utente la possibilità, attraverso degli input numerici, di selezionare il problema e con quale metodo risolverlo;
  - `sh Test/scriptTest.sh`  
esegue automaticamente una serie di esempi di problemi con vari metodi di risoluzione, tracciando i risultati ottenuti nel file *Test/testResult.txt*.



```
Prompt dei comandi
Microsoft Windows [Versione 10.0.17134.112]
(c) 2018 Microsoft Corporation. Tutti i diritti sono riservati.

C:\Users\Mauro>D:

D:\>cd D:\Data\Mauro\Università\Intelligenza Artificiale\IA\IQPuzzlerSolver
D:\Data\Mauro\Università\Intelligenza Artificiale\IA\IQPuzzlerSolver>python main.py
Select difficulty level from 0 to 4:
2
Select how to solve IQPuzzler
1: DFS
2: Default Backtracking
3: Recursive Backtracking
4: MinConflicts Backtracking
3
Would you a like to use the "Connected Components" checks?
0: No
1: Yes
1
```

Figura 1.2: Esempio di esecuzione del programma





## Capitolo 2

# Metodi di risoluzione

### 2.1 DFS

#### 2.1.1 Descrizione

#### 2.1.2 Utilizzo

### 2.2 CSP

#### 2.2.1 Descrizione

Backtracking

Backtracking ricorsivo

MinConflict

#### 2.2.2 Utilizzo

### 2.3 Nostri miglioramenti



## Capitolo 3

# Performance

### 3.1 Confronto dei vari metodi

### 3.2 Possibili miglioramenti



Capitolo 4

Conclusioni



Parte I

title

