Contents

1 Funzioni euristiche **3**
Implementazione 3
Euristica 1 3
Euristica 2 4
Analisi 5
Euristica 1 5

Chapter 1

Funzioni euristiche

Abbiamo implementato e confrontato due euristiche differenti, nella formulazione delle euristiche siamo partiti da un concetto di fondo, ovvero il confronto tra lo stato finale e lo stato iniziale. Dopo aver effettuato delle ricerche abbiamo deciso di implementare le seguenti euristiche:

Euristica 1: Calcolare il numero di blocchi correnti che non sono nella posizione corretta

Euristica 2: Calcolare la differenza tra stato corrente e stato finale considerando la posizione di ogni blocco rispetto al blocco sottostante e sovrastante. Se il blocco `A` nello stato finale dovrebbe trovarsi sopra il blocco `B` e sotto il blocco `C` e nello stato corrente si trova sotto il blocco `B` e sopra il blocco `C` allora bisogna aggiungere il valore 2 all'euristica.

Implementazione

L'idea è quella di considerare il risultato della sottrazione tra l'insieme dei fatti che descrivono lo stato finale e l'insieme dei fatti che descrivono lo stato iniziale, in modo da ottenere tutti i fatti che differiscono tra i due stati.

Abbiamo considerato solo i fatti `ontable(X,Y)` e `on(X)` poiché i fatti `clear` ci danno delle informazioni superflue sulle differenze tra stato iniziale e stato finale.

Euristica 1

```
euristica(StatoAttuale, Valore) :-
    goal(StatoFinale),
    ord_subtract(StatoFinale, StatoAttuale, DifferenzaStati),
```

```
    include(is_on, DifferenzaStati, StatiOn),
    length(StatiOn, LunghezzaStatiOn),
    Valore is max(1,LunghezzaStatiOn).
```

La regola ha due parametri, la soglia attuale e una variabile in cui inseriamo il valore dell'euristica calcolato. Per prima cosa effettuiamo la sottrazione tra insiemi attraverso la funzione `ord_subtract`, successivamente selezioniamo i fatti che ci interessano (tutti tranne i `clear`) utilizzando la funzione `include`.

Abbiamo inserito inoltre due fatti che ci hanno permesso di inserire un unico parametro nella funzione `include` al fine di selezionare sia i fatti `on` che i fatti `ontable`.

```
is_on(on(_,_)).
is_on(ontable(_)).
```

Infine inseriamo il risultato in una variabile utilizzando la funzione `max`, questo per evitare che l'algoritmo ricada in un ciclo infinito nel caso in cui il valore dell'euristica fosse 0.

Euristica 2

```
euristica(StatoAttuale, Valore) :-
    goal(StatoFinale),
    ord_subtract(StatoFinale, StatoAttuale, DifferenzaStati),
    include(is_on, DifferenzaStati, StatiOn),
    include(is_ontable, DifferenzaStati, StatiOntable),
    length(StatiOn, LunghezzaStatiOn),
    length(StatiOntable, LunghezzaStatiOntable),
    ValoreOn is LunghezzaStatiOn ` 2,
    ValoreTable is LunghezzaStatiOntable,
    Valore is ValoreOn + ValoreTable.
```

L'implementazione della seconda euristica è molto simile a quella della prima. Di fatto ci è bastato incrementare di due il valore dell'euristica per ogni fatto `on` presente nell'insieme risultante dalla sottrazione dei due insiemi.

La strategia è quella di contare tutti i fatti `on` riguardanti un determinato cubo. Se nell'insieme risultante da `ordsubtract` ci fosse `on(A,B) on(C,B)` l'euristica incrementerebbe di due a causa del cubo `B`, perché si troverebbe nella posizione errata sia rispetto al cubo sovrastante sia rispetto al cubo sottostante. Successivamente incrementerebbe di uno per il cubo `A` e ancora di uno per il cubo `C`. Dal momento che ogni fatto `on` si riferisce a due cubi distinti ci basta incrementare l'euristica di due per ognuno dei fatti `on` presenti nell'insieme risultante dalla sottrazione.

Analisi

Euristica 1

we	ciao	come