



KECERDASAN BUATAN F

Heuristic

ooo

Misplaced

INFORMED SEARCH

Presented by Keluarga Berencana



Manhattan

8-puzzle

5025211129



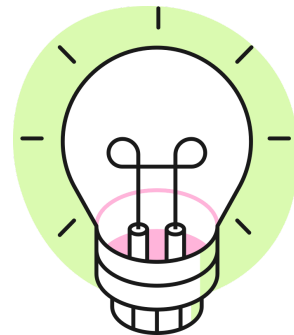
Farrela Ranku Mahhisa

5025211134



Faizah Nurdianti
Maghfirah

5025211222



Shafa Nabilah Hanin

INFORMED SEARCH

Algoritma pencarian yang memanfaatkan pengetahuan khusus atau informasi tambahan untuk memberikan petunjuk untuk solusi masalah.



HEURISTIC FUNCTION

Menjadi jalan pintas untuk memecahkan masalah yang tidak memiliki solusi yang tepat dan memakan waktu yang lama dalam mendapatkan solusi.

MANHATTAN DISTANCE

Menghitung jarak yang akan ditempuh untuk berpindah dari satu titik data ke titik lain

MISPLACED TILES

Dimana petak yang tidak pada tempatnya harus dipindahkan setidaknya satu kali untuk mengaturnya ke keadaan tujuan

MANHATTAN DISTANCE

```
57 #membuat pergerakan puzzle
58 def move(ar, p, st):
59     rh = 9999
60     store_st = st.copy()
61
62     for i in range(len(ar)):
63
64         dupl_st = st.copy()
65
66         tmp = dupl_st[p]
67         dupl_st[p] = dupl_st[arr[i]]
68         dupl_st[arr[i]] = tmp
69
70         trh = count(dupl_st)
71
72         if trh < rh:
73             rh = trh
74             store_st = dupl_st.copy()
75
76     #print(rh, store_st)
77
78     return store_st, rh
79
80
81 state = [1, 2, 3,
82         4, 5, 6,
83         0, 7, 8]
84
85 h = count(state)
86 Level = 1
87
88 print("\n----- Level "+str(Level)+" -----")
89 print_in_format(state)
90 print("\nHeuristic Value(Manhattan Distance) : "+str(h))
```

```
91
92
93 while h>0:
94     pos = int(state.index(0))
95
96     Level += 1
97
98     if pos==0:
99         arr = [1, 3]
100         state, h = move(arr, pos, state)
101     elif pos==1:
102         arr = [0, 2, 4]
103         state, h = move(arr, pos, state)
104     elif pos==2:
105         arr = [1, 5]
106         state, h = move(arr, pos, state)
107     elif pos==3:
108         arr = [0, 4, 6]
109         state, h = move(arr, pos, state)
110     elif pos==4:
111         arr = [1, 3, 5, 7]
112         state, h = move(arr, pos, state)
113     elif pos==5:
114         arr = [2, 4, 8]
115         state, h = move(arr, pos, state)
116     elif pos==6:
117         arr = [3, 7]
118         state, h = move(arr, pos, state)
119     elif pos==7:
120         arr = [4, 6, 8]
121         state, h = move(arr, pos, state)
122     elif pos==8:
123         arr = [5, 6]
124         state, h = move(arr, pos, state)
125
126     print("\n----- Level "+str(Level)+" -----")
127     print_in_format(state)
128     print("\nHeuristic Value(Manhattan Distance) : "+str(h))
```

MANHATTAN DISTANCE

```
1 # print matrix hasil akhir
2 def print_in_format(matrix):
3     for i in range(9):
4         if i%3==0 and i>0:
5             print("")
6             print(str(matrix[i])+" ", end = "")
7
8 # transfer inputan jadi bentuk matriks
9 def convert(s):
10     mat = []
11     a = []
12     b = []
13     c = []
14     for i in range(9):
15         if i<3:
16             a.append(s[i])
17         if i>=3 and i<=5:
18             b.append(s[i])
19         if i>5:
20             c.append(s[i])
21
22     mat.append(a)
23     mat.append(b)
24     mat.append(c)
25     return mat
```

```
27 # mencari hasil manhattan distance
28 def ideal_distFind(val):
29     x1 = 999
30     y1 = 999
31     ideal = [[1, 2, 3],
32             [4, 5, 6],
33             [7, 8, 0]]
34
35     for i in range(3):
36         for j in range(3):
37             if ideal[i][j]==val:
38                 x1 = i
39                 y1 = j
40                 break
41     return x1, y1
42
43 def count(initial_state):
44     inits = initial_state.copy()
45     inicon = convert(inits)
46     x1 = y1 = x2 = y2 = 999
47     total_h = 0;
48
49     for i in range(3):
50         for j in range(3):
51             x1, y1 = ideal_distFind(inicon[i][j])
52             x2, y2 = i, j
53             total_h += abs(x1-x2)+abs(y1-y2)
54
55     return total_h
56
```

OUTPUT

```
----- Level 1 -----  
1 2 3  
4 5 6  
7 0 8  
Heuristic Value(Manhattan Distance) : 2  
  
----- Level 2 -----  
1 2 3  
4 5 6  
7 8 0  
Heuristic Value(Manhattan Distance) : 0  
PS D:\SEMESTER 4\KB>
```

MISPLACED TILES

```
4 def print_in_format(matrix):
5     for i in range(9):
6         if i%3==0 and i>0:
7             print("")
8             print(str(matrix[i])+" ", end = "")
9
10 def count(s):
11     c = 0
12     ideal = [1, 2, 3,
13             4, 5, 6,
14             7, 8, 0]
15
16     for i in range(9):
17         if s[i]!=0 and s[i]!=ideal[i]:
18             c+=1
19     return c
20
21
22 def move(ar, p, st):
23     rh = 9999
24     store_st = st.copy()
25
26     for i in range(len(ar)):
27
28         dupl_st = st.copy()
29
30         tmp = dupl_st[p]
31         dupl_st[p] = dupl_st[arr[i]]
32         dupl_st[arr[i]] = tmp
33
34         trh = count(dupl_st)
35
36         if trh<rh:
37             rh = trh
38             store_st = dupl_st.copy()
39
40     return store_st, rh
```

```
43 state = [1, 2, 3,
44          0, 5, 6,
45          4, 7, 8]
46
47 h = count(state)
48 Level = 1
49
50 print("\n----- Level "+str(Level)+" -----")
51 print_in_format(state)
52 print("\nHeuristic Value(Misplaced) : "+str(h))
```


OUTPUT

```
----- Level 1 -----  
1 2 3  
0 5 6  
4 7 8  
Heuristic Value(Misplaced) : 3
```

```
----- Level 2 -----  
1 2 3  
4 5 6  
0 7 8  
Heuristic Value(Misplaced) : 2
```

```
----- Level 3 -----  
1 2 3  
4 5 6  
7 0 8  
Heuristic Value(Misplaced) : 1
```

```
----- Level 4 -----  
1 2 3  
4 5 6  
7 8 0  
Heuristic Value(Misplaced) : 0
```



KECERDASAN BUATAN F

**THANK
YOU**

KELUARGA
BERENCANA

