

Tugas 3 Kecerdasan Buatan

8 QUEEN DENGAN LOCAL SEARCH

Oleh :
Kelompok 5

KELOMPOK 5

5025211040 - Fihriz Ilham Rabbany

5025211250 - Syomeron Ansell Widjaya

5025211251 - Muhammad Ahyun Irsyada

LOCAL SEARCH

PENGERTIAN



local search adalah metode heuristik untuk memecahkan masalah optimisasi yang sulit secara komputasi. Local search dapat digunakan pada masalah yang dapat dirumuskan sebagai pencarian solusi yang memaksimalkan kriteria di antara sejumlah kandidat solusi. Algoritme local search bergerak dari solusi ke solusi dalam ruang kandidat solusi (ruang pencarian) dengan menerapkan perubahan lokal, hingga solusi yang dianggap optimal ditemukan atau batas waktu berlalu.

Algoritme local search diterapkan secara luas untuk banyak masalah komputasi yang sulit, termasuk masalah dari ilmu komputer (khususnya kecerdasan buatan), matematika, riset operasi, teknik, dan bioinformatika.

LOCAL SEARCH 8- QUEENS

SOURCE CODE



```

File Edit Selection View Go Run Terminal Help • from random import randint • Untitled-1 - Visual Studio Co File Edit Selection View Go Run Terminal Help • from random import randint • Untitled-1 - Vis
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More
$ 2-kobeni-liburan.sh $ university_survey.sh ⚡ from random import randint Untitled-1 ●
1 from random import randint
2
3 # Set the number of queens and board size
4 N = 8
5
6 # Define a function to randomly place queens on the board
7 def configure_randomly():
8     state = [randint(0, N-1) for i in range(N)]
9     return state
10
11 # Define a function to calculate the number of queens attacking each other
12 def calculate_attacking_queens(state):
13     attacking = 0
14     for i in range(N):
15         for j in range(i+1, N):
16             if state[i] == state[j] or abs(state[i]-state[j]) == abs(i-j):
17                 attacking += 1
18     return attacking
19
20 # Define a function to find the best state (with the fewest attacking queens)
21 def find_best_state():
22     state = configure_randomly()
23     attacking = calculate_attacking_queens(state)
24     while True:
25         if attacking == 0:
26             return state
27         new_state = configure_randomly()
28         new_attacking = calculate_attacking_queens(new_state)
29         if new_attacking < attacking:
30             state = new_state
31             attacking = new_attacking
32             if attacking == 0:
33                 return state
34     elif new_attacking == attacking:
35         state = new_state if randint(0, 1) == 0 else state
36
37 # Run the algorithm and print the result
38 result = find_best_state()
39 print(result)
40
41
42 for i in range(N):
43     for j in range(N):
44         if result[i] == j:
45             print("1", end=" ")
46         else:
47             print("0", end=" ")
48     print()

```

PENJELASAN



1. Pertama tama kita buat status acak (yaitu, konfigurasi acak papan).
2. Pindai semua kemungkinan tetangga dari keadaan saat ini dan lompat ke tetangga dengan nilai objektif tertinggi, jika ada. Jika tidak ada, tetangga, dengan tujuan yang lebih tinggi dari keadaan saat ini tetapi ada satu dengan yang sama maka lompat ke tetangga acak (melarikan diri dari bahu dan / atau optimal lokal).
3. Ulangi langkah 2, sampai keadaan yang tujuannya benar-benar lebih tinggi dari semua tujuan tetangganya, ditemukan dan kemudian pergi ke langkah 4.
4. Negara bagian yang ditemukan setelah pencarian lokal adalah optimal lokal atau optimal global. Tidak ada cara untuk melarikan diri dari optima lokal tetapi menambahkan tetangga acak atau restart acak setiap kali optimal lokal ditemui meningkatkan peluang untuk mencapai optimal global (solusi untuk masalah kita).
5. Output status dan kembali.



OUTPUT



```
[4, 1, 3, 5, 7, 2, 0, 6]
0 0 0 0 1 0 0 0
0 1 0 0 0 0 0 0
0 0 0 1 0 0 0 0
0 0 0 0 0 1 0 0
0 0 0 0 0 0 0 1
0 0 1 0 0 0 0 0
1 0 0 0 0 0 0 0
0 0 0 0 0 0 1 0
>
```

TERIMA KASIH

