

# **TUGAS BESAR 1**

## **Pencarian Solusi Diagonal Magic Cube dengan Local Search**

**IF3070**  
**Dasar Inteligensi Buatan**



**Disusun oleh:**

Audra Zelvania Putri Harjanto / 18222106

Rizqi Andhika Pratama / 18222118

Sekar Anindita Nurjadini / 18222125

Khayla Belva Annandira / 18222138

**Institut Teknologi Bandung**

**2024**

# Daftar Isi

<b>Daftar Isi</b>	<b>2</b>
<b>Deskripsi Persoalan</b>	<b>3</b>
<b>Pembahasan</b>	<b>4</b>
1. Pemilihan Objective Function	4
1.1 Penjelasan	4
1.2 Implementasi	8
2. Penjelasan Algoritma Local Search	11
2.1 Steepest Ascent Hill-Climbing	11
2.2 Hill-Climbing with Sideways Move	11
2.3 Random Restart Hill-Climbing	12
2.4 Stochastic Hill-Climbing	13
2.5 Simulated Annealing	14
2.6 Genetic Algorithm	15
3. Implementasi Algoritma Local Search	16
3.1 Struktur Folder	16
3.2 Hill Climbing	16
3.2.1 Steepest Ascent Hill Climbing	19
3.2.2 Sideways Move Hill Climbing	21
3.2.3 Random Restart Hill Climbing	23
3.2.4 Stochastic Hill Climbing	25
3.6 Simulated Annealing	27
3.7 Genetic Algorithm	31
4. Hasil Eksperimen	38
4.1 Hill Climbing	38
4.2.1 Steepest Ascent Hill Climbing	38
4.2.2 Hill Climbing with Sideways Move	44
4.2.3 Random Restart Hill Climbing	50
4.2.4 Stochastic Hill Climbing	56
4.2 Simulated Annealing	62
4.3 Genetic Algorithm	70
5. Analisis	108
5.1 Tingkat Kedekatan Tiap Algoritma terhadap Global Optima	108

---

5.2 Perbandingan Hasil Pencarian Tiap-Tiap Algoritma Dengan Algoritma Local Search yang Lain	108
5.3 Perbandingan Durasi Proses Pencarian Tiap Algoritma Terhadap Algoritma Lainnya	109
5.4 Konsistensi Hasil Akhir yang Didapatkan dari Tiap-Tiap Eksperimen yang Dilakukan	110
5.5 Pengaruh Jumlah Iterasi dan Jumlah Populasi Terhadap Hasil Akhir Pencarian pada Genetic Algorithm	112
6. Kesimpulan dan Saran	116
6.1 Kesimpulan	116
6.2 Saran	116
7. Pembagian Tugas Tiap Anggota Kelompok	117
8. Lampiran	118
<b>Referensi</b>	<b>119</b>

## Deskripsi Persoalan

Tugas Besar 1 ini merupakan bagian dari mata kuliah Dasar Intelektualisasi Artifisial yang bertujuan untuk memberikan pemahaman mengenai cara mengimplementasikan algoritma *local search* dalam mencari solusi permasalahan optimasi pada diagonal magic cube berukuran 5x5x5.

Diagonal magic cube merupakan kubus yang disusun dari angka-angka 1 hingga  $n^3$  pengulangan, dengan panjang sisi kubus adalah  $n$ . Setiap baris, kolom, dan tiang dari kubus tersebut harus memiliki jumlah angka yang sama dengan magic number yang ditentukan. Selain itu, jumlah angka pada diagonal ruang serta diagonal pada potongan bidang kubus juga harus sesuai dengan magic number tersebut.

Kami diharapkan untuk mengimplementasikan algoritma *local search* dengan beberapa pendekatan, yaitu Steepest Ascent Hill-climbing, Hill-climbing with Sideways Move, Random Restart Hill-climbing, Stochastic Hill-climbing, Simulated Annealing, dan Genetic Algorithm. Setiap algoritma dilakukan Pengujian dan dicatat hasilnya, termasuk *state* awal dan akhir dari kubus, nilai *objective function*, serta durasi proses pencarian. Untuk algoritma tertentu seperti Steepest Ascent Hill-Climbing dan Stochastic Hill-Climbing, perlu dicatat banyaknya iterasi hingga pencarian berhenti, sementara untuk Simulated Annealing, frekuensi kejadian 'stuck' di local optima juga perlu diamati. Khusus untuk Genetic Algorithm, dilakukan uji coba dengan variasi jumlah populasi dan iterasi untuk melihat pengaruhnya terhadap hasil pencarian.

Selain itu, hasil dari Pengujian ini perlu divisualisasikan untuk menunjukkan *state* awal dan akhir dari kubus, serta perubahan nilai *objective function* selama iterasi. Hal tersebut dilakukan untuk memberikan pengetahuan mengenai efektivitas dan efisiensi berbagai algoritma *local search*, serta faktor-faktor yang mempengaruhi kinerja pencarian solusi diagonal magic cube.

# Pembahasan

## 1. Pemilihan Objective Function

### 1.1 Penjelasan

Objective function adalah fungsi yang digunakan untuk mengevaluasi seberapa baik suatu solusi dalam sebuah masalah. Dalam konteks tugas Diagonal Magic Cube, objective function berfungsi untuk mengukur seberapa dekat susunan angka di dalam kubus  $5 \times 5 \times 5$  dengan kondisi ideal, yaitu ketika semua baris, kolom, diagonal, dan bidang memiliki jumlah angka yang sama dengan magic number. Berikut adalah cara yang kami gunakan untuk menentukan objective function dari persoalan ini:

- **Menentukan magic number.** Magic number dalam konteks magic cube adalah jumlah konstan yang harus dicapai oleh semua baris, kolom, tiang, dan diagonal dalam kubus tersebut. Magic cube terdiri dari bilangan bulat yang diatur dalam susunan  $n \times n \times n$ , dan magic number memastikan bahwa jumlah dari angka-angka di setiap dimensi kubus (baris, kolom, tiang, dan diagonal) selalu sama. Magic number pada magic cube tiga dimensi dapat dihitung dengan rumus:

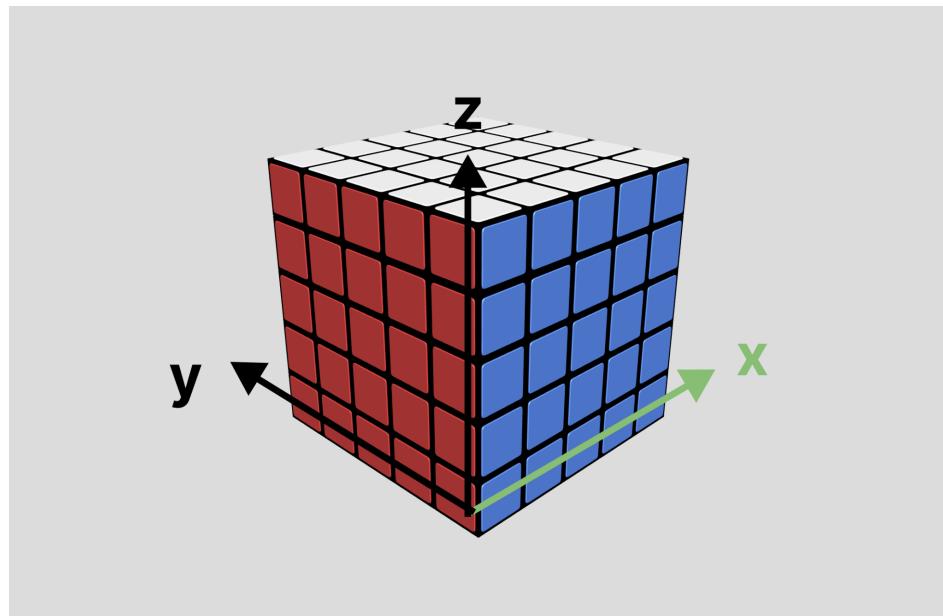
$$\text{Magic Number} = \frac{n(n^3 + 1)}{2}$$

Kemudian, pada tugas ini dipaparkan sebuah magic cube  $5 \times 5 \times 5$  yang memiliki arti bahwa nilai  $n$  adalah 5. Oleh karena itu, kita dapat menyimpulkan bahwa magic number dari magic cube  $5 \times 5 \times 5$  ini adalah sebagai berikut:

$$\text{Magic Number} = \frac{5(5^3 + 1)}{2} = 315$$

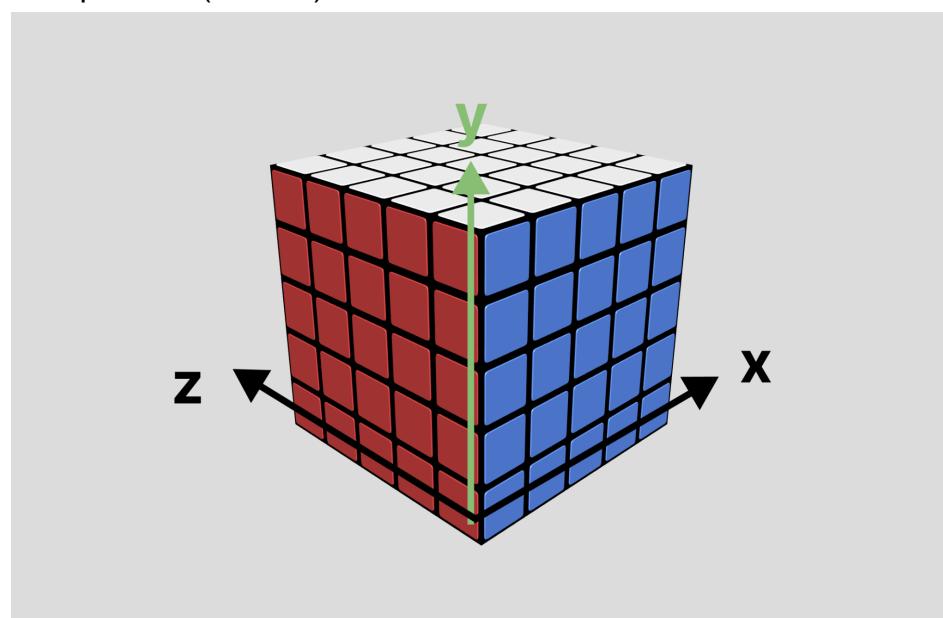
Kemudian, kita dapat merumuskan objective function dengan magic number yang sudah kita dapatkan dengan memperhatikan syarat-syarat dari magic cube.

- **Evaluasi penjumlahan angka-angka pada setiap bagian dari syarat magic cube.** Objective function akan menghitung jumlah angka-angka pada:
  - ❖ Setiap baris (horizontal)



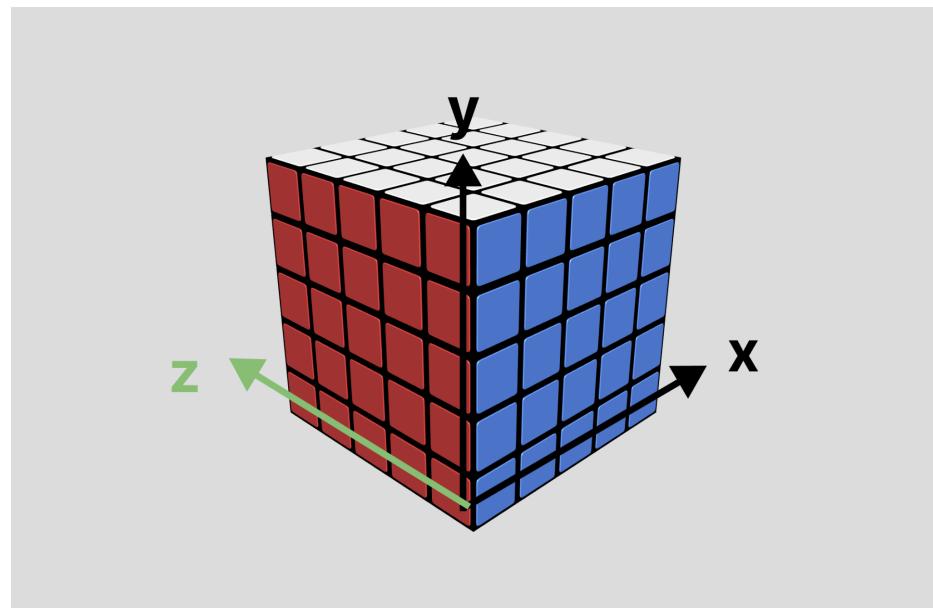
Gambar 1.1. Baris pada Cube (sumbu-x)

- ❖ Setiap kolom (vertikal)



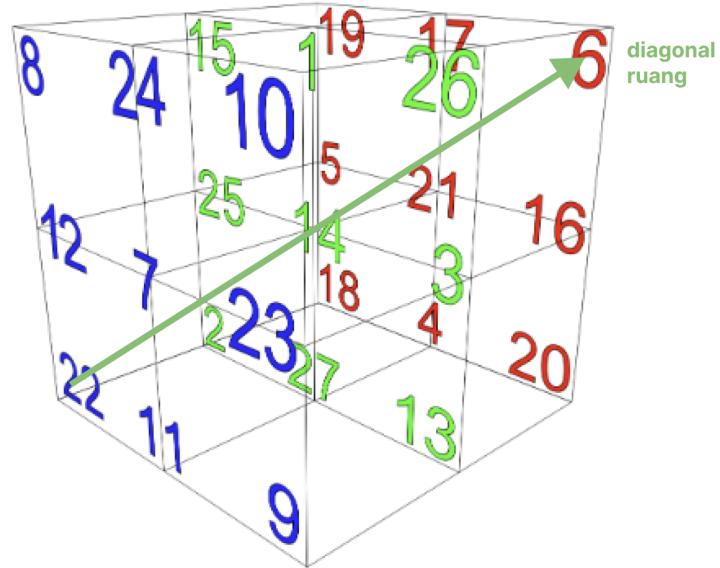
Gambar 1.2. Kolom pada Cube (sumbu-y)

- ❖ Setiap tiang (z-axis)



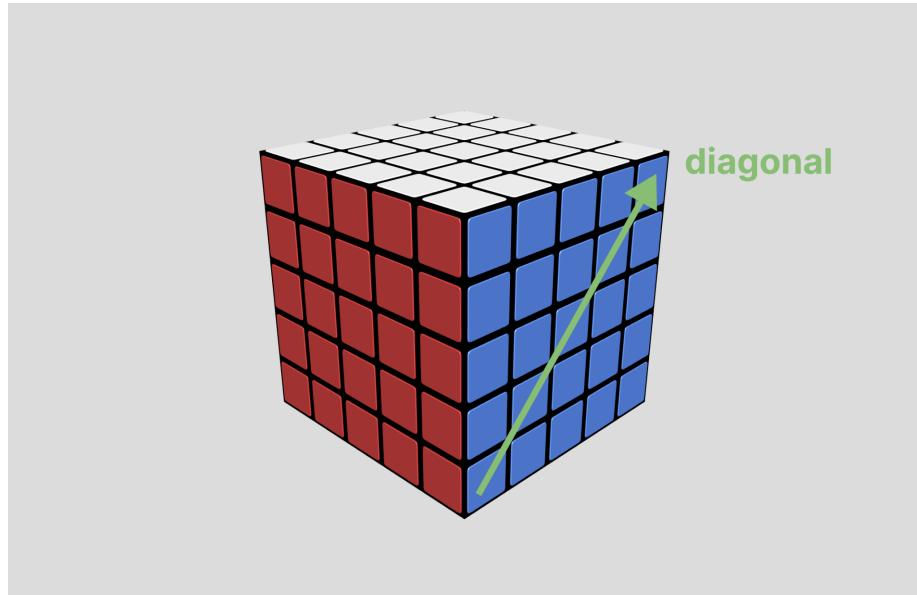
Gambar 1.3. Tiang pada Cube (sumbu-z)

- ❖ Seluruh diagonal ruang (melintasi kubus 3D)



Gambar 1.4. Diagonal Ruang pada Cube

- ❖ Seluruh diagonal pada suatu potongan bidang dari kubus



Gambar 1.5. Diagonal bidang pada Cube

- **Menghitung selisih.** Setelah menghitung jumlah setiap bagian, objective function akan membandingkan jumlah tersebut dengan magic number. Jika jumlahnya benar, maka selisihnya 0. Jika tidak benar, objective function akan menghitung seberapa besar perbedaannya. Berdasarkan pembahasan ini, objective function dapat dirumuskan sebagai berikut:

$$f(s) = \sum |M - S_{row}| + \sum |M - S_{col}| + \sum |M - S_{pillar}| + \sum |M - S_{diagonal}| + \sum |M - S_{slice}|$$

dengan:

- ❖  $f(s)$  adalah nilai dari objective function untuk state  $s$
- ❖  $M$  adalah magic number dengan nilai 315
- ❖  $S_{row}$ ,  $S_{col}$ ,  $S_{pillar}$ ,  $S_{diagonal}$ , dan  $S_{slice}$  masing-masing adalah jumlah dari angka-angka pada baris, kolom, tiang, diagonal ruang, dan potongan bidang.

Objective function ini akan memberikan nilai error yang menunjukkan seberapa jauh kubus tersebut dari kondisi magic cube ideal. Semakin kecil nilai objective function, semakin dekat state dengan kondisi yang diinginkan. Jika nilai objective function mencapai 0, berarti state yang ditemukan sudah benar, di mana setiap baris, kolom, diagonal, dan

bidang pada magic cube menjumlahkan angka-angka yang sama dengan magic number.

## 1.2 Implementasi

Implementasi dari *objective function* pada masalah Diagonal Magic Cube 5x5x5 bertujuan untuk mengukur sejauh mana konfigurasi angka-angka dalam kubus tersebut mendekati kondisi ideal dari sebuah magic cube.

Untuk penerapan *objective function* terdapat pada file cube.py

**Tabel 1.2.1.** Fungsi Objective Magic Cube

Nama Fungsi	objective_function(self)
Deskripsi Fungsi	Fungsi ini berguna untuk menghitung nilai objektif dari sebuah cube dan melihat seberapa jauh state cube dengan state ideal (nilai objektif = 0)
Kode	<pre>def objective_function(self):     # Mengambil magic number     magic_number = self.magic_number     total_difference = 0 # Nilai objektif     paling optimal      # Perbedaan jumlah di setiap baris     for layer in self.cube:         for row in layer:             row_sum = np.sum(row)             total_difference += abs(magic_number - row_sum)      # Perbedaan jumlah di setiap kolom     for layer in self.cube:         for col in range(self.n):             col_sum = np.sum(layer[:, col])             total_difference += abs(magic_number - col_sum)</pre>

```
# Perbedaan jumlah di setiap tiang  
(melintasi layer)  
    for row in range(self.n):  
        for col in range(self.n):  
            pillar_sum =  
np.sum(self.cube[:, row, col])  
            total_difference +=  
abs(magic_number - pillar_sum)  
  
# Perbedaan jumlah di setiap diagonal  
ruang  
diagonals = [  
    np.sum([self.cube[i, i, i] for i  
in range(self.n)]),  
    np.sum([self.cube[i, i, self.n - i  
- 1] for i in range(self.n)]),  
    np.sum([self.cube[i, self.n - i -  
1, i] for i in range(self.n)]),  
    np.sum([self.cube[i, self.n - i -  
1, self.n - i - 1] for i in range(self.n)]),  
]  
    total_difference +=  
sum(abs(magic_number - diag_sum) for diag_sum  
in diagonals)  
  
# Perbedaan jumlah di setiap diagonal  
bidang  
    for layer in self.cube:  
        diagonal1 = np.sum([layer[i, i]  
for i in range(self.n)]) # Diagonal dari kiri  
atas ke kanan bawah di setiap layer  
        diagonal2 = np.sum([layer[i,  
self.n - i - 1] for i in range(self.n)]) #  
Diagonal dari kanan atas ke kiri bawah di  
setiap layer  
        total_difference +=
```

```
abs(magic_number - diagonal1)
        total_difference +=
abs(magic_number - diagonal2)

        for row in range(self.n):
            diagonal1 = np.sum([self(cube[row,
i, i] for i in range(self.n))]) # Diagonal
dari kiri atas ke kanan bawah di setiap baris
(3D)
            diagonal2 = np.sum([self(cube[row,
i, self.n - i - 1] for i in range(self.n))]) # Diagonal dari kanan atas ke kiri bawah di
setiap baris (3D)
            total_difference +=
abs(magic_number - diagonal1)
            total_difference +=
abs(magic_number - diagonal2)

        for col in range(self.n):
            diagonal1 = np.sum([self(cube[i,
i, col] for i in range(self.n))]) # Diagonal
dari kiri atas ke kanan bawah di setiap kolom
(3D)
            diagonal2 = np.sum([self(cube[i,
self.n - i - 1, col] for i in range(self.n))]) # Diagonal dari kanan atas ke kiri bawah di
setiap kolom (3D)
            total_difference +=
abs(magic_number - diagonal1)
            total_difference +=
abs(magic_number - diagonal2)

    return total_difference
```

## 2. Penjelasan Algoritma Local Search

### 2.1 Steepest Ascent Hill-Climbing

Steepest ascent hill climbing adalah salah satu metode optimasi berbasis pencarian lokal yang bertujuan untuk menemukan solusi terbaik dengan memperbaiki current state secara iteratif. Dalam konteks Magic Cube 5x5x5, algoritma ini diawali dengan initial state acak di mana jumlah angka pada beberapa baris, kolom, dan diagonal mungkin jauh dari magic number (315). Pada setiap iterasi, algoritma mengidentifikasi semua neighbor dari current state yang dihasilkan dari kemungkinan pertukaran dua angka dalam kubus. Setiap neighbor kemudian dievaluasi menggunakan objective function yang mengukur seberapa dekat konfigurasi tersebut dengan kondisi magic cube yang ideal. Algoritma memilih neighbor dengan nilai objective function terendah, yaitu yang paling mendekati 0, yang menandakan konfigurasi lebih optimal dibandingkan current state.

Proses ini berlanjut secara iteratif, di mana current state diperbarui dengan neighbor terbaik dari iterasi sebelumnya, dan algoritma kembali mencari neighbor yang memberikan penurunan nilai objective function terbesar. Algoritma berhenti ketika tidak ada lagi neighbor yang menawarkan hasil yang lebih baik dari current state saat ini, menghasilkan state terbaik yang ditemukan dalam proses tersebut. Namun, steepest ascent hill climbing memiliki keterbatasan dalam menjamin solusi global maximum karena algoritma ini dapat berhenti pada local maximum, yaitu konfigurasi yang tidak dapat diperbaiki lebih lanjut tetapi masih jauh dari konfigurasi ideal (global maximum). Misalnya, algoritma mungkin berhenti dengan nilai objective function 50, yang menunjukkan state optimal lokal tetapi bukan solusi ideal yang bernilai 0.

### 2.2 Hill-Climbing with Sideways Move

Sideways move hill climbing adalah varian dari hill climbing yang memperkenankan gerakan lateral ketika tidak ada neighbor yang secara langsung meningkatkan kualitas solusi, yaitu menurunkan nilai objective function. Dalam penerapannya pada masalah Magic Cube 5x5x5, algoritma dimulai dari initial state acak di mana jumlah angka pada beberapa baris, kolom, dan diagonal dapat sangat berbeda dari magic number (315). Seperti pada Steepest Ascent Hill Climbing, algoritma mengevaluasi semua

neighbor dari current state yang diperoleh dari kemungkinan pertukaran dua angka dalam kubus. Setiap neighbor dinilai berdasarkan nilai objective function untuk melihat sejauh mana state tersebut mendekati solusi ideal. Jika ada neighbor dengan nilai objective function lebih rendah dari current state, algoritma akan berpindah ke neighbor tersebut.

Namun, yang membedakan sideways move hill climbing adalah kemampuannya untuk melakukan gerakan lateral atau "sideways move" ketika tidak ada neighbor yang memperbaiki nilai objective function tetapi ada yang memiliki nilai yang sama. Dengan melakukan gerakan sideways, algoritma mencoba menghindari local maximum dengan eksplorasi yang lebih luas, berharap menemukan jalan keluar menuju solusi yang lebih baik. Misalnya, jika current state memiliki nilai objective function 430 dan tidak ada neighbor yang memiliki nilai lebih rendah, tetapi ada neighbor dengan nilai yang sama, algoritma akan memilih salah satu neighbor dengan nilai 430. Proses ini berlanjut hingga algoritma menemukan neighbor dengan nilai yang lebih kecil, misalnya 400, atau berhenti ketika tidak ada perbaikan lebih lanjut. Meskipun sideways moves meningkatkan peluang keluar dari local maximum sementara, algoritma ini masih memiliki risiko berhenti pada local maximum tanpa mencapai solusi optimal (global maximum) karena bergantung pada eksplorasi terbatas yang mungkin tidak selalu menemukan state dengan nilai objective function 0.

### 2.3 Random Restart Hill-Climbing

Random restart hill climbing adalah metode optimasi yang memperluas algoritma hill climbing untuk meningkatkan kemungkinan menemukan solusi optimal dalam ruang pencarian yang kompleks, seperti dalam permasalahan Magic Cube 5x5x5. Algoritma ini dimulai dengan initial state acak di mana jumlah angka pada baris, kolom, dan diagonal mungkin jauh dari nilai ideal (magic number 315). Pada setiap iterasi, algoritma memeriksa semua neighbor dari current state yang dihasilkan dari pertukaran dua angka pada kubus, lalu mengevaluasi setiap neighbor berdasarkan nilai objective function  $f(s)$ . Jika ada neighbor dengan nilai objective function yang lebih rendah, algoritma bergerak ke neighbor tersebut sebagai current state baru. Jika tidak ada perbaikan langsung, algoritma dapat melakukan sideways move dengan memilih neighbor yang memiliki nilai objective function sama dengan current state saat ini.

Namun, berbeda dengan steepest ascent hill climbing, ketika algoritma terjebak pada local maximum (di mana tidak ada neighbor yang memiliki nilai objective lebih kecil), algoritma ini melakukan random restart, yaitu memulai kembali dari posisi acak baru. Proses ini memberi kesempatan untuk mengeksplorasi bagian lain dari ruang pencarian yang mungkin mengarah pada solusi yang lebih baik. Berdasarkan probabilitas keberhasilan  $p$  dalam menemukan solusi yang lebih baik setelah satu kali pencarian, jumlah restart yang diharapkan dihitung menggunakan rumus  $Expected \ nb \ of \ restarts = \frac{1}{p}$ , sementara jumlah langkah yang diharapkan untuk mencapai solusi lebih baik dihitung dengan  $Expected \ nb \ of \ steps = s + \frac{f(1-p)}{p} = 100 + \frac{50(1-0.2)}{0.2} = 300$ , dengan  $s$  sebagai langkah pencarian sebelum mencapai local maximum dan  $f$  sebagai langkah tambahan setelah restart. Dengan metode ini, algoritma dapat mencoba lebih banyak kemungkinan konfigurasi state, sehingga meningkatkan peluang untuk menemukan global maximum, meskipun tetap tidak menjamin tercapainya nilai objective function 0.

## 2.4 Stochastic Hill-Climbing

Stochastic hill climbing adalah metode optimasi yang berbeda dari steepest ascent hill climbing karena tidak mengevaluasi semua neighbor dari current state. Dalam algoritma ini, hanya satu neighbor yang dipilih secara acak pada setiap iterasi untuk dievaluasi berdasarkan nilai objective function  $f(s)$ , yang mengukur seberapa dekat konfigurasi tersebut dengan solusi ideal. Jika neighbor yang dipilih memiliki nilai objective function yang lebih baik dibandingkan current state, maka algoritma memperbarui current state ke neighbor tersebut. Pendekatan ini memungkinkan variasi dalam proses pencarian dan dapat membantu algoritma untuk keluar dari local maximum yang mungkin sulit dijangkau oleh metode hill climbing lain yang lebih deterministik.

Penggunaan pemilihan neighbor secara acak membuat stochastic hill climbing lebih fleksibel dalam mengeksplorasi ruang pencarian dan memberikan peluang yang lebih besar untuk menemukan solusi alternatif. Namun, algoritma ini juga memiliki batasan, karena tanpa seleksi yang sistematis, algoritma bisa terjebak pada local maximum dan pencarian menjadi kurang terarah. Untuk mencegah pencarian yang berkepanjangan tanpa perbaikan yang signifikan, batas jumlah iterasi dapat ditetapkan,

misalnya berhenti setelah  $n_{max}$  kali Pengujian tanpa peningkatan. Meskipun pendekatan acak ini tidak selalu menjamin solusi optimal, stochastic hill climbing dapat menjelajahi konfigurasi yang lebih luas dalam ruang pencarian, yang pada beberapa kasus memberikan peluang lebih tinggi untuk mendekati kondisi optimal daripada algoritma hill climbing konvensional yang hanya mencari solusi terbaik dari seluruh neighbor yang ada.

## 2.5 Simulated Annealing

Simulated annealing adalah metode optimasi yang memperkenalkan elemen probabilistik dalam proses pencarian solusi, bertujuan untuk menghindari jebakan local maximum dan menemukan solusi optimal atau mendekati optimal. Pada masalah Magic Cube 5x5x5, algoritma ini dimulai dari initial state acak, di mana susunan angka pada beberapa baris, kolom, dan diagonal mungkin jauh dari magic number (315). Setiap iterasi melibatkan pemilihan neighbor dari current state, yang diperoleh melalui pertukaran dua angka dalam kubus. Namun, tidak seperti algoritma hill climbing lainnya, simulated annealing memungkinkan pilihan neighbor dengan nilai objective function yang lebih buruk dari current state dengan probabilitas tertentu, yang dikendalikan oleh parameter "temperatur". Di awal pencarian, dengan temperatur yang tinggi, algoritma cenderung lebih fleksibel, sehingga menerima state yang lebih buruk untuk mendorong eksplorasi ruang pencarian yang lebih luas.

Seiring berjalannya iterasi, temperatur akan diturunkan secara bertahap, yang mengurangi probabilitas algoritma menerima neighbor dengan nilai objective function yang lebih tinggi dari current state. Penurunan temperatur ini membuat algoritma semakin selektif dalam memilih neighbor, berfokus hanya pada neighbor yang memperbaiki current state seiring mendekati solusi optimal. Proses ini berlanjut hingga temperatur mencapai batas minimum, iterasi maksimum tercapai, atau solusi optimal (global maximum) ditemukan. Pendekatan ini memungkinkan algoritma untuk menjelajahi lebih banyak alternatif di awal pencarian dan secara bertahap memfokuskan pencarian pada solusi yang lebih baik. Dengan simulasi penurunan temperatur yang tepat, simulated annealing memiliki potensi yang lebih besar dalam mencapai state mendekati global maximum dibandingkan metode optimasi lokal lain yang tidak memiliki mekanisme untuk menerima solusi sementara yang lebih buruk.

## 2.6 Genetic Algorithm

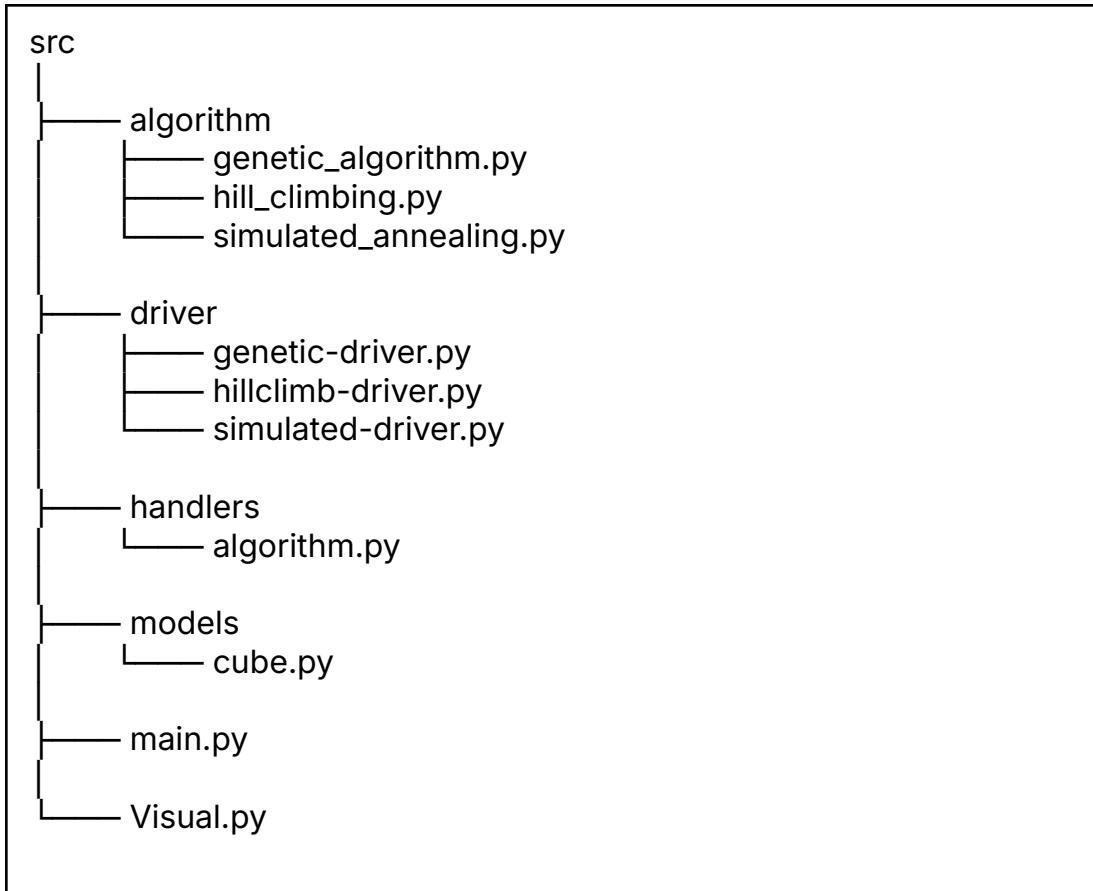
Genetic Algorithm (GA) merupakan algoritma optimasi yang terinspirasi oleh proses evolusi biologis, di mana solusi dalam bentuk kromosom mengalami seleksi, crossover, dan mutasi untuk mencapai solusi optimal atau mendekati optimal. Dalam penerapannya pada permasalahan Magic Cube 5x5x5, GA diawali dengan pembentukan initial state dalam bentuk deretan kromosom acak, misalnya dua kromosom ( $K=2$ ), yang masing-masing merepresentasikan susunan angka 0 hingga 125, menggambarkan seluruh elemen dalam Magic Cube yang telah didekomposisi. Setiap kromosom dievaluasi menggunakan objective function untuk menilai seberapa dekat susunan tersebut dengan solusi optimal. Nilai objective function yang lebih tinggi mengindikasikan susunan kromosom yang kurang sesuai, dan sebaliknya. Berdasarkan hasil objective function, persentase probabilitas seleksi dihitung, yang memungkinkan kromosom dengan nilai terbaik memiliki peluang lebih besar untuk berpartisipasi dalam tahap crossover.

Setelah seleksi, tahap crossover dilakukan dengan menentukan satu titik crossover secara acak untuk menukar bagian dari kromosom sehingga tercipta kombinasi karakteristik dari kedua kromosom induk. Mekanisme mutasi diterapkan secara acak untuk mencegah hasil konvergensi terlalu cepat dan meningkatkan keragaman solusi. Mutasi ini dilakukan dengan mengganti elemen tertentu pada kromosom dengan angka baru secara acak, lalu nilai objective function dihitung kembali untuk kedua kromosom hasil crossover. Proses ini berulang hingga mencapai solusi global optimum atau iterasi maksimal. Namun, dalam konteks Magic Cube, GA memiliki keterbatasan. Constraint dari Magic Cube, seperti susunan angka unik dan properti diagonal yang harus terpenuhi, membuat peluang menemukan solusi yang valid melalui kombinasi kromosom sangat kecil. Selain itu, GA memerlukan waktu iterasi yang panjang, menjadikannya kurang efisien untuk menyelesaikan masalah Magic Cube secara langsung.

### 3. Implementasi Algoritma Local Search

#### 3.1 Struktur Folder

Berikut adalah struktur folder yang diterapkan pada keseluruhan program Magic Cube Solver kami.



#### 3.2 Hill Climbing

Untuk penerapan genetic algorithm terdapat pada file hill\_climbing.py

**Tabel 3.2.1.** Fungsi Get Neighbor Hill Climbing

Nama Fungsi	get_neighbor(self, cube, current_score)
Deskripsi Fungsi	Fungsi ini melakukan pencarian neighbor terbaik dari cube dengan melakukan berbagai kemungkinan



	<pre> new_score     best_swap = (i, j,   k, x, y, z)             # Mengembalikan kubus   ke keadaan semula           cube(cube[i, j, k],   cube(cube[x, y, z] = cube(cube[x, y, z], cube(cube[i,   j, k]             # Jika ditemukan swap terbaik, maka dilakukan   swap tersebut pada kubus           if best_swap:               i, j, k, x, y, z = best_swap               cube(cube[i, j, k], cube(cube[x, y, z] =   cube(cube[x, y, z], cube(cube[i, j, k]           else:               cube = None             # Mengembalikan kubus yang telah di-swap   return cube </pre>
--	--

**Tabel 3.2.2.** Fungsi Get All Neighbors Hill Climbing

Nama Fungsi	get_all_neighbors(self, cube)
Deskripsi Fungsi	Fungsi ini menghasilkan semua kemungkinan neighbor dari suatu cube. Neighbor dihasilkan dengan melakukan swap antar elemen cube. Fungsi mengembalikan list yang mencakup semua kemungkinan neighbor yang dapat dihasilkan dari initial cube.
Kode	<pre> def get_all_neighbors(self, cube):     # Inisialisasi variabel     n = cube.n     neighbors = []      # Looping untuk semua kemungkinan swap pada     setiap state </pre>

```
for i in range(n):
    for j in range(n):
        for k in range(n):
            for x in range(i, n):
                for y in range(j if x == i
else 0, n):
                    for z in range(k + 1 if x
== i and y == j else 0, n):
                        # Melakukan swap pada
                        kubus
                        cube(cube[i, j, k],
cube(cube[x, y, z] = cube(cube[x, y, z], cube(cube[i,
j, k]

# Membuat kubus baru
new_cube = Cube()
new_cube(cube =
np.copy(cube(cube)

# Menambahkan kubus
yang telah di-swap ke dalam list neighbors

neighbors.append(new_cube)

# Mengembalikan kubus
ke keadaan semula
cube(cube[i, j, k],
cube(cube[x, y, z] = cube(cube[x, y, z], cube(cube[i,
j, k]

# Mengembalikan semua neighbors
return neighbors
```

### 3.2.1 Steepest Ascent Hill Climbing

Untuk penerapan Steepest Ascent terdapat pada file hill\_climbing.py

**Tabel 3.2.1.1.** Fungsi Steepest Ascent Hill Climbing

Nama Fungsi	steepest_ascent(self, cube, max_iterations)
Deskripsi Fungsi	Fungsi ini merupakan fungsi utama yang melakukan pencarian dengan algoritma Steepest Ascent Hill Climbing. Pada setiap iterasi, fungsi ini memanggil get_neighbor untuk mencari neighbor terbaik, yang kemudian dijadikan sebagai current state. Fungsi akan mengembalikan cube terbaik, nilai objective terbaik, jumlah iterasi, dan semua objective function.
Kode	<pre>def steepest_ascent(self, cube, max_iterations):     # Inisialisasi variabel untuk     menyimpan kubus saat ini dan objective     function saat ini     current_state = cube     current_score =     current_state.objective_function()     iterations = 0     scores = []     display_text = []      # Looping hingga mencapai maksimum     iterasi     while iterations &lt; max_iterations:          # Menyimpan skor pada iterasi saat         ini         scores.append(current_score)          # Mendapatkan neighbor terbaik         dari kubus saat ini         neighbor =         self.get_neighbor(current_state,         current_score)          # Jika tidak ditemukan neighbor         terbaik, maka berhenti         if neighbor is None:</pre>

```

        break # Break loop ketika
tidak ada neighbor yang lebih baik
# Jika ditemukan neighbor terbaik,
maka kubus saat ini diganti dengan neighbor
terbaik

else:
    current_state = neighbor
    current_score =
current_state.objective_function()

    display_text =
Visual.display_iteration(display_text,
iterations, current_score, "steepest_ascent")

    iterations += 1

    # Mengembalikan kubus terbaik, nilai
objektif terbaik, jumlah iterasi, dan semua
objektif function
    return current_state, current_score,
len(scores), scores

```

### 3.2.2 Sideways Move Hill Climbing

Untuk penerapan Sideways Move terdapat pada file hill\_climbing.py

**Tabel 3.2.2.1.** Fungsi Sideways Move Hill Climbing

Nama Fungsi	sideways_move(self, cube, max_sideways)
Deskripsi Fungsi	Fungsi ini merupakan fungsi utama untuk melakukan searching menggunakan hill climbing with sideways move algorithm. Fungsi ini akan menerima input cube dan max_sideways dan memanggil get_neighbor untuk kemudian mengevaluasi objective function. Fungsi ini akan mengembalikan cube terbaik, nilai objective terbaik, jumlah iterasi, dan semua objective function.

**Kode**

```
def sideways_move(self, cube, max_sideways):
    # Inisialisasi variabel untuk menyimpan kubus
    saat ini dan objective function saat ini
    current_state = cube
    current_score =
    current_state.objective_function()
    scores = []
    display_text = []
    iterations = 0
    sideways_count = 0

    # Looping hingga ditemukan solusi optimal
    while True:

        # Menyimpan skor pada iterasi saat ini
        scores.append(current_score)

        # Mendapatkan neighbor terbaik dari kubus
        saat ini
        neighbor =
        self.get_neighbor(current_state, current_score)

        # Mengambil neighbor dan menghitung skor
        dari neighbor terbaik
        next_state = neighbor
        next_score =
        next_state.objective_function()

        # Jika skor dari neighbor lebih baik dari
        skor saat ini, maka kubus saat ini diganti dengan
        neighbor terbaik
        if next_score < current_score:
            current_state = next_state
            current_score = next_score
            sideways_count = 0
        # Jika skor dari neighbor sama dengan skor
        saat ini, maka kubus saat ini diganti dengan neighbor
        terbaik
        else:
```

```
        # Jika skor dari neighbor sama dengan
        # skor saat ini, maka kubus saat ini diganti dengan
        # neighbor terbaik

        if next_score == current_score:
            current_state = next_state
            current_score = next_score
            sideways_count += 1 # Menambahkan
            jumlah sideways

            # Jika skor dari neighbor lebih buruk
            # dari skor saat ini, maka berhenti
            else:
                break

        display_text =
Visual.display_iteration(display_text, iterations,
current_score, "sideways_move")

iterations += 1

        # Berhenti jika jumlah sideways lebih dari
        # maksimum yang ditentukan
        if sideways_count >= max_sideways:
            print("Reached max sideways moves,
stopping.")
            break

        # Mengembalikan kubus terbaik, nilai objektif
        # terbaik, jumlah iterasi, dan semua objektif function
        return current_state, current_score,
len(scores), scores
```

### 3.2.3 Random Restart Hill Climbing

Untuk penerapan Random Restart terdapat pada file hill\_climbing.py

**Tabel 3.2.3.1.** Fungsi Random Restart Hill Climbing

Nama Fungsi	random_restart(self, cube, max_restarts,
-------------	--

	max_iterations)
Deskripsi Fungsi	Fungsi ini merupakan fungsi utama yang melakukan pencarian dengan algoritma Random Restart Hill Climbing. Fungsi ini akan menerima input max_restarts dan max_iterations, lalu memanggil fungsi steepest_ascent untuk mendapatkan state terbaik. Fungsi ini akan mengembalikan cube terbaik, nilai objective terbaik, jumlah iterasi, dan semua objective function.
Kode	<pre>def random_restart(self, cube, max_restarts, max_iterations):     # Inisialisasi variabel untuk     menyimpan kubus terbaik, nilai objektif     terbaik, dan skor terbaik     best_state = cube     best_score = best_state.objective_function()     scores = []     display_text = []      # Looping untuk setiap restart     for restart in range(max_restarts):          # Inisialisasi kubus baru         cube_instance = Cube()          # Menjalankan algoritma steepest         ascent         current_state, current_score, _, current_scores = self.steepest_ascent(cube_instance, max_iterations)          # Menyimpan skor dari iterasi saat         ini         scores.extend(current_scores)</pre>

```
# Cek jika skor dari iterasi ini  
lebih baik dari best_score  
  
    if current_score < best_score:  
        best_state = current_state  
        best_score = current_score  
  
    display_text =  
Visual.display_iteration(display_text,  
restart, current_score, "random_restart",  
max_restarts)  
  
    # Berhenti jika ditemukan solusi  
optimal (skor 0)  
    if current_score == 0:  
        print("Optimal solution  
found!")  
        break  
  
    # Mengembalikan hasil terbaik dan  
semua skor dari semua iterasi  
    return best_state, best_score,  
len(scores), scores
```

### 3.2.4 Stochastic Hill Climbing

Untuk penerapan Stochastic terdapat pada file hill\_climbing.py

**Tabel 3.2.4.1.** Fungsi Stochastic Hill Climbing

Nama Fungsi	stochastic(self, cube, max_iterations)
Deskripsi Fungsi	Fungsi ini merupakan fungsi utama untuk melakukan searching menggunakan stochastic hill climbing. Fungsi ini akan menerima input cube dan max_iterations dan memanggil get_all_neighbor untuk kemudian memilih neighbor secara acak dan mengevaluasi objective function. Fungsi ini akan mengembalikan cube terbaik, nilai objective terbaik,

	jumlah iterasi, dan semua objective function.
Kode	<pre>def stochastic(self, cube, max_iterations):     # Inisialisasi variabel untuk menyimpan kubus     # saat ini dan objective function saat ini     current_state = cube     current_score =     current_state.objective_function()     scores = []     display_text = []      # Looping hingga mencapai maksimum iterasi     for iteration in range(max_iterations):          # Menyimpan skor pada iterasi saat ini         scores.append(current_score)          # Mendapatkan neighbor terbaik dari kubus         # saat ini         neighbors =         self.get_all_neighbors(current_state)          # Mengambil neighbor secara acak         next_state = random.choice(neighbors)         next_score =         next_state.objective_function()          # Jika skor dari neighbor lebih baik dari         # skor saat ini, maka kubus saat ini diganti dengan         # neighbor terbaik         if next_score &lt; current_score:             current_state = next_state             current_score = next_score         # Jika skor dari neighbor sama dengan skor         # saat ini, maka kubus saat ini diganti dengan neighbor         # terbaik         else:             pass          # Menampilkan skor dan iterasi saat ini</pre>

	<pre>         display_text = Visual.display_iteration(display_text, iteration, current_score, "stochastic")          # Mengembalikan kubus terbaik, nilai objektif         # terbaik, jumlah iterasi, dan semua objektif function         return current_state, current_score, len(scores), scores     </pre>
--	---

### 3.6 Simulated Annealing

Untuk penerapan Simulated Annealing terdapat pada file `simulated_annealing.py`

**Tabel 3.6.1.** Fungsi Konstruktor Simulated Annealing

Nama Fungsi	<code>__init__(self)</code>
Deskripsi Fungsi	Fungsi ini merupakan konstruktor dari Simulated Annealing untuk menginisialisasi variabel <code>best_cube</code> (state terbaik) dan <code>best_value</code> (nilai objektif terbaik).
Kode	<pre> def __init__(self):     self.best_cube = None     self.best_value = float('inf')     </pre>

**Tabel 3.6.2.** Fungsi Set State Simulated Annealing

Nama Fungsi	<code>set_state(self, cube, objective_value)</code>
Deskripsi Fungsi	Fungsi ini menyimpan <code>current state</code> terbaik dan nilai objektifnya, serta memperbarui variabel <code>best_cube</code> dan <code>best_value</code> .
Kode	<pre> def set_state(self, cube, objective_value):     cubeInstance = Cube()     cubeInstance(cube)     self.best_cube = cubeInstance     self.best_value = objective_value     </pre>

**Tabel 3.6.3.** Fungsi Generate Neighbor Simulated Annealing

Nama Fungsi	generate_neighbor(self, current_state)
Deskripsi Fungsi	Fungsi ini menghasilkan <i>neighbor state</i> dengan menukar dua angka secara acak pada kubus saat ini, serta menghitung nilai objektif dari <i>neighbor state</i> yang dihasilkan.
Kode	<pre>def generate_neighbor(self, current_state, ):     # Melakukan copy pada current state     neighbor_state = np.copy(current_state)      # Melakukan swap pada dua angka secara acak     x1, y1, z1 = random.randint(0, 4), random.randint(0, 4), random.randint(0, 4)     x2, y2, z2 = random.randint(0, 4), random.randint(0, 4), random.randint(0, 4)     neighbor_state[x1, y1, z1], neighbor_state[x2, y2, z2] = neighbor_state[x2, y2, z2], neighbor_state[x1, y1, z1]      # Menghitung nilai objektif dari neighbor     cube_instance = Cube()     cube_instance.cube = neighbor_state     neighbor_objective = cube_instance.objective_function()      # Mengembalikan neighbor state dan objective     return neighbor_state, neighbor_objective</pre>

**Tabel 3.6.4.** Fungsi Search Simulated Annealing

Nama Fungsi	simulated_annealing(self, cube_instance, initial_temperature, cooling_rate, min_temperature, max_iterations)
Deskripsi Fungsi	Fungsi ini merupakan fungsi utama untuk melakukan searching menggunakan algoritma Simulated Annealing. Fungsi ini menerima input <i>state awal</i> ( <i>cube_instance</i> ), <i>initial_temperature</i> , <i>cooling_rate</i> , <i>min_temperature</i> , dan <i>max_iterations</i> kemudian

	<p>memanggil fungsi generate_neighbor. Fungsi ini mengembalikan state terbaik, nilai objektif terbaik, nilai objektif setiap iterasi, <i>acceptance probability neighbor</i> yang lebih buruk, dan jumlah iterasi ketika algoritma terjebak pada <i>local optimum</i>.</p>
Kode	<pre>def simulated_annealing(self, cube_instance, initial_temperature, cooling_rate, min_temperature, max_iterations):     # Inisialisasi variabel     T = initial_temperature     current_state = np.copy(cube_instance.cube)     current_score =     cube_instance.objective_function()     display_text = []      # Set state awal     self.set_state(current_state, current_score)      # Inisialisasi list untuk menyimpan nilai     objective function dan acceptance probability     objective_values = []     acceptance_probabilities = []      # Inisialisasi counter stuck dan unchanged     iterations     stuck_counter = 0     unchanged_iterations = 0      # Melakukan iterasi sebanyak iterasi maksimal     # yang di-input     for iteration in range(max_iterations):         # Jika temperatur kurang dari temperatur         minimum, maka stop iterasi         if T &lt; min_temperature:             break          # Generate neighbor dengan memanggil         # fungsi         neighbor_state, neighbor_objective =</pre>

```
self.generate_neighbor(current_state)

        # Menghitung ΔE
        delta_E = neighbor_objective -
current_score

        # Menghitung acceptance probability
        acceptance_probability = math.exp(-delta_E
/ T) if delta_E > 0 else 1

        # Menerima neighbor jika acceptance
probability lebih besar dari random uniform
        if random.uniform(0, 1) <
acceptance_probability:
            current_state = neighbor_state
            current_score = neighbor_objective

            # Update best state jika nilai
objective lebih kecil
            if current_score < self.best_value:
                self.set_state(current_state,
current_score)

        # Menyimpan nilai objective function
        objective_values.append(current_score)

        # Menyimpan acceptance probability jika ΔE
> 0
        if delta_E > 0:
            acceptance_probabilities.append(acceptance_probabilit
y)

        # Jika ΔE = 0, maka tambah unchanged
iterations dan stuck counter
        if delta_E == 0:
            unchanged_iterations += 1
            stuck_counter += 1
```

```

        else:
            unchanged_iterations = 0

            # Mengurangi temperatur dengan cooling
            rate
            T *= cooling_rate

            # Menampilkan nilai objective function
            setiap iterasi
            display_text =
            Visual.display_iteration(display_text, iteration,
            current_score, "simulated_annealing")

            # Mengembalikan kubus terbaik, nilai objektif
            terbaik, nilai objektif, acceptance probability, dan
            stuck counter
            return self.best_cube, self.best_value,
            objective_values, acceptance_probabilities,
            stuck_counter
    
```

### 3.7 Genetic Algorithm

Untuk penerapan Genetic Algorithm terdapat pada file genetic\_algorithm.py

**Tabel 3.7.1.** Fungsi Konstruktor Genetic Algorithm

Nama Fungsi	<code>__init__(self)</code>
Deskripsi Fungsi	Fungsi ini merupakan konstruktor dari Genetic Algorithm untuk kemudian di-instansiasi di file algorithm.py.
Kode	<pre> def __init__(self):     self.best_cube = Cube()     self.best_value = float('inf')     self.display_text = []     </pre>

**Tabel 3.7.2.** Fungsi Populate Genetic Algorithm

Nama Fungsi	populate(self, population_size)
Deskripsi Fungsi	Fungsi ini merupakan fungsi untuk melakukan inisiasi populasi berupa sejumlah cube sesuai inputan user.
Kode	<pre>def populate(self, population_size):     population = [Cube() for _ in range(population_size)]     return population</pre>

**Tabel 3.7.3.** Fungsi Selection Genetic Algorithm

Nama Fungsi	selection(self, population)
Deskripsi Fungsi	Fungsi ini merupakan fungsi untuk melakukan proses seleksi pada populasi yang di-input oleh user.
Kode	<pre>def selection(self, population):     # Menghitung nilai objektif untuk     # setiap kubus dalam populasi     fitness_scores = [cube.objective_function() for cube in population]      # Menghitung probabilitas seleksi     # secara kumulatif     total_fitness = sum(fitness_scores)     selection_probs = [(total_fitness - score) / total_fitness for score in fitness_scores]     cumulative_probs = np.cumsum(selection_probs) / np.sum(selection_probs) * 100      # Memilih parents berdasarkan     # probabilitas kumulatif     parents = []     for cube in range(len(population)):         random_percentage =</pre>

	<pre> random.uniform(0, 100)         for i, prob in enumerate(cumulative_probs):             if random_percentage &lt;= prob:                 parents.append(population[i])                 break          # Mengembalikan parents yang telah dipilih     return parents </pre>
--	---

**Tabel 3.7.4.** Fungsi Crossover Genetic Algorithm

Nama Fungsi	crossover(self, parents)
Deskripsi Fungsi	Fungsi ini merupakan fungsi untuk melakukan proses crossover pada parents hasil proses seleksi
Kode	<pre> # Crossover function ensuring unique elements in parentren  def crossover(self, parents):     new_population = []      # Proses crossover untuk setiap pasangan parents     for i in range(0, len(parents), 2):         parent1 = parents[i]         parent2 = parents[i + 1]          # Menentukan titik crossover secara acak         crossover_point = random.randint(0, 124)          # Membuat dua kubus baru sebagai child         child1 = Cube() </pre>

```
child2 = Cube()

        # Melakukan crossover pada titik
        yang telah ditentukan

child1(cube.flat[crossover_point:],
child2(cube.flat[crossover_point:] =
parent2(cube.flat[crossover_point:],
parent1(cube.flat[crossover_point:]

        # Mengisi sisa elemen yang belum
        terisi pada child1 dan memastikan elemen yang
        diisi unik
        currentIndex = 0
        for j in range(crossover_point,
125):
            for k in range(currentIndex,
125):
                if parent2(cube.flat[k]
not in child1(cube.flat:
                    child1(cube.flat[j] =
parent2(cube.flat[k]
                    currentIndex = k
                    break

        # Mengisi sisa elemen yang belum
        terisi pada child2 dan memastikan elemen yang
        diisi unik
        currentIndex = 0
        for j in range(crossover_point,
125):
            for k in range(currentIndex,
125):
                if parent1(cube.flat[k]
not in child2(cube.flat:
                    child2(cube.flat[j] =
parent1(cube.flat[k]
```

	<pre>         currentIdx = k         break          # Menambahkan parents yang telah         di-cross-over ke populasi baru         new_population.append(parent1)         new_population.append(parent2)      return new_population </pre>
--	---

**Tabel 3.7.5.** Fungsi Mutation Genetic Algorithm

Nama Fungsi	mutation(self, new_population)
Deskripsi Fungsi	Fungsi ini merupakan fungsi untuk melakukan proses mutasi pada populasi baru hasil dari proses crossover
Kode	<pre> # Proses Mutasi def mutation(self, new_population):     for individual in new_population:         # Melakukan swap pada dua elemen         # secara acak         idx1, idx2 = random.randint(0, 124), random.randint(0, 124)         individual.cube.flat[idx1], individual.cube.flat[idx2] = individual.cube.flat[idx2], individual.cube.flat[idx1]      return new_population </pre>

**Tabel 3.7.6.** Fungsi Search Genetic Algorithm

Nama Fungsi	genetic_search(self, population_size, max_iterations)
Deskripsi Fungsi	Fungsi ini merupakan fungsi utama untuk melakukan searching menggunakan genetic algorithm. Fungsi ini akan memanggil fungsi populate, selection,

	crossover, dan mutation kemudian mengembalikan state terbaik, score terbaik, populasi akhir, kumpulan score terbaik, dan rata-rata score dari semua iterasi yang sudah dilalui.
Kode	<pre># Proses pencarian menggunakan algoritma genetik      def genetic_search(self, population_size, max_iterations):         # Inisialisasi array skor         min_scores = []         avg_scores = []          # Inisialisasi populasi         population = self.populate(population_size)          for iteration in range(max_iterations):             # Proses seleksi             parents = self.selection(population)              # Proses crossover             crossover_result = self.crossover(parents)              # Proses mutasi             population = self.mutation(crossover_result)              # Menghitung nilai objektif untuk setiap kubus dalam populasi             scores = [cube.objective_function() for cube in population]             min_score = min(scores)             max_score = max(scores)</pre>

```
        # Memperbarui skor terbaik dan
        cube terbaik
        if min_score < self.best_value:
            self.best_value = min_score
            self.best_cube =
deepcopy(population[scores.index(self.best_ va
lue)])
        # Menyimpan skor minimum dan
        rata-rata
        min_scores.append(self.best_value)
        avg_scores.append(sum(scores) /
len(scores))

        # Menampilkan informasi iterasi
        self.display_text =
Visual.display_iteration(self.display_text,
iteration, self.best_value, "genetic")

        # Mengembalikan kubus terbaik, nilai
        objektif terbaik, populasi, skor minimum, dan
        skor rata-rata
        return self.best_cube,
self.best_value, population, min_scores,
avg_scores
```

## 4. Hasil Eksperimen

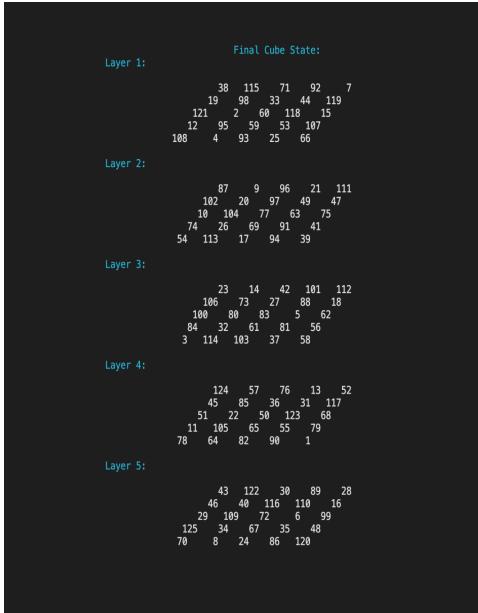
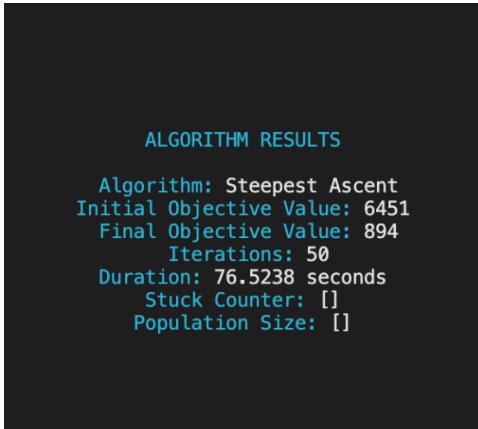
### 4.1 Hill Climbing

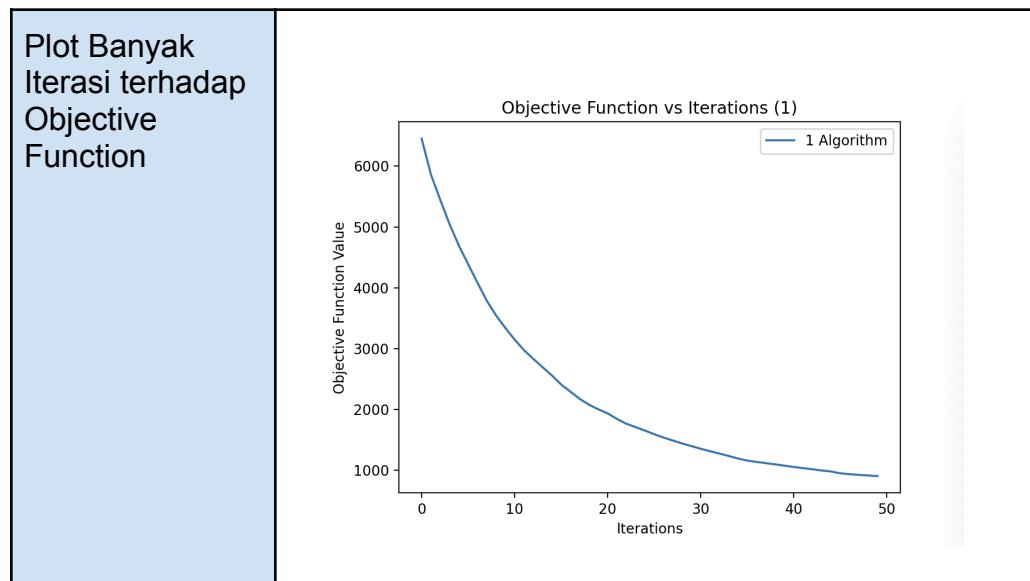
#### 4.1.1 Steepest Ascent Hill Climbing

- Pengujian 1 - Iterasi 50

**Tabel 4.1.1.1.** Pengujian 1 Steepest Ascent

State Awal	<pre> Initial Cube State: Layer 1:       38   115   71   92   7       19   98    33   44  119      121    2   60   118  15      12    95   59   53  107      108    4   93   25   66  Layer 2:       87    9   96   21   111       102   20   97   49   47       10    104   77   63   75       74   26   69   91   41       54   113   17   94   39  Layer 3:       23   14   42   101  112       106   80   83   5    62       100   32   61   81   56       84   114  103   37   58  Layer 4:       124   57   76   13   52       45   85   36   31  117       51   22   65   123  68       11   105   55   79   1       78   64   82   90   1  Layer 5:       43   122   30   89   28       46   40   116   110  16       29   109   72    6   99       34   67   35   48       70     8   24   86  120   </pre>
Objective Awal	6451

State Akhir	
Objective Akhir	894
Banyak Iterasi	50
Hasil Eksperimen	
Durasi Pencarian	76.5238 detik

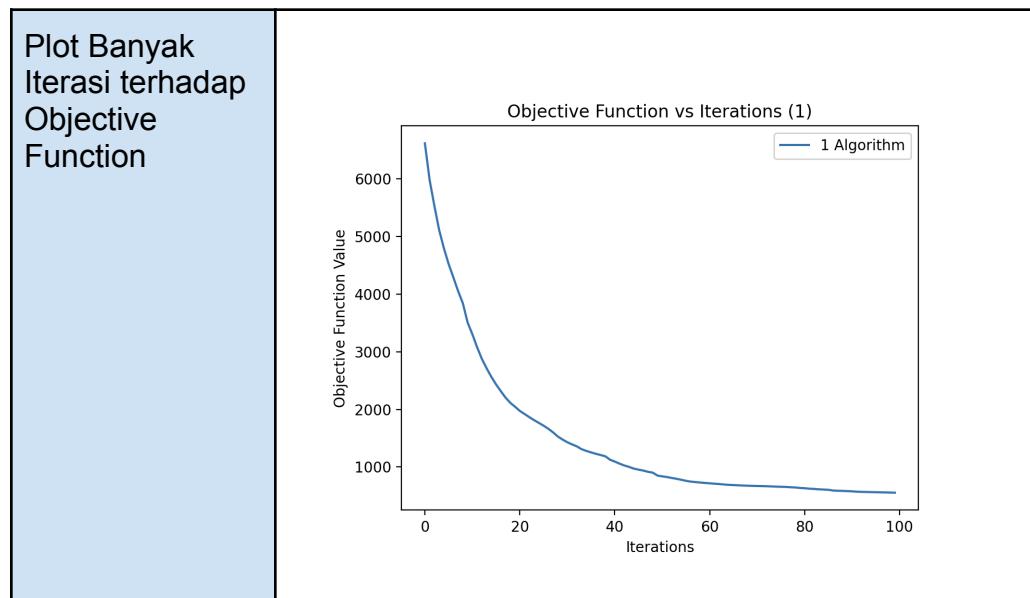


## 2. Pengujian 2 - Iterasi 100

**Tabel 4.1.1.2. Pengujian 2 Steepest Ascent**

State Awal	<pre> Initial Cube State:  Layer 1:       38   48   119   8   102      117   64    3   26   104       91   15   111   86   11       62   70    1   93   89        7  118   84   97    9  Layer 2:       35   56   125   16   80       75   107   32   88   12       67   53   43   82   73       41    6   45   99  120       98   92   68   29   30  Layer 3:       110   50   23   109   22       21   46   77   112   57       52   90   36   55   83       81   96   66   18   54       49   28  113   20  105  Layer 4:       17   122    4   121   51       100   78   116    5   24       31   33   65   79   106       44   71   101   60   39       123   10   27   59   95  Layer 5:       115   40   34   61   72        2   19   94   85   42       74   124   58   14   42       87   63  103   47   13       37   69   25  108   76   </pre>
Objective Awal	6617

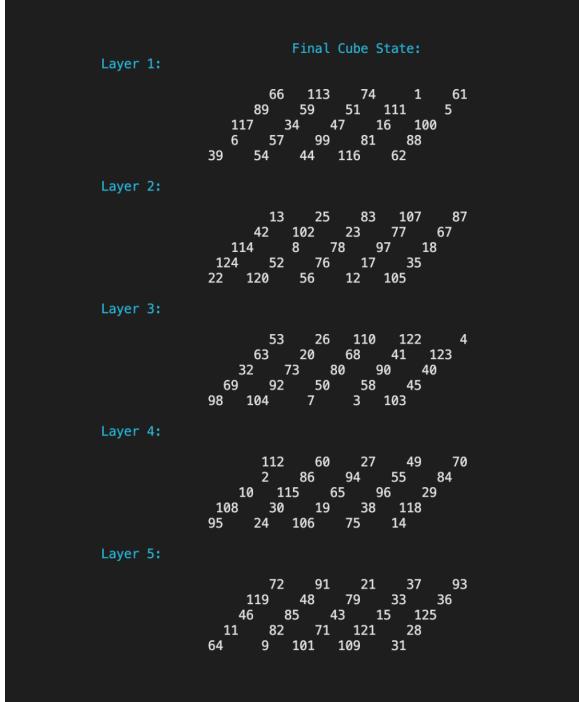
State Akhir	<pre> Final Cube State: Layer 1:       38   48   119   8   102       117   64    3   26   104       91   15   111   86   11       62   70    1   93   89       7   118   84   97    9  Layer 2:       35   56   125   16   80       75   107   32   88   12       67   53   43   82   73       41    6   45   99   120       98   92   68   29   30  Layer 3:       110   50   23   109   22       21   46   77   112   57       52   90   36   55   83       81   96   66   18   54       49   28   113   20   105  Layer 4:       17   122   4   121   51       100   78   116   5   24       31   33   65   79   39       44   71   101   60   95       123   10   27   59   95  Layer 5:       115   40   34   61   72       2   19   94   85   114       74   124   58   14   42       87   63   103   47   13       37   69   25   108   76   </pre>
Objective Akhir	552
Banyak Iterasi	100
Hasil Eksperimen	<pre> ALGORITHM RESULTS  Algorithm: Steepest Ascent Initial Objective Value: 6617 Final Objective Value: 552 Iterations: 100 Duration: 147.5732 seconds Stuck Counter: [] Population Size: []   </pre>
Durasi Pencarian	147.5732

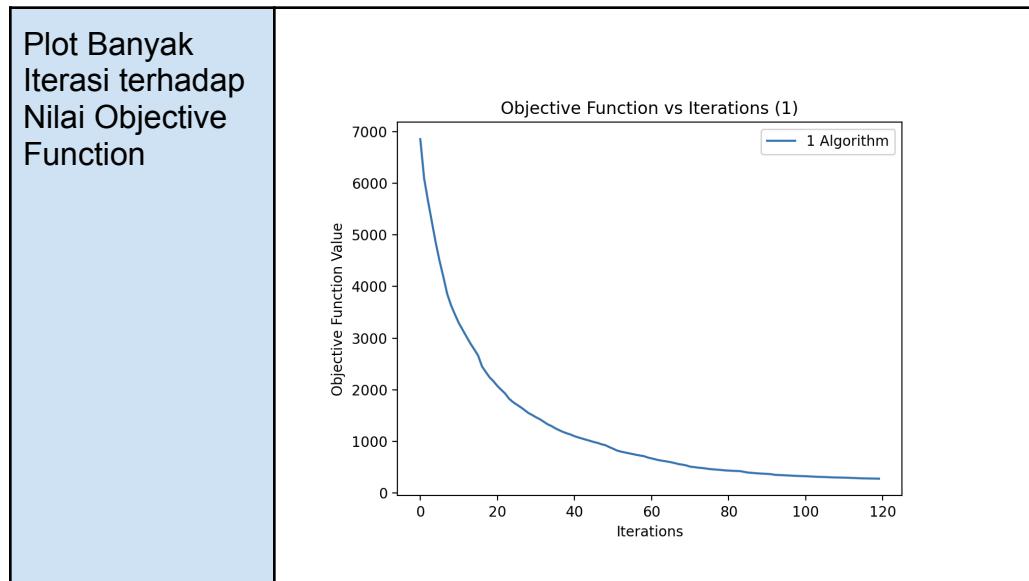


### 3. Pengujian 3 - Iterasi 120

**Tabel 4.1.1.3.** Pengujian 2 Steepest Ascent

State Awal	<pre> Initial Cube State:  Layer 1:       66   113   74   1   61       89   59    51   111  100      117   34    47   16   88        6   57    99   81   88       39   54    44   116  62  Layer 2:       13   25   83   107  87       42   102  23   77   67      114   8    78   97   18      124   52   76   17   35       22   120  56   12   105  Layer 3:       53   26   110  122  4       32   73   80   41   123       69   92   50   58   45       98   104  7    3    103  Layer 4:       112  60   27   49   70        2   86   94   55   84       10   115  65   96   29      108   30   19   38   118       95   24   106  75   14  Layer 5:       72   91   21   37   93       119  48   79   33   36       46   85   43   15   125       11   82   71   121  28       64   9    101  109  31   </pre>
------------	---

Objective Awal	6858
State Akhir	<p>Final Cube State:</p>  <pre> Layer 1:       66 113 74 1 61       89 59 51 111 5       117 34 47 81 100          6 57 99 16 88          39 54 44 116 62  Layer 2:       13 25 83 107 87       42 102 23 77 67       114 8 78 97 18       124 52 76 17 35          22 120 56 12 105  Layer 3:       53 26 110 122 4       63 20 68 41 123       32 73 80 90 40       69 92 50 58 45          98 104 7 3 103  Layer 4:       112 60 27 49 70       2 86 94 55 84       10 115 65 96 29       108 30 19 38 118          95 24 106 75 14  Layer 5:       72 91 21 37 93       119 48 79 33 36       46 85 43 15 125       11 82 71 121 28          64 9 101 109 31   </pre>
Objective Akhir	272
Banyak Iterasi	120
Hasil Eksperimen	<p>ALGORITHM RESULTS</p> <pre> Algorithm: Steepest Ascent Initial Objective Value: 6858 Final Objective Value: 272 Iterations: 120 Duration: 186.9664 seconds Stuck Counter: [] Population Size: []   </pre>
Durasi Pencarian	186.9664 detik



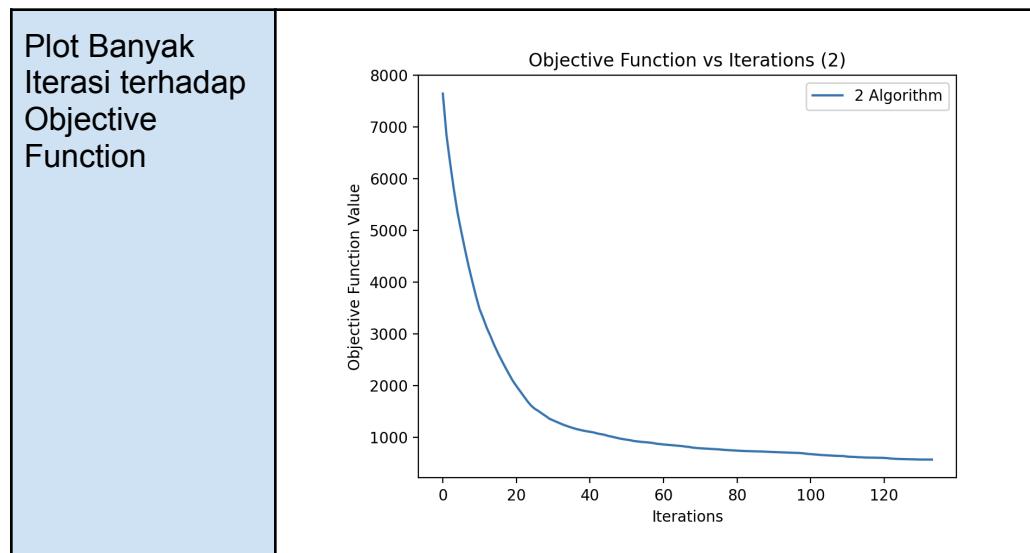
#### 4.1.2 Hill Climbing with Sideways Move

1. Pengujian 1 - Iterasi Max Sideways 5

**Tabel 4.1.2.1.** Pengujian 1 Hill Climbing with Sideways Move

State Awal	Initial Cube State:
	Layer 1: 4 45 105 86 75 79 115 71 68 3 101 37 25 15 123 91 108 53 57 7 39 10 54 89 114
	Layer 2: 103 90 29 32 60 38 72 44 50 104 21 102 81 97 9 73 47 93 17 100 77 8 67 119 42
	Layer 3: 117 28 85 34 51 96 14 46 66 95 87 63 58 84 23 11 120 16 74 94 20 83 110 49 52
	Layer 4: 40 31 92 43 109 41 80 125 12 48 2 70 62 106 99 121 22 30 98 24 111 107 6 56 35
	Layer 5: 55 116 5 113 26 59 33 27 124 64 118 36 82 13 61 19 18 122 69 88 65 112 78 1 76

Objective Awal	7646
State Akhir	<pre>           Final Cube State: Layer 1:         4   45   105   86   75       79   115   25   68   3     101   37   15   123     91   108   53   57   7     39   10   54   89   114  Layer 2:         103   90   29   32   60       38   72   44   50   104     21   102   81   97   9     73   47   93   17   100     77   8   67   119   42  Layer 3:         117   28   85   34   51       96   14   46   66   95     87   63   58   84   23     11   120   16   74   94     20   83   110   49   52  Layer 4:         40   31   92   43   109       41   80   125   12   48       2   70   62   106   99     121   22   30   98   24     111   107   6   56   35  Layer 5:         55   116   5   113   26       59   33   27   124   64     118   36   82   13   61     19   18   122   69   88     65   112   78   1   76 </pre>
Objective Akhir	565
Banyak Iterasi	134
Hasil Eksperimen	<pre> ALGORITHM RESULTS  Algorithm: Sideways Initial Objective Value: 7646 Final Objective Value: 565 Iterations: 134 Duration: 217.1456 seconds Stuck Counter: [] Population Size: [] </pre>
Durasi Pencarian	217.1456

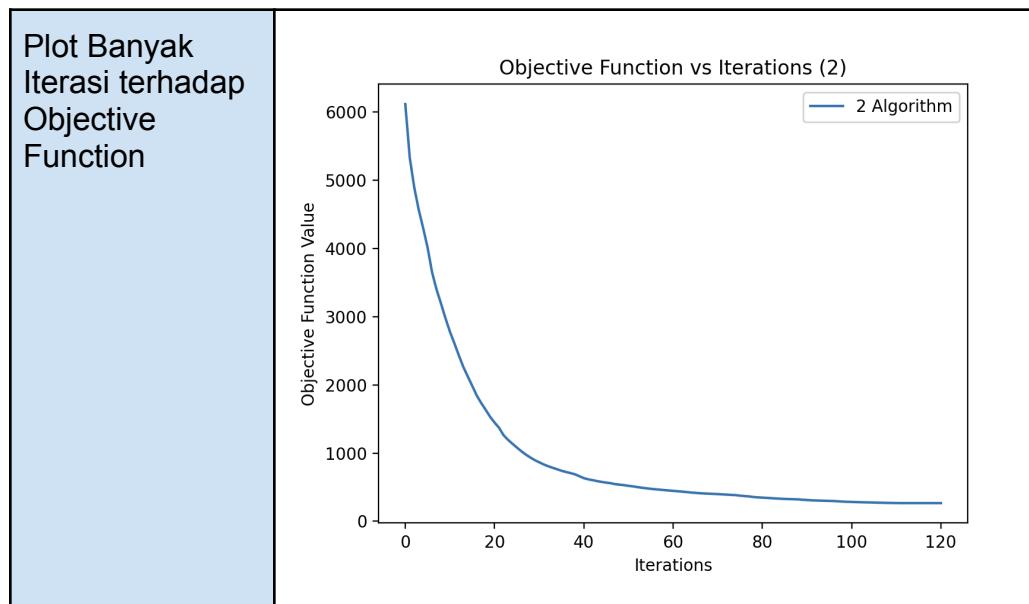


## 2. Pengujian 2 - Iterasi Max Sideways 10

**Tabel 4.1.2.2.** Pengujian 2 Hill Climbing with Sideways Move

State Awal	<pre> Initial Cube State:  Layer 1:       18   32   123   52   90       76   122   66   26   24       78   61     1  125   51       33   88   104   59   31     110   12    22   53   117  Layer 2:       121   44   10   106   34       93   37   23   58   109       16   91   86   48   75     21   73   111   42   68     64   70   84   67   29  Layer 3:       39   20   124   50   82       17   118   14   72   94       45   114   56   62   35     113     3    6   95   97     101   60   112   36    7  Layer 4:       46   119   40   8   102       47   13   107   71   77       103   55   28   49   69       2   116   54   80   63  Layer 5:       92   100   19   99   5       83   25   105   87   15       74   41   89   30   81       27   96   65   11   115       38   57   43   79   98   </pre>
Objective Awal	6113

State Akhir	<pre> Final Cube State: Layer 1:       18   32   123   52   90       76   122   66   26   24       78   61    1   125   51       33   88   104   59   31      110   12   22   53   117  Layer 2:       121   44   10   106   34       93   37   23   58   109       16   91   86   48   75       21   73   111   42   68       64   70   84   67   29  Layer 3:       39   20   124   50   82       17   118   14   72   94       45   114   56   62   35      113   3   6   95   97      101   60   112   36   7  Layer 4:       46   119   40   8   102       47   13   107   71   77       103   9   85   49   69      120   55   28   108   4       2   116   54   80   63  Layer 5:       92   100   19   99   5       83   25   105   87   15       74   41   89   30   81       27   96   65   11   115       38   57   43   79   98 </pre>
Objective Akhir	267
Banyak Iterasi	121
Hasil Eksperimen	<pre> ALGORITHM RESULTS  Algorithm: Sideways Initial Objective Value: 6113 Final Objective Value: 267 Iterations: 121 Duration: 197.2577 seconds Stuck Counter: [] Population Size: [] </pre>
Durasi Pencarian	197.2577 detik

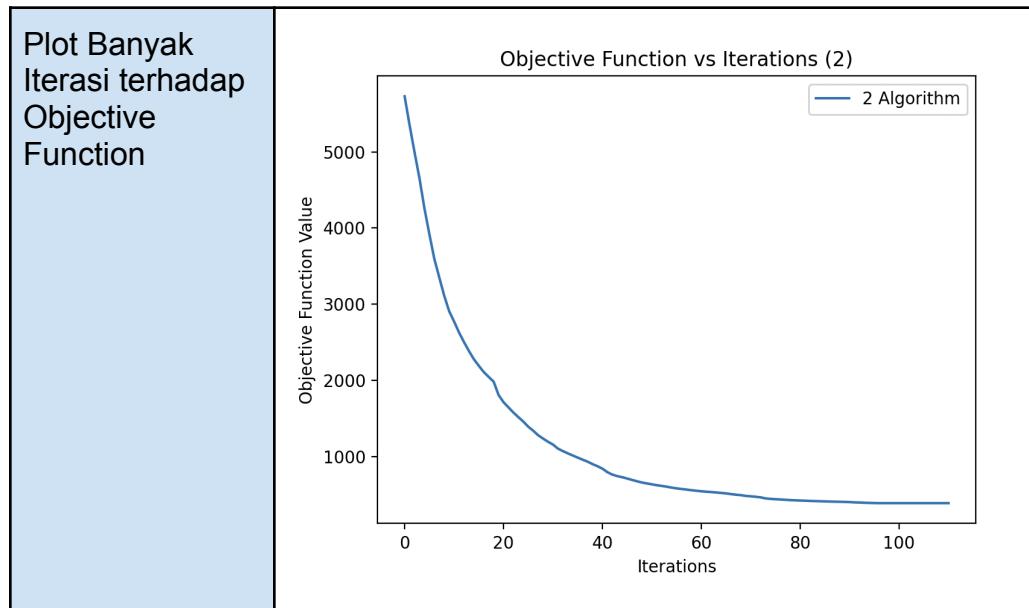


### 3. Pengujian 3 - Iterasi Max Sideways 15

**Tabel 4.1.2.3.** Pengujian 3 Hill Climbing with Sideways Move

State Awal	Initial Cube State: Layer 1: 78 110 47 28 53 8 98 59 112 38 45 29 79 119 43 122 4 18 24 123 67 74 111 27 37  Layer 2: 12 48 121 66 64 125 51 6 71 61 120 23 70 16 86 22 90 83 94 36 20 106 34 68 88  Layer 3: 9 11 100 116 81 87 97 85 33 14 41 77 62 46 89 72 30 17 107 91 109 102 50 13 40  Layer 4: 101 118 21 2 75 80 7 108 39 84 49 82 56 124 3 19 76 105 55 58 69 32 25 93 96  Layer 5: 115 31 26 103 42 15 65 57 63 117 60 104 44 10 99 73 114 92 35 5 1 95 113 54
------------	---

Objective Awal	5731
State Akhir	<pre> Final Cube State: Layer 1:       78   110   47   28   53       8    98   59   112   38      45   29   79   119   43      122   4   18   24   123      67   74   111   27   37  Layer 2:       12   48   121   66   64       125   51   6   71   61      120   23   70   16   86      22   90   83   94   36      20   106   34   68   88  Layer 3:       9   11   100   116   81       87   97   85   33   14      41   77   62   46   89      72   30   17   107   91      109   102   50   13   40  Layer 4:       101   118   21   2   75       80   7   108   39   84      49   82   56   124   3      19   76   105   55   58      69   32   25   93   96  Layer 5:       115   31   26   103   42       15   65   57   63   117       60   104   44   10   99      73   114   92   35   5       1   95   113   54 </pre>
Objective Akhir	387
Banyak Iterasi	111
Hasil Eksperimen	<pre> ALGORITHM RESULTS  Algorithm: Sideways Initial Objective Value: 5731 Final Objective Value: 387 Iterations: 111 Duration: 181.8339 seconds Stuck Counter: [] Population Size: [] </pre>
Durasi Pencarian	181.8339 detik

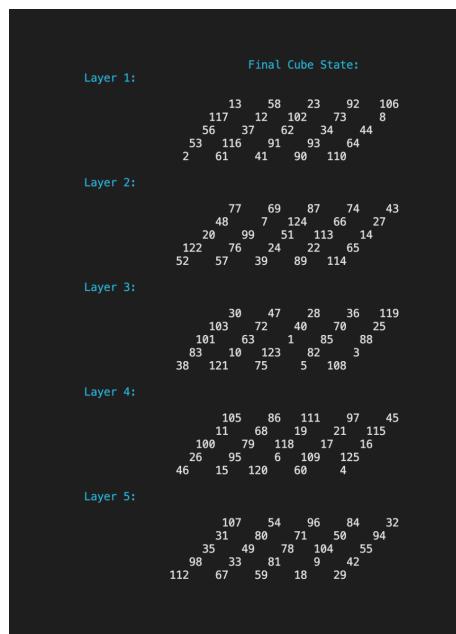


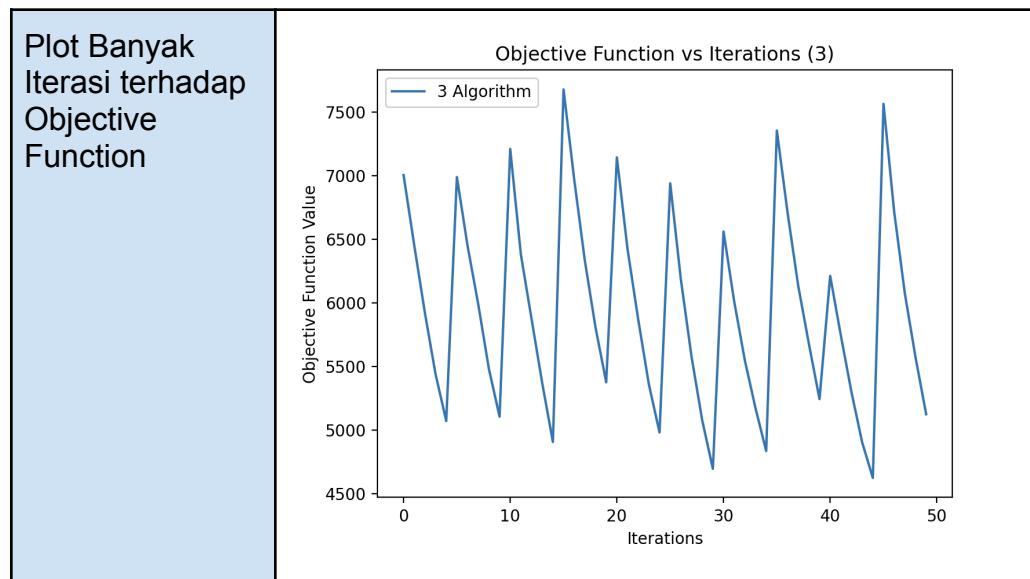
#### 4.1.3 Random Restart Hill Climbing

- Pengujian 1 - Restart 10 dengan 5 Iterasi

**Tabel 4.1.3.1.** Pengujian 1 Random Restart Hill Climbing

State Awal	<pre> Initial Cube State: Layer 1:       64   32   50   51   52       16   62   90   15   121       53   102  30   42   74       186  68   84   88   77       73   17   91   79   43  Layer 2:       59   37   75   4   24       97   117  27   39       106  104  47   78       11   125  49   34       81   7   13   44   18       85   68   116  45   58  Layer 3:       48   70   69   6   38       99   33   72   101  111       112  71   36   103  14       12   116  45   26   10       85   68   116  45   26  Layer 4:       124  80   96   48   119       113  57   56   123  95       93   63   23   61   92       5    100  55   107  54       28   3    120  76   122  Layer 5:       82   109  89   67   118       29   105  110  87   114       1    2    86   35   94       98   19   83   21   66       115  65   9    41   8   </pre>
Objective Awal	6788

State Akhir	
Objective Akhir	4345
Banyak Iterasi	50
Hasil Eksperimen	
Durasi Pencarian	86.3300 detik

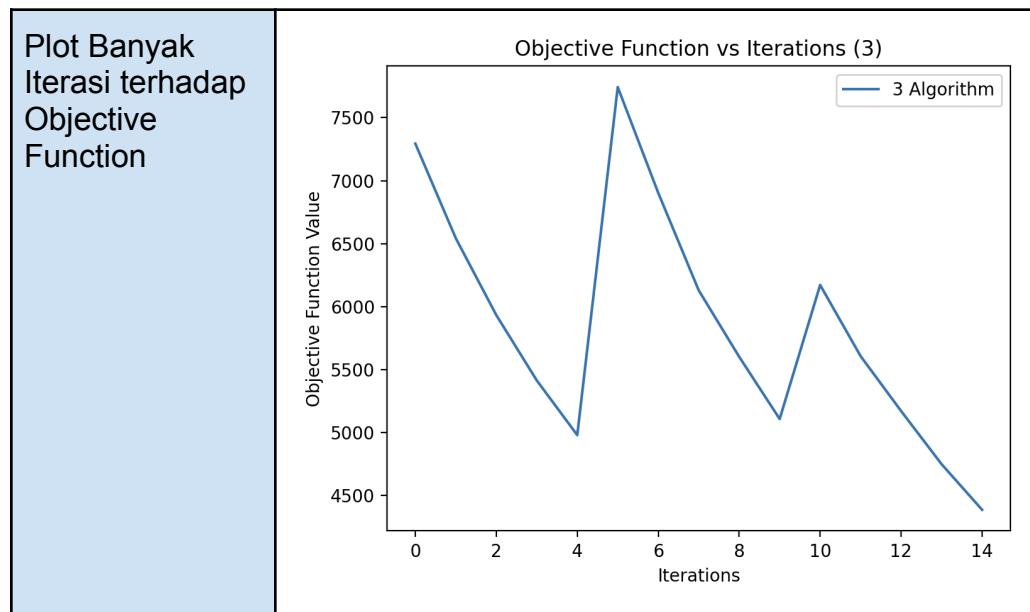


## 2. Pengujian 2 - Restart 3 dengan 5 Iterasi

**Tabel 4.1.3.2.** Pengujian 2 Random Restart Hill Climbing

State Awal	Initial Cube State: Layer 1: <table border="1"> <tr><td>105</td><td>61</td><td>60</td><td>125</td><td>47</td></tr> <tr><td>71</td><td>57</td><td>98</td><td>120</td><td>99</td></tr> <tr><td>5</td><td>64</td><td>74</td><td>28</td><td>2</td></tr> <tr><td>12</td><td>14</td><td>34</td><td>36</td><td>25</td></tr> <tr><td>79</td><td>69</td><td>53</td><td>116</td><td>94</td></tr> </table> Layer 2: <table border="1"> <tr><td>67</td><td>123</td><td>68</td><td>107</td><td>20</td></tr> <tr><td>118</td><td>65</td><td>51</td><td>13</td><td>44</td></tr> <tr><td>122</td><td>39</td><td>88</td><td>70</td><td>30</td></tr> <tr><td>8</td><td>117</td><td>90</td><td>92</td><td>62</td></tr> <tr><td>59</td><td>121</td><td>3</td><td>17</td><td>111</td></tr> </table> Layer 3: <table border="1"> <tr><td>119</td><td>110</td><td>10</td><td>77</td><td>22</td></tr> <tr><td>115</td><td>48</td><td>55</td><td>32</td><td>19</td></tr> <tr><td>42</td><td>106</td><td>72</td><td>80</td><td>73</td></tr> <tr><td>108</td><td>4</td><td>24</td><td>89</td><td>26</td></tr> <tr><td>95</td><td>29</td><td>21</td><td>103</td><td>27</td></tr> </table> Layer 4: <table border="1"> <tr><td>63</td><td>56</td><td>83</td><td>109</td><td>112</td></tr> <tr><td>45</td><td>97</td><td>33</td><td>41</td><td>1</td></tr> <tr><td>43</td><td>82</td><td>85</td><td>114</td><td>40</td></tr> <tr><td>101</td><td>15</td><td>66</td><td>84</td><td>104</td></tr> <tr><td>87</td><td>9</td><td>86</td><td>31</td><td>6</td></tr> </table> Layer 5: <table border="1"> <tr><td>35</td><td>96</td><td>7</td><td>100</td><td>78</td></tr> <tr><td>23</td><td>46</td><td>58</td><td>75</td><td>52</td></tr> <tr><td>81</td><td>38</td><td>49</td><td>93</td><td>37</td></tr> <tr><td>124</td><td>18</td><td>50</td><td>54</td><td>91</td></tr> <tr><td>16</td><td>113</td><td>76</td><td>11</td><td>102</td></tr> </table>	105	61	60	125	47	71	57	98	120	99	5	64	74	28	2	12	14	34	36	25	79	69	53	116	94	67	123	68	107	20	118	65	51	13	44	122	39	88	70	30	8	117	90	92	62	59	121	3	17	111	119	110	10	77	22	115	48	55	32	19	42	106	72	80	73	108	4	24	89	26	95	29	21	103	27	63	56	83	109	112	45	97	33	41	1	43	82	85	114	40	101	15	66	84	104	87	9	86	31	6	35	96	7	100	78	23	46	58	75	52	81	38	49	93	37	124	18	50	54	91	16	113	76	11	102
105	61	60	125	47																																																																																																																										
71	57	98	120	99																																																																																																																										
5	64	74	28	2																																																																																																																										
12	14	34	36	25																																																																																																																										
79	69	53	116	94																																																																																																																										
67	123	68	107	20																																																																																																																										
118	65	51	13	44																																																																																																																										
122	39	88	70	30																																																																																																																										
8	117	90	92	62																																																																																																																										
59	121	3	17	111																																																																																																																										
119	110	10	77	22																																																																																																																										
115	48	55	32	19																																																																																																																										
42	106	72	80	73																																																																																																																										
108	4	24	89	26																																																																																																																										
95	29	21	103	27																																																																																																																										
63	56	83	109	112																																																																																																																										
45	97	33	41	1																																																																																																																										
43	82	85	114	40																																																																																																																										
101	15	66	84	104																																																																																																																										
87	9	86	31	6																																																																																																																										
35	96	7	100	78																																																																																																																										
23	46	58	75	52																																																																																																																										
81	38	49	93	37																																																																																																																										
124	18	50	54	91																																																																																																																										
16	113	76	11	102																																																																																																																										

Objective Awal	6566
State Akhir	<pre> Final Cube State: Layer 1:       23   52   103   17   77   104       10   106   50   71   5       28   49   62   113   94   5       123   80   56   25   75   102       57   8   42   75   102  Layer 2:       21   41   40   105   15       119   74   35   70   26       61   32   89   24   67       121   112   14   97   33       37   28   128   48   45  Layer 3:       78   79   46   92   31       64   65   6   111   76       101   68   53   116   36       60   91   66   22   11       34   82   122   3   98  Layer 4:       108   43   96   1   63       7   38   99   44   72       86   85   18   88   118       13   95   114   87   51       110   39   69   107   19  Layer 5:       38   115   12   27   93       125   29   90   55   100       59   117   124   16   73       58   47   109   84   81       9   2   4   54   83   </pre>
Objective Akhir	4073
Banyak Iterasi	15
Hasil Eksperimen	<pre> ALGORITHM RESULTS  Algorithm: Random Restart Initial Objective Value: 6566 Final Objective Value: 4073 Iterations: 15 Duration: 26.3102 seconds Stuck Counter: [] Population Size: []   </pre>
Durasi Pencarian	26.3102

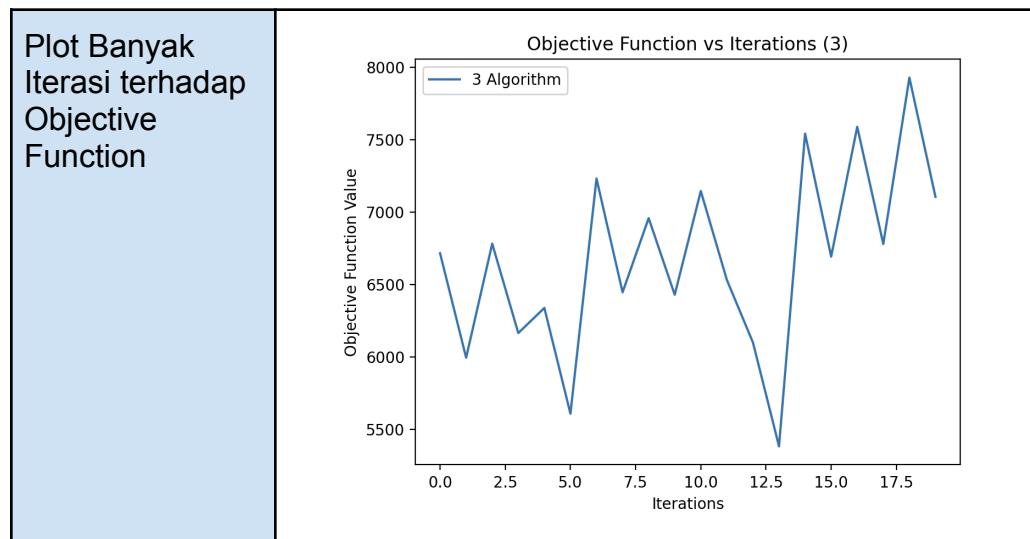


### 3. Pengujian 3 - Restart 10 dengan 2 Iterasi

**Tabel 4.1.3.3.** Pengujian 3 Random Restart Hill Climbing

State Awal	<p>Initial Cube State:</p> <table border="0"> <tr><td>Layer 1:</td><td>60</td><td>104</td><td>85</td><td>34</td><td>82</td></tr> <tr><td></td><td>108</td><td>116</td><td>79</td><td>22</td><td>8</td></tr> <tr><td></td><td>84</td><td>50</td><td>53</td><td>92</td><td>106</td></tr> <tr><td></td><td>74</td><td>95</td><td>120</td><td>72</td><td>35</td></tr> <tr><td></td><td>45</td><td>112</td><td>94</td><td>20</td><td>23</td></tr> </table> <table border="0"> <tr><td>Layer 2:</td><td>21</td><td>109</td><td>31</td><td>117</td><td>76</td></tr> <tr><td></td><td>13</td><td>115</td><td>17</td><td>11</td><td>105</td></tr> <tr><td></td><td>16</td><td>1</td><td>96</td><td>15</td><td>118</td></tr> <tr><td></td><td>89</td><td>125</td><td>102</td><td>65</td><td>3</td></tr> <tr><td></td><td>78</td><td>68</td><td>69</td><td>111</td><td>56</td></tr> </table> <table border="0"> <tr><td>Layer 3:</td><td>99</td><td>103</td><td>25</td><td>47</td><td>122</td></tr> <tr><td></td><td>107</td><td>83</td><td>29</td><td>18</td><td>28</td></tr> <tr><td></td><td>73</td><td>101</td><td>66</td><td>80</td><td>57</td></tr> <tr><td></td><td>67</td><td>33</td><td>36</td><td>30</td><td>55</td></tr> <tr><td></td><td>54</td><td>75</td><td>48</td><td>5</td><td>77</td></tr> </table> <table border="0"> <tr><td>Layer 4:</td><td>58</td><td>90</td><td>59</td><td>88</td><td>64</td></tr> <tr><td></td><td>38</td><td>19</td><td>49</td><td>9</td><td>41</td></tr> <tr><td></td><td>40</td><td>62</td><td>37</td><td>119</td><td>91</td></tr> <tr><td></td><td>14</td><td>97</td><td>93</td><td>32</td><td>87</td></tr> <tr><td></td><td>81</td><td>123</td><td>39</td><td>27</td><td>113</td></tr> </table> <table border="0"> <tr><td>Layer 5:</td><td>6</td><td>44</td><td>98</td><td>100</td><td>114</td></tr> <tr><td></td><td>124</td><td>12</td><td>71</td><td>52</td><td>2</td></tr> <tr><td></td><td>10</td><td>63</td><td>4</td><td>121</td><td>70</td></tr> <tr><td></td><td>46</td><td>42</td><td>24</td><td>7</td><td>26</td></tr> <tr><td></td><td>43</td><td>61</td><td>51</td><td>110</td><td>86</td></tr> </table>	Layer 1:	60	104	85	34	82		108	116	79	22	8		84	50	53	92	106		74	95	120	72	35		45	112	94	20	23	Layer 2:	21	109	31	117	76		13	115	17	11	105		16	1	96	15	118		89	125	102	65	3		78	68	69	111	56	Layer 3:	99	103	25	47	122		107	83	29	18	28		73	101	66	80	57		67	33	36	30	55		54	75	48	5	77	Layer 4:	58	90	59	88	64		38	19	49	9	41		40	62	37	119	91		14	97	93	32	87		81	123	39	27	113	Layer 5:	6	44	98	100	114		124	12	71	52	2		10	63	4	121	70		46	42	24	7	26		43	61	51	110	86
Layer 1:	60	104	85	34	82																																																																																																																																																		
	108	116	79	22	8																																																																																																																																																		
	84	50	53	92	106																																																																																																																																																		
	74	95	120	72	35																																																																																																																																																		
	45	112	94	20	23																																																																																																																																																		
Layer 2:	21	109	31	117	76																																																																																																																																																		
	13	115	17	11	105																																																																																																																																																		
	16	1	96	15	118																																																																																																																																																		
	89	125	102	65	3																																																																																																																																																		
	78	68	69	111	56																																																																																																																																																		
Layer 3:	99	103	25	47	122																																																																																																																																																		
	107	83	29	18	28																																																																																																																																																		
	73	101	66	80	57																																																																																																																																																		
	67	33	36	30	55																																																																																																																																																		
	54	75	48	5	77																																																																																																																																																		
Layer 4:	58	90	59	88	64																																																																																																																																																		
	38	19	49	9	41																																																																																																																																																		
	40	62	37	119	91																																																																																																																																																		
	14	97	93	32	87																																																																																																																																																		
	81	123	39	27	113																																																																																																																																																		
Layer 5:	6	44	98	100	114																																																																																																																																																		
	124	12	71	52	2																																																																																																																																																		
	10	63	4	121	70																																																																																																																																																		
	46	42	24	7	26																																																																																																																																																		
	43	61	51	110	86																																																																																																																																																		
Objective Awal	7342																																																																																																																																																						

State Akhir	<pre> Final Cube State: Layer 1:       109   31   75   6   115       95   21   40   23   103       122   55   48   54       2   111   93   58   1       17   48   62   15   100  Layer 2:       8   10   59   116   56       60   107   71   9   86       53   105   110   102   123       45   87   112   73   70       36   37   83   98   16  Layer 3:       72   57   92   88   42       27   76   119   34   20       104   106   11   90   101       4   61   46   114   117       121   14   29   118   117  Layer 4:       69   89   25   94   49       124   13   51   81   49       43   63   96   32   30       67   7   125   39   41       82   120   99   19   41  Layer 5:       64   26   33   80   66       79   44   108   68   85       35   74   113   38   12       65   50   28   22   78       52   97   47   91   77     </pre>
Objective Akhir	4901
Banyak Iterasi	20
Hasil Eksperimen	<pre> ALGORITHM RESULTS  Algorithm: Random Restart Initial Objective Value: 7342 Final Objective Value: 4901 Iterations: 20 Duration: 40.9341 seconds Stuck Counter: [] Population Size: []     </pre>
Durasi Pencarian	40.9341 detik



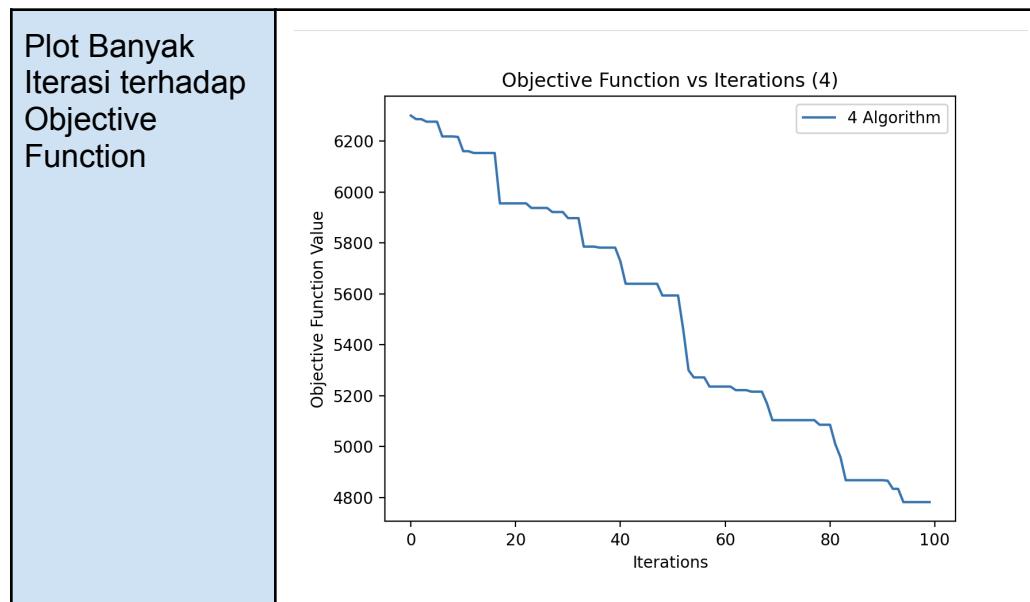
#### 4.1.4 Stochastic Hill Climbing

##### 1. Pengujian 1 - Iterasi 100

**Tabel 4.1.4.1.** Pengujian 1 Stochastic Hill Climbing

State Awal	<pre> Initial Cube State:  Layer 1:       93   114   62   47   54      110   91    41   19   55       13   105   72   111   18       38   24   115   34   67      117   21   89   28    6  Layer 2:       20   108   80   69   101       88   81    95   124   85       94   23    27   57    7       40   46   118   45   84       1   86    4    39   59  Layer 3:       90   113   9    5   70       22   12    119   48   116       74   79    51   77   71       66   29    50   35   76      125   121   50   35   76  Layer 4:       44   65   87   83   75       99   33   82   26   32       42   43   112   64   78       31   2    104   98   37       52   58   60   10   37  Layer 5:       96   109   63   103   8       36   3    17   30   68       49   92   15   25   14       73   120   106   16   100       56   107   122   11   102   </pre>
Objective Awal	6301

State Akhir	<pre> Final Cube State:  Layer 1:       93   110   29   47   54      114   91   26   19   71       10   105   98   111   18       38   24   115   34   85      117   21   89   62   35  Layer 2:       20   108   67   68   101       88   25   55   112   39      123   23   92   30   7       40   46   60   66   84       1   113   56   80   59  Layer 3:       90   86   9   5   70       22   12   119   48   73       61   75   53   77   95       45   28   51   94   74      125   121   50   6   76  Layer 4:       44   16   87   83   79       99   33   82   41   97      118   43   57   64   32       31   2   36   72   78       52   58   42   13   37  Layer 5:       96   109   63   103   8       104   3   17   124   69       4   27   15   81   14      116   120   106   65   100       49   107   122   11   102   </pre>
Objective Akhir	4782
Banyak Iterasi	100
Hasil Eksperimen	<pre> ALGORITHM RESULTS  Algorithm: Stochastic Initial Objective Value: 6301 Final Objective Value: 4782 Iterations: 100 Duration: 21.9491 seconds Stuck Counter: [] Population Size: []   </pre>
Durasi Pencarian	21.9491

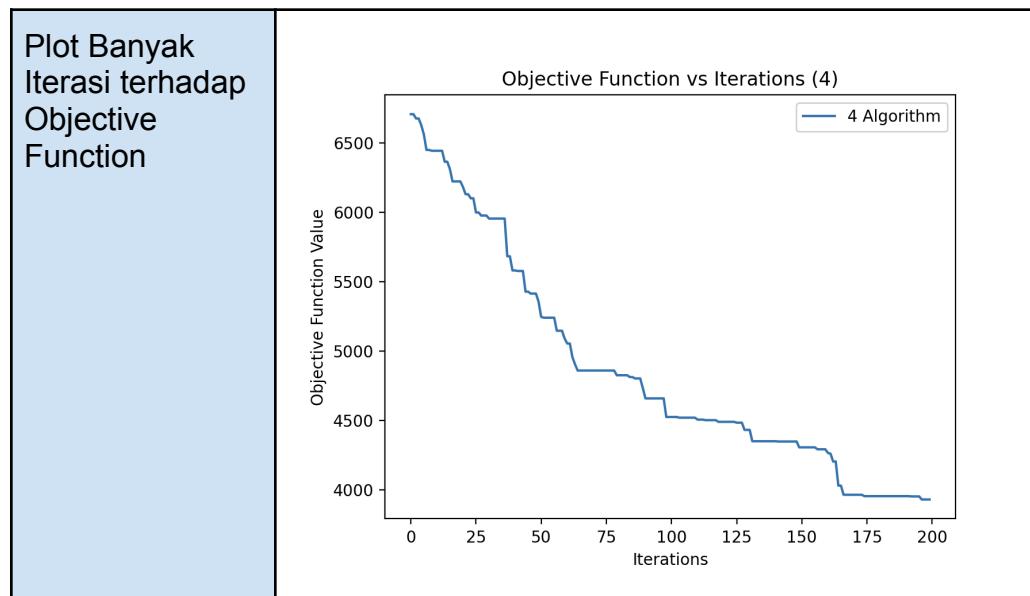


## 2. Pengujian 2 - Iterasi 200

**Tabel 4.1.4.2.** Pengujian 2 Stochastic

State Awal	<pre> Initial Cube State:  Layer 1:       47   27   48   61   118      109  112   57    3   108      116  122  101  125   84       20    64   30   37   89       71   81    1  119   80  Layer 2:       115   93   120   104   58      106   40   114    9   77       4     5    7   60   35       10   39   13   90   68       59   65   87   29   34  Layer 3:       95   23   86   107   24      26   15   72   52   111      96   43   97   42   56       6   62   63   76  123  Layer 4:       53   66   124   55   41      25   98   22   88   73      18   21   74   28   49      102   8   83   50   85       91   67   82   38  110  Layer 5:       105   121   94   14   100      113    2   44   79   117      16   45   36   19   75      69   31   99   70   32       11   92   51   78   33   </pre>
------------	--

Objective Awal	6709
State Akhir	<pre> Final Cube State: Layer 1:       47   27   31   115   119       109   112   57   3   15       116   5   58   54   19       42   64   125   1   89       71   123   23   87   80  Layer 2:       62   93   41   39   76       69   55   114   9   98       67   60   7   121   35       10   99   13   90   85       102   21   118   30   34  Layer 3:       95   122   86   107   24       26   38   72   82   111       96   44   97   32   43       77   91   103   16   46       11   61   63   101   81  Layer 4:       53   66   124   49   8       25   79   22   88   29       18   65   74   17   73       110   37   83   50   2       120   4   51   108   100  Layer 5:       105   40   94   14   59       113   68   56   84   117       28   45   36   104   75       106   48   12   70   20       6   92   52   78   33   </pre>
Objective Akhir	3930
Banyak Iterasi	200
Hasil Eksperimen	<pre> ALGORITHM RESULTS  Algorithm: Stochastic Initial Objective Value: 6709 Final Objective Value: 3930 Iterations: 200 Duration: 44.3460 seconds Stuck Counter: [] Population Size: []   </pre>
Durasi Pencarian	44.3460

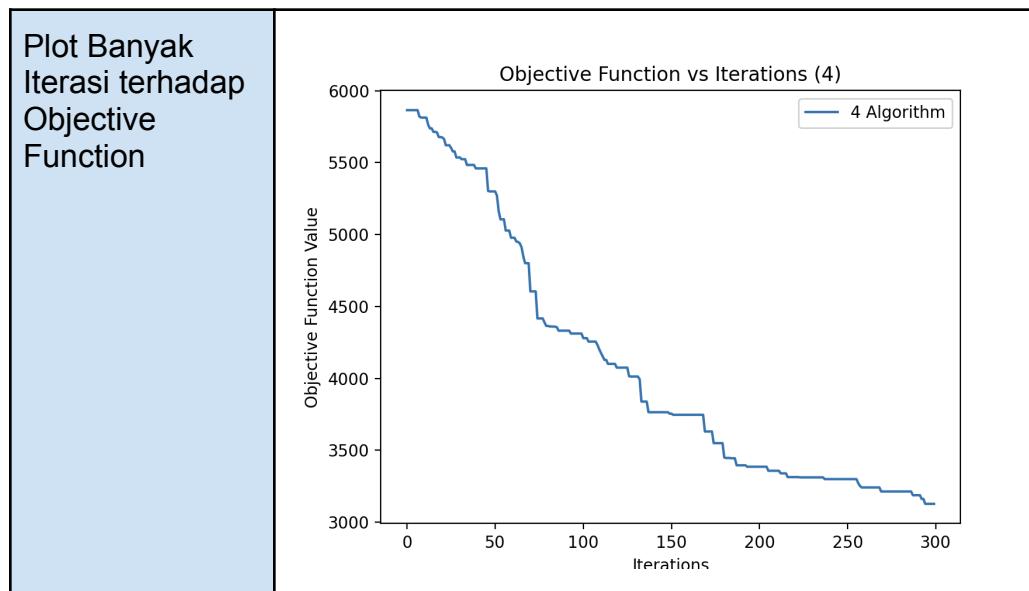


### 3. Pengujian 3 - Iterasi 300

**Tabel 4.1.4.3.** Pengujian 3 Stochastic

State Awal	<pre> Layer 1:           Initial Cube State:                 93   61   101   35   64                 90   53   69    14   106                 56   125  29    84   70                 55   32   23    60   81  Layer 2:                 22   10   109   112  12                 72   44   66    67   114                 31   78   121   94   118                 83   11   79    41   27                 43   115  85    40   105  Layer 3:                 87   28   50    15   123                 119  122  98    33   3                 75   26   73    74   34                 21   86   36    71   19                 48   65   76    107  24  Layer 4:                 16   80   103   9    7                 51   113  58    47   92                 57   5    62    20   30                 38   25   108   100  91                 110  68   1    77   49  Layer 5:                 42   96   117   89   99                   2   116  111   46   97                 17   63   59    104  95                 18   45   4     6    102                 13   88   54    120  52 </pre>
Objective Awal	5865

State Akhir	<pre> Final Cube State: Layer 1:       57   70   90   25   64       97   28   62   14   106       37   83   18   84   123       19   117   98   122   1 103   32    5   104   81  Layer 2:       10   22   124   110   20       72   77    2   89   114       26   39   86   94   34       111   115   101   3   96 119   115   101   3   96  Layer 3:       87   95   38   15   59       82   53   79   33   40       21   31   69   45   23       75   121   36   71   56       48   44   93   100   35 112   80   65    8   49  Layer 4:       16   24    7   125   74       51   113   58   47   30       109   60   73   12   91       50   29   88   68   49       112   80   65    8   49  Layer 5:       13   105   43   67   55       66   116   108   46   120       99   42   118   85   102       107   17    4    6   52       63    9   54   92   52 </pre>
Objective Akhir	3128
Banyak Iterasi	300
Hasil Eksperimen	<pre> ALGORITHM RESULTS  Algorithm: Stochastic Initial Objective Value: 5865 Final Objective Value: 3128 Iterations: 300 Duration: 70.6912 seconds Stuck Counter: [] Population Size: [] </pre>
Durasi Pencarian	70.6912 detik



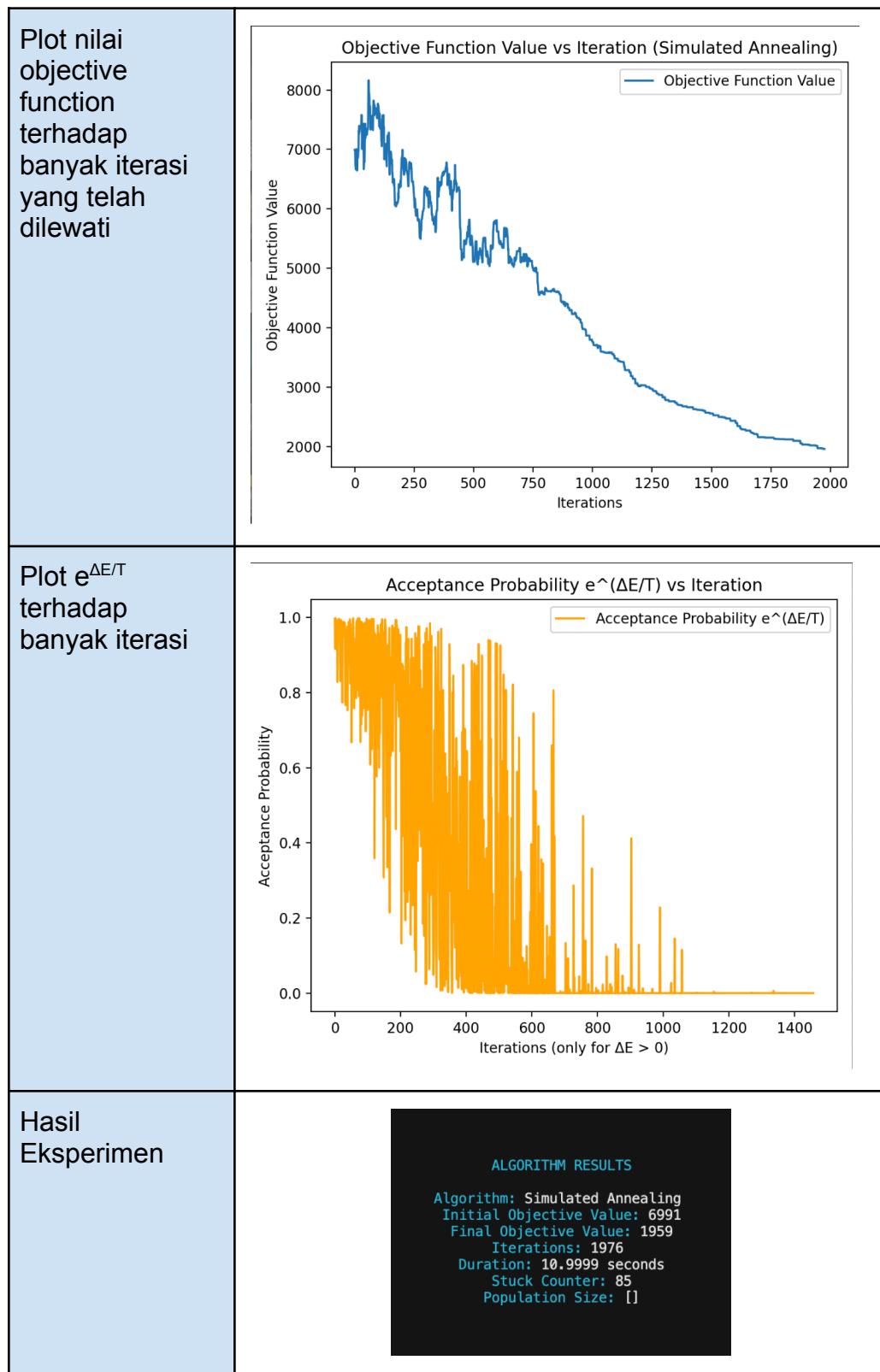
## 4.2 Simulated Annealing

### 1. Pengujian 1

**Tabel 4.2.1.** Pengujian 1 Simulated Annealing

Temperatur awal	2000
Cooling rate	0.995
Minimum temperature	0.1
Iterasi maksimum	10000

State Awal	<p><b>Initial Cube State:</b></p> <table border="0"> <tr><td>104</td><td>85</td><td>7</td><td>113</td><td>61</td></tr> <tr><td>54</td><td>116</td><td>20</td><td>53</td><td>51</td></tr> <tr><td>39</td><td>98</td><td>88</td><td>8</td><td>63</td></tr> <tr><td>5</td><td>108</td><td>28</td><td>55</td><td>27</td></tr> <tr><td>49</td><td>21</td><td>9</td><td>106</td><td>34</td></tr> </table> <p><b>Layer 2:</b></p> <table border="0"> <tr><td>58</td><td>35</td><td>119</td><td>32</td><td>71</td></tr> <tr><td>125</td><td>57</td><td>87</td><td>124</td><td>67</td></tr> <tr><td>101</td><td>62</td><td>105</td><td>38</td><td>41</td></tr> <tr><td>18</td><td>89</td><td>13</td><td>59</td><td>4</td></tr> <tr><td>66</td><td>19</td><td>70</td><td>12</td><td>117</td></tr> </table> <p><b>Layer 3:</b></p> <table border="0"> <tr><td>111</td><td>79</td><td>99</td><td>81</td><td>118</td></tr> <tr><td>17</td><td>121</td><td>15</td><td>6</td><td>52</td></tr> <tr><td>97</td><td>11</td><td>29</td><td>43</td><td>107</td></tr> <tr><td>31</td><td>80</td><td>102</td><td>82</td><td>95</td></tr> <tr><td>77</td><td>58</td><td>48</td><td>14</td><td>91</td></tr> </table> <p><b>Layer 4:</b></p> <table border="0"> <tr><td>122</td><td>40</td><td>46</td><td>45</td><td>103</td></tr> <tr><td>22</td><td>23</td><td>1</td><td>44</td><td>42</td></tr> <tr><td>109</td><td>37</td><td>72</td><td>120</td><td>64</td></tr> <tr><td>90</td><td>123</td><td>112</td><td>96</td><td>75</td></tr> <tr><td>92</td><td>25</td><td>73</td><td>30</td><td>100</td></tr> </table> <p><b>Layer 5:</b></p> <table border="0"> <tr><td>114</td><td>68</td><td>26</td><td>24</td><td>83</td></tr> <tr><td>2</td><td>115</td><td>36</td><td>110</td><td>66</td></tr> <tr><td>86</td><td>33</td><td>56</td><td>10</td><td>47</td></tr> <tr><td>3</td><td>78</td><td>76</td><td>69</td><td>16</td></tr> <tr><td>93</td><td>74</td><td>94</td><td>84</td><td>65</td></tr> </table>	104	85	7	113	61	54	116	20	53	51	39	98	88	8	63	5	108	28	55	27	49	21	9	106	34	58	35	119	32	71	125	57	87	124	67	101	62	105	38	41	18	89	13	59	4	66	19	70	12	117	111	79	99	81	118	17	121	15	6	52	97	11	29	43	107	31	80	102	82	95	77	58	48	14	91	122	40	46	45	103	22	23	1	44	42	109	37	72	120	64	90	123	112	96	75	92	25	73	30	100	114	68	26	24	83	2	115	36	110	66	86	33	56	10	47	3	78	76	69	16	93	74	94	84	65
104	85	7	113	61																																																																																																																										
54	116	20	53	51																																																																																																																										
39	98	88	8	63																																																																																																																										
5	108	28	55	27																																																																																																																										
49	21	9	106	34																																																																																																																										
58	35	119	32	71																																																																																																																										
125	57	87	124	67																																																																																																																										
101	62	105	38	41																																																																																																																										
18	89	13	59	4																																																																																																																										
66	19	70	12	117																																																																																																																										
111	79	99	81	118																																																																																																																										
17	121	15	6	52																																																																																																																										
97	11	29	43	107																																																																																																																										
31	80	102	82	95																																																																																																																										
77	58	48	14	91																																																																																																																										
122	40	46	45	103																																																																																																																										
22	23	1	44	42																																																																																																																										
109	37	72	120	64																																																																																																																										
90	123	112	96	75																																																																																																																										
92	25	73	30	100																																																																																																																										
114	68	26	24	83																																																																																																																										
2	115	36	110	66																																																																																																																										
86	33	56	10	47																																																																																																																										
3	78	76	69	16																																																																																																																										
93	74	94	84	65																																																																																																																										
Objective Awal	6991																																																																																																																													
State Akhir	1959																																																																																																																													
Objective Akhir	<p><b>Final Cube State:</b></p> <table border="0"> <tr><td>88</td><td>96</td><td>12</td><td>32</td><td>81</td></tr> <tr><td>3</td><td>2</td><td>125</td><td>103</td><td>83</td></tr> <tr><td>85</td><td>56</td><td>63</td><td>5</td><td>91</td></tr> <tr><td>73</td><td>1</td><td>84</td><td>118</td><td>18</td></tr> <tr><td>82</td><td>113</td><td>31</td><td>79</td><td>58</td></tr> </table> <p><b>Layer 2:</b></p> <table border="0"> <tr><td>59</td><td>60</td><td>53</td><td>86</td><td>66</td></tr> <tr><td>196</td><td>28</td><td>46</td><td>111</td><td>13</td></tr> <tr><td>17</td><td>105</td><td>65</td><td>24</td><td>57</td></tr> <tr><td>8</td><td>21</td><td>112</td><td>70</td><td>92</td></tr> <tr><td>123</td><td>54</td><td>36</td><td>26</td><td>90</td></tr> </table> <p><b>Layer 3:</b></p> <table border="0"> <tr><td>35</td><td>64</td><td>99</td><td>48</td><td>90</td></tr> <tr><td>101</td><td>117</td><td>18</td><td>119</td><td>115</td></tr> <tr><td>43</td><td>25</td><td>71</td><td>78</td><td>108</td></tr> <tr><td>121</td><td>30</td><td>34</td><td>52</td><td>108</td></tr> <tr><td>9</td><td>110</td><td>114</td><td>22</td><td>42</td></tr> </table> <p><b>Layer 4:</b></p> <table border="0"> <tr><td>23</td><td>49</td><td>107</td><td>102</td><td>74</td></tr> <tr><td>44</td><td>95</td><td>47</td><td>102</td><td>93</td></tr> <tr><td>120</td><td>68</td><td>11</td><td>39</td><td>33</td></tr> <tr><td>69</td><td>104</td><td>40</td><td>72</td><td>33</td></tr> <tr><td>29</td><td>7</td><td>97</td><td>51</td><td>116</td></tr> </table> <p><b>Layer 5:</b></p> <table border="0"> <tr><td>109</td><td>45</td><td>58</td><td>76</td><td>37</td></tr> <tr><td>55</td><td>87</td><td>27</td><td>19</td><td>124</td></tr> <tr><td>20</td><td>75</td><td>98</td><td>80</td><td>41</td></tr> <tr><td>38</td><td>100</td><td>94</td><td>6</td><td>77</td></tr> <tr><td>61</td><td>15</td><td>67</td><td>122</td><td>16</td></tr> </table>	88	96	12	32	81	3	2	125	103	83	85	56	63	5	91	73	1	84	118	18	82	113	31	79	58	59	60	53	86	66	196	28	46	111	13	17	105	65	24	57	8	21	112	70	92	123	54	36	26	90	35	64	99	48	90	101	117	18	119	115	43	25	71	78	108	121	30	34	52	108	9	110	114	22	42	23	49	107	102	74	44	95	47	102	93	120	68	11	39	33	69	104	40	72	33	29	7	97	51	116	109	45	58	76	37	55	87	27	19	124	20	75	98	80	41	38	100	94	6	77	61	15	67	122	16
88	96	12	32	81																																																																																																																										
3	2	125	103	83																																																																																																																										
85	56	63	5	91																																																																																																																										
73	1	84	118	18																																																																																																																										
82	113	31	79	58																																																																																																																										
59	60	53	86	66																																																																																																																										
196	28	46	111	13																																																																																																																										
17	105	65	24	57																																																																																																																										
8	21	112	70	92																																																																																																																										
123	54	36	26	90																																																																																																																										
35	64	99	48	90																																																																																																																										
101	117	18	119	115																																																																																																																										
43	25	71	78	108																																																																																																																										
121	30	34	52	108																																																																																																																										
9	110	114	22	42																																																																																																																										
23	49	107	102	74																																																																																																																										
44	95	47	102	93																																																																																																																										
120	68	11	39	33																																																																																																																										
69	104	40	72	33																																																																																																																										
29	7	97	51	116																																																																																																																										
109	45	58	76	37																																																																																																																										
55	87	27	19	124																																																																																																																										
20	75	98	80	41																																																																																																																										
38	100	94	6	77																																																																																																																										
61	15	67	122	16																																																																																																																										

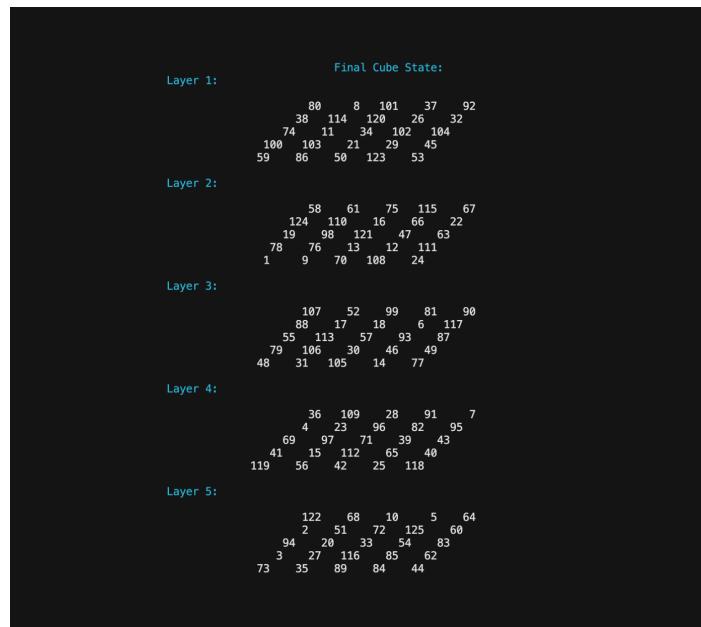
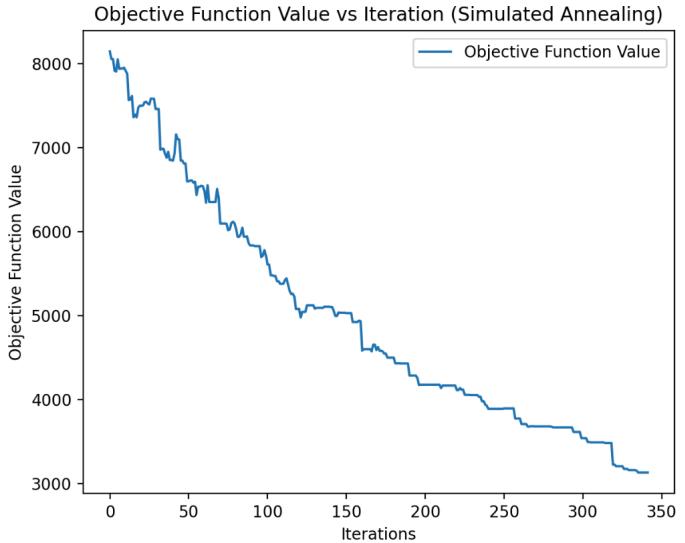


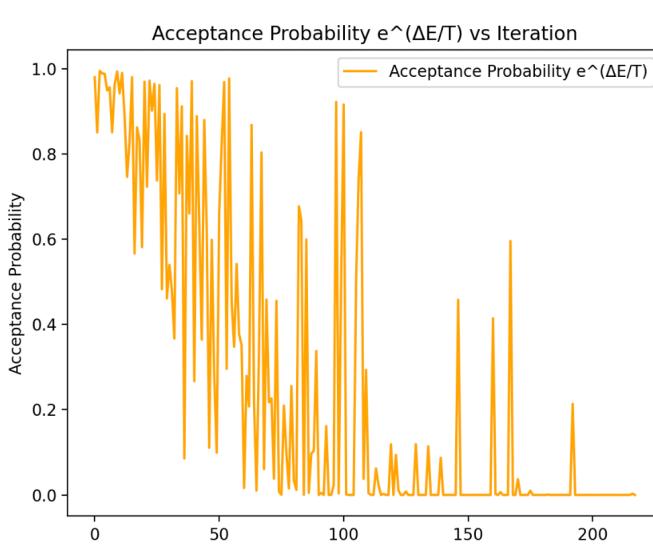
Banyak Iterasi	1976
Durasi Pencarian	10.999 detik
Jumlah Stuck	85

## 2. Pengujian 2

**Tabel 4.2.2.** Pengujian 2 Simulated Annealing

Temperatur awal	1000																																																																																																																																																						
Cooling rate	0.98																																																																																																																																																						
Minimum temperature	1																																																																																																																																																						
Iterasi maksimum	10000																																																																																																																																																						
State Awal	<p style="text-align: center;">Initial Cube State:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>104</td><td>85</td><td>7</td><td>113</td><td>61</td></tr> <tr><td>54</td><td>116</td><td>20</td><td>53</td><td>51</td></tr> <tr><td>39</td><td>98</td><td>88</td><td>8</td><td>63</td></tr> <tr><td>5</td><td>108</td><td>28</td><td>55</td><td>27</td></tr> <tr><td>49</td><td>21</td><td>9</td><td>106</td><td>34</td></tr> </table> <p style="text-align: center;">Layer 1:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>58</td><td>35</td><td>119</td><td>32</td><td>71</td></tr> <tr><td>125</td><td>57</td><td>87</td><td>124</td><td>67</td></tr> <tr><td>101</td><td>62</td><td>105</td><td>38</td><td>41</td></tr> <tr><td>18</td><td>89</td><td>13</td><td>59</td><td>4</td></tr> <tr><td>66</td><td>19</td><td>70</td><td>12</td><td>117</td></tr> </table> <p style="text-align: center;">Layer 2:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>111</td><td>79</td><td>99</td><td>81</td><td>118</td></tr> <tr><td>17</td><td>121</td><td>15</td><td>6</td><td>52</td></tr> <tr><td>97</td><td>11</td><td>29</td><td>43</td><td>107</td></tr> <tr><td>31</td><td>80</td><td>102</td><td>82</td><td>95</td></tr> <tr><td>77</td><td>50</td><td>48</td><td>14</td><td>91</td></tr> </table> <p style="text-align: center;">Layer 3:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>122</td><td>40</td><td>46</td><td>45</td><td>103</td></tr> <tr><td>22</td><td>23</td><td>72</td><td>1</td><td>44</td></tr> <tr><td>189</td><td>37</td><td>112</td><td>120</td><td>64</td></tr> <tr><td>90</td><td>123</td><td>30</td><td>100</td><td>75</td></tr> <tr><td>92</td><td>25</td><td>73</td><td>96</td><td>10</td></tr> </table> <p style="text-align: center;">Layer 4:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>114</td><td>68</td><td>26</td><td>24</td><td>83</td></tr> <tr><td>2</td><td>115</td><td>36</td><td>110</td><td>60</td></tr> <tr><td>86</td><td>33</td><td>56</td><td>10</td><td>47</td></tr> <tr><td>3</td><td>78</td><td>76</td><td>69</td><td>16</td></tr> <tr><td>93</td><td>74</td><td>94</td><td>84</td><td>65</td></tr> </table> <p style="text-align: center;">Layer 5:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>114</td><td>68</td><td>26</td><td>24</td><td>83</td></tr> <tr><td>2</td><td>115</td><td>36</td><td>110</td><td>60</td></tr> <tr><td>86</td><td>33</td><td>56</td><td>10</td><td>47</td></tr> <tr><td>3</td><td>78</td><td>76</td><td>69</td><td>16</td></tr> <tr><td>93</td><td>74</td><td>94</td><td>84</td><td>65</td></tr> </table>	104	85	7	113	61	54	116	20	53	51	39	98	88	8	63	5	108	28	55	27	49	21	9	106	34	58	35	119	32	71	125	57	87	124	67	101	62	105	38	41	18	89	13	59	4	66	19	70	12	117	111	79	99	81	118	17	121	15	6	52	97	11	29	43	107	31	80	102	82	95	77	50	48	14	91	122	40	46	45	103	22	23	72	1	44	189	37	112	120	64	90	123	30	100	75	92	25	73	96	10	114	68	26	24	83	2	115	36	110	60	86	33	56	10	47	3	78	76	69	16	93	74	94	84	65	114	68	26	24	83	2	115	36	110	60	86	33	56	10	47	3	78	76	69	16	93	74	94	84	65
104	85	7	113	61																																																																																																																																																			
54	116	20	53	51																																																																																																																																																			
39	98	88	8	63																																																																																																																																																			
5	108	28	55	27																																																																																																																																																			
49	21	9	106	34																																																																																																																																																			
58	35	119	32	71																																																																																																																																																			
125	57	87	124	67																																																																																																																																																			
101	62	105	38	41																																																																																																																																																			
18	89	13	59	4																																																																																																																																																			
66	19	70	12	117																																																																																																																																																			
111	79	99	81	118																																																																																																																																																			
17	121	15	6	52																																																																																																																																																			
97	11	29	43	107																																																																																																																																																			
31	80	102	82	95																																																																																																																																																			
77	50	48	14	91																																																																																																																																																			
122	40	46	45	103																																																																																																																																																			
22	23	72	1	44																																																																																																																																																			
189	37	112	120	64																																																																																																																																																			
90	123	30	100	75																																																																																																																																																			
92	25	73	96	10																																																																																																																																																			
114	68	26	24	83																																																																																																																																																			
2	115	36	110	60																																																																																																																																																			
86	33	56	10	47																																																																																																																																																			
3	78	76	69	16																																																																																																																																																			
93	74	94	84	65																																																																																																																																																			
114	68	26	24	83																																																																																																																																																			
2	115	36	110	60																																																																																																																																																			
86	33	56	10	47																																																																																																																																																			
3	78	76	69	16																																																																																																																																																			
93	74	94	84	65																																																																																																																																																			
Objective Awal	8125																																																																																																																																																						

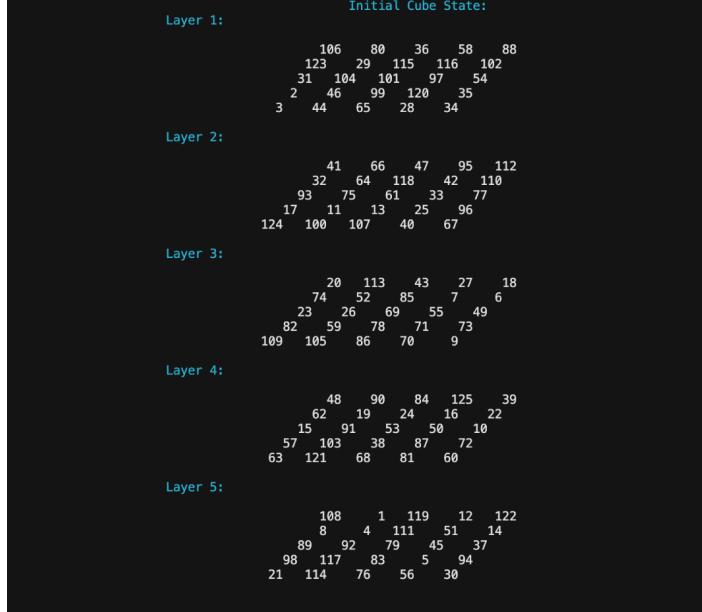
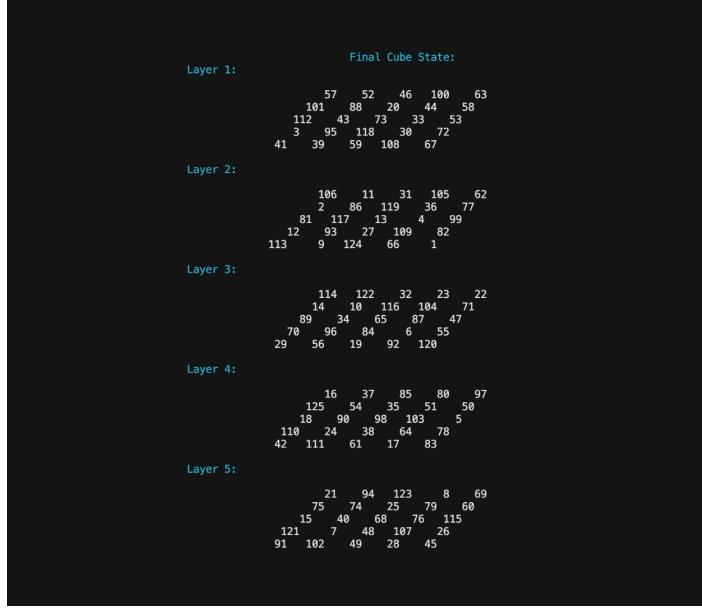
State Akhir																			
Objective Akhir	3134																		
Plot nilai objective function terhadap banyak iterasi yang telah dilewati	 <p>The graph plots the Objective Function Value against Iterations. The x-axis represents Iterations from 0 to 350, and the y-axis represents Objective Function Value from 3000 to 8000. The curve shows a general downward trend, indicating improvement in the objective function value over time.</p> <table border="1"><caption>Data points estimated from the Objective Function Value vs Iteration graph</caption><thead><tr><th>Iterations</th><th>Objective Function Value</th></tr></thead><tbody><tr><td>0</td><td>8000</td></tr><tr><td>50</td><td>6500</td></tr><tr><td>100</td><td>5500</td></tr><tr><td>150</td><td>5000</td></tr><tr><td>200</td><td>4500</td></tr><tr><td>250</td><td>4000</td></tr><tr><td>300</td><td>3500</td></tr><tr><td>350</td><td>3200</td></tr></tbody></table>	Iterations	Objective Function Value	0	8000	50	6500	100	5500	150	5000	200	4500	250	4000	300	3500	350	3200
Iterations	Objective Function Value																		
0	8000																		
50	6500																		
100	5500																		
150	5000																		
200	4500																		
250	4000																		
300	3500																		
350	3200																		

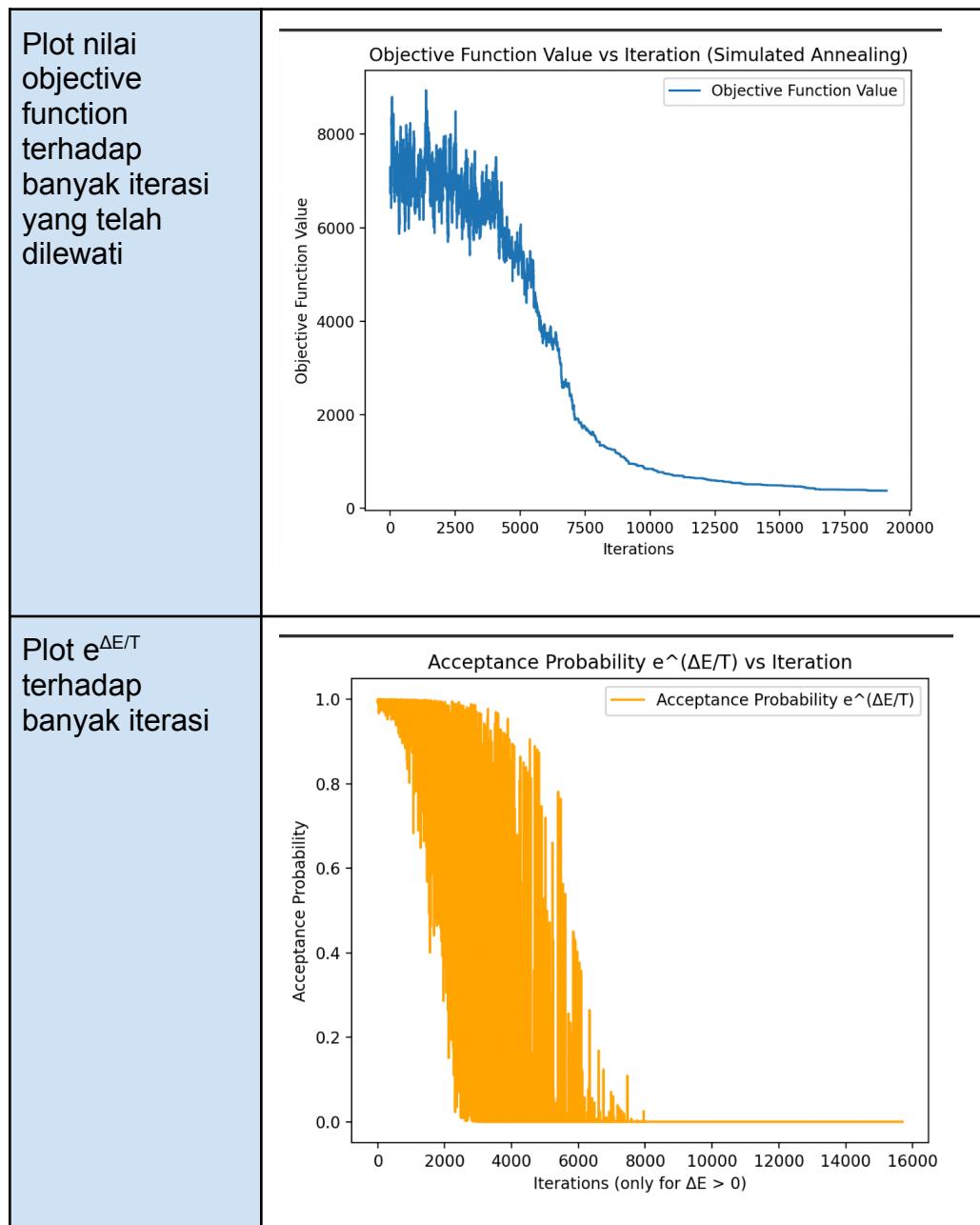
Plot $e^{\Delta E/T}$ terhadap banyak iterasi	
Hasil Eksperimen	<pre> ALGORITHM RESULTS Algorithm: Simulated Annealing Initial Objective Value: 8125 Final Objective Value: 3134 Iterations: 342 Duration: 2.0781 seconds Stuck Counter: 19 Population Size: [] </pre>
Banyak Iterasi	342
Durasi Pencarian	2.0781
Jumlah Stuck	19

### 3. Pengujian 3

**Tabel 4.2.3.** Pengujian 3 Simulated Annealing

Temperatur awal	20000
Cooling rate	0.999
Minimum temperature	0.0001

Iterasi maksimum	50000																																																																																																																													
State Awal	<p style="text-align: center;">Initial Cube State:</p>  <p>Layer 1:</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr><td>106</td><td>80</td><td>36</td><td>58</td><td>88</td></tr> <tr><td>123</td><td>29</td><td>115</td><td>116</td><td>102</td></tr> <tr><td>31</td><td>104</td><td>101</td><td>97</td><td>54</td></tr> <tr><td>2</td><td>46</td><td>99</td><td>120</td><td>35</td></tr> <tr><td>3</td><td>44</td><td>65</td><td>28</td><td>34</td></tr> </table> <p>Layer 2:</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr><td>41</td><td>66</td><td>47</td><td>95</td><td>112</td></tr> <tr><td>32</td><td>64</td><td>118</td><td>42</td><td>110</td></tr> <tr><td>93</td><td>75</td><td>61</td><td>33</td><td>77</td></tr> <tr><td>17</td><td>11</td><td>13</td><td>25</td><td>96</td></tr> <tr><td>124</td><td>100</td><td>107</td><td>48</td><td>67</td></tr> </table> <p>Layer 3:</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr><td>20</td><td>113</td><td>43</td><td>27</td><td>18</td></tr> <tr><td>74</td><td>52</td><td>85</td><td>7</td><td>6</td></tr> <tr><td>23</td><td>26</td><td>69</td><td>55</td><td>49</td></tr> <tr><td>82</td><td>59</td><td>78</td><td>71</td><td>73</td></tr> <tr><td>109</td><td>105</td><td>86</td><td>70</td><td>9</td></tr> </table> <p>Layer 4:</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr><td>48</td><td>90</td><td>84</td><td>125</td><td>39</td></tr> <tr><td>62</td><td>19</td><td>24</td><td>16</td><td>22</td></tr> <tr><td>15</td><td>91</td><td>53</td><td>50</td><td>10</td></tr> <tr><td>57</td><td>103</td><td>38</td><td>87</td><td>72</td></tr> <tr><td>63</td><td>121</td><td>68</td><td>81</td><td>60</td></tr> </table> <p>Layer 5:</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr><td>108</td><td>1</td><td>119</td><td>12</td><td>122</td></tr> <tr><td>8</td><td>4</td><td>111</td><td>51</td><td>14</td></tr> <tr><td>89</td><td>92</td><td>79</td><td>45</td><td>37</td></tr> <tr><td>98</td><td>117</td><td>83</td><td>5</td><td>94</td></tr> <tr><td>21</td><td>114</td><td>76</td><td>56</td><td>30</td></tr> </table>	106	80	36	58	88	123	29	115	116	102	31	104	101	97	54	2	46	99	120	35	3	44	65	28	34	41	66	47	95	112	32	64	118	42	110	93	75	61	33	77	17	11	13	25	96	124	100	107	48	67	20	113	43	27	18	74	52	85	7	6	23	26	69	55	49	82	59	78	71	73	109	105	86	70	9	48	90	84	125	39	62	19	24	16	22	15	91	53	50	10	57	103	38	87	72	63	121	68	81	60	108	1	119	12	122	8	4	111	51	14	89	92	79	45	37	98	117	83	5	94	21	114	76	56	30
106	80	36	58	88																																																																																																																										
123	29	115	116	102																																																																																																																										
31	104	101	97	54																																																																																																																										
2	46	99	120	35																																																																																																																										
3	44	65	28	34																																																																																																																										
41	66	47	95	112																																																																																																																										
32	64	118	42	110																																																																																																																										
93	75	61	33	77																																																																																																																										
17	11	13	25	96																																																																																																																										
124	100	107	48	67																																																																																																																										
20	113	43	27	18																																																																																																																										
74	52	85	7	6																																																																																																																										
23	26	69	55	49																																																																																																																										
82	59	78	71	73																																																																																																																										
109	105	86	70	9																																																																																																																										
48	90	84	125	39																																																																																																																										
62	19	24	16	22																																																																																																																										
15	91	53	50	10																																																																																																																										
57	103	38	87	72																																																																																																																										
63	121	68	81	60																																																																																																																										
108	1	119	12	122																																																																																																																										
8	4	111	51	14																																																																																																																										
89	92	79	45	37																																																																																																																										
98	117	83	5	94																																																																																																																										
21	114	76	56	30																																																																																																																										
Objective Awal	7194																																																																																																																													
State Akhir	<p style="text-align: center;">Final Cube State:</p>  <p>Layer 1:</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr><td>57</td><td>52</td><td>46</td><td>100</td><td>63</td></tr> <tr><td>101</td><td>88</td><td>73</td><td>44</td><td>58</td></tr> <tr><td>112</td><td>43</td><td>28</td><td>33</td><td>53</td></tr> <tr><td>3</td><td>95</td><td>118</td><td>30</td><td>72</td></tr> <tr><td>41</td><td>39</td><td>59</td><td>108</td><td>67</td></tr> </table> <p>Layer 2:</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr><td>166</td><td>11</td><td>31</td><td>185</td><td>62</td></tr> <tr><td>2</td><td>86</td><td>119</td><td>36</td><td>77</td></tr> <tr><td>81</td><td>93</td><td>27</td><td>4</td><td>99</td></tr> <tr><td>12</td><td>9</td><td>124</td><td>66</td><td>1</td></tr> <tr><td>113</td><td></td><td></td><td></td><td></td></tr> </table> <p>Layer 3:</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr><td>114</td><td>122</td><td>32</td><td>23</td><td>22</td></tr> <tr><td>14</td><td>10</td><td>116</td><td>104</td><td>71</td></tr> <tr><td>89</td><td>34</td><td>65</td><td>87</td><td>47</td></tr> <tr><td>70</td><td>96</td><td>84</td><td>6</td><td>95</td></tr> <tr><td>29</td><td>56</td><td>19</td><td>92</td><td>120</td></tr> </table> <p>Layer 4:</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr><td>16</td><td>37</td><td>85</td><td>80</td><td>97</td></tr> <tr><td>125</td><td>54</td><td>35</td><td>51</td><td>50</td></tr> <tr><td>18</td><td>90</td><td>98</td><td>103</td><td>5</td></tr> <tr><td>110</td><td>24</td><td>38</td><td>64</td><td>78</td></tr> <tr><td>42</td><td>111</td><td>61</td><td>17</td><td>83</td></tr> </table> <p>Layer 5:</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr><td>21</td><td>94</td><td>123</td><td>8</td><td>69</td></tr> <tr><td>75</td><td>74</td><td>25</td><td>79</td><td>60</td></tr> <tr><td>15</td><td>40</td><td>68</td><td>76</td><td>115</td></tr> <tr><td>121</td><td>7</td><td>48</td><td>107</td><td>26</td></tr> <tr><td>91</td><td>102</td><td>49</td><td>28</td><td>45</td></tr> </table>	57	52	46	100	63	101	88	73	44	58	112	43	28	33	53	3	95	118	30	72	41	39	59	108	67	166	11	31	185	62	2	86	119	36	77	81	93	27	4	99	12	9	124	66	1	113					114	122	32	23	22	14	10	116	104	71	89	34	65	87	47	70	96	84	6	95	29	56	19	92	120	16	37	85	80	97	125	54	35	51	50	18	90	98	103	5	110	24	38	64	78	42	111	61	17	83	21	94	123	8	69	75	74	25	79	60	15	40	68	76	115	121	7	48	107	26	91	102	49	28	45
57	52	46	100	63																																																																																																																										
101	88	73	44	58																																																																																																																										
112	43	28	33	53																																																																																																																										
3	95	118	30	72																																																																																																																										
41	39	59	108	67																																																																																																																										
166	11	31	185	62																																																																																																																										
2	86	119	36	77																																																																																																																										
81	93	27	4	99																																																																																																																										
12	9	124	66	1																																																																																																																										
113																																																																																																																														
114	122	32	23	22																																																																																																																										
14	10	116	104	71																																																																																																																										
89	34	65	87	47																																																																																																																										
70	96	84	6	95																																																																																																																										
29	56	19	92	120																																																																																																																										
16	37	85	80	97																																																																																																																										
125	54	35	51	50																																																																																																																										
18	90	98	103	5																																																																																																																										
110	24	38	64	78																																																																																																																										
42	111	61	17	83																																																																																																																										
21	94	123	8	69																																																																																																																										
75	74	25	79	60																																																																																																																										
15	40	68	76	115																																																																																																																										
121	7	48	107	26																																																																																																																										
91	102	49	28	45																																																																																																																										
Objective Akhir	375																																																																																																																													



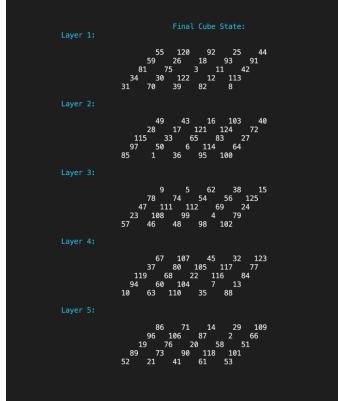
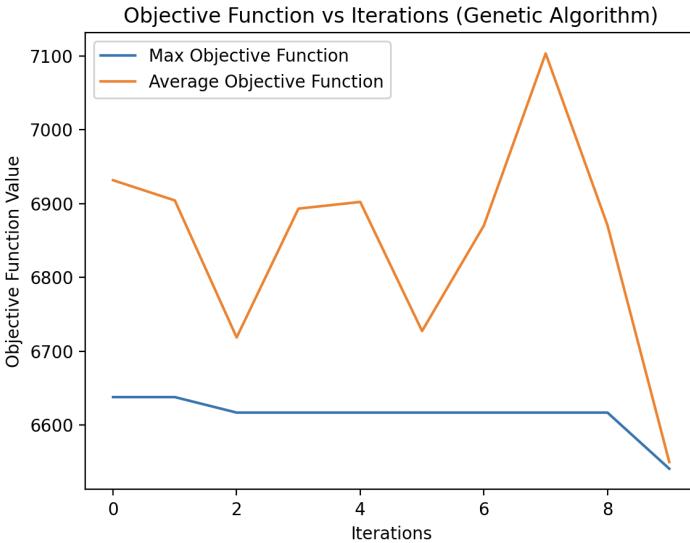
Hasil Eksperimen	<pre> ALGORITHM RESULTS  Algorithm: Simulated Annealing Initial Objective Value: 7194 Final Objective Value: 375 Iterations: 19105 Duration: 79.8124 seconds Stuck Counter: 597 Population Size: [] </pre>
Banyak Iterasi	19105
Durasi Pencarian	79.8124 detik
Jumlah Stuck	597

### 4.3 Genetic Algorithm

- Pengujian 1 - Populasi sebagai Kontrol dengan 10 Iterasi

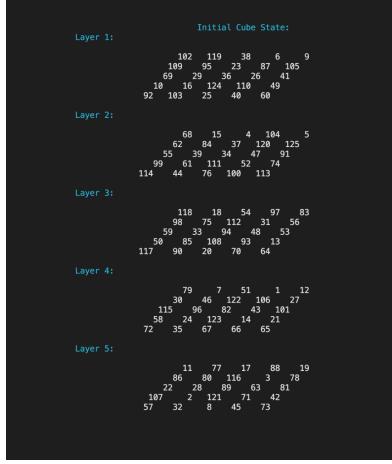
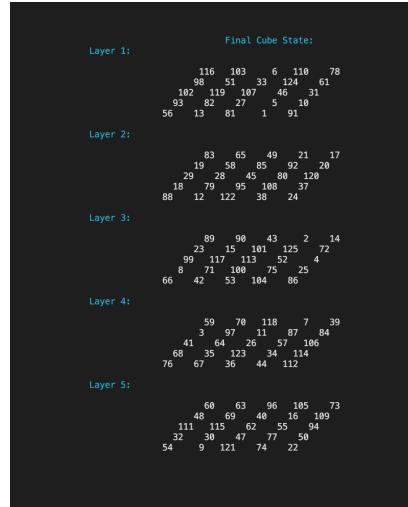
**Tabel 4.3.1.** Pengujian 1 Populasi sebagai Kontrol

State Awal	<pre> Initial Cube State:  Layer 1:       44   5   20   70   125      48   21   64   121   41       3   7   26   82   87      26   40   59   14   23       8   4   9   92   34  Layer 2:       16   42   115   29   22      47   77   78   69   93       67   114   43   111      186   124   112   86   118       63   62   84   15   81  Layer 3:       27   66   17   98   25      76   118   24   56   45       122   198   19   183   38      68   95   102   116   117       39   79  Layer 4:       91   65   189   51   99      52   108   35   94   107       104   50   57   32   49      12   82   38   11   71      96   88   85   90   120  Layer 5:       53   31   13   88   188      18   123   75   55   68       105   72   119   61      97   37   73   2   46      101   6   58   89   113 </pre>
Objective Awal	7111

State Akhir	
Objective Akhir	6541
Plot nilai objective function terhadap banyak iterasi yang telah dilewati	
Hasil Eksperimen	<pre> ALGORITHM RESULTS  Algorithm: Genetic Algorithm Initial Objective Value: 7111 Final Objective Value: 6541 Iterations: 10 Duration: 0.7574 seconds Stuck Counter: [] Population Size: 10 </pre>

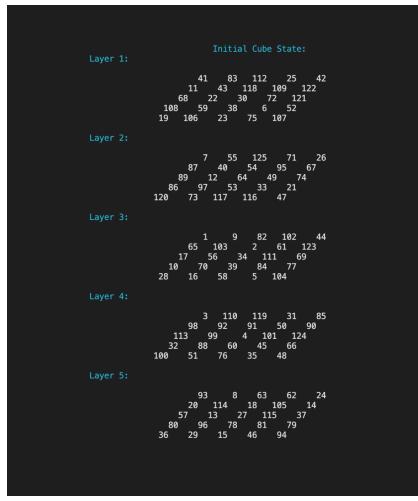
Jumlah Populasi	10
Banyak Iterasi	10
Durasi Pencarian	0.7574 detik

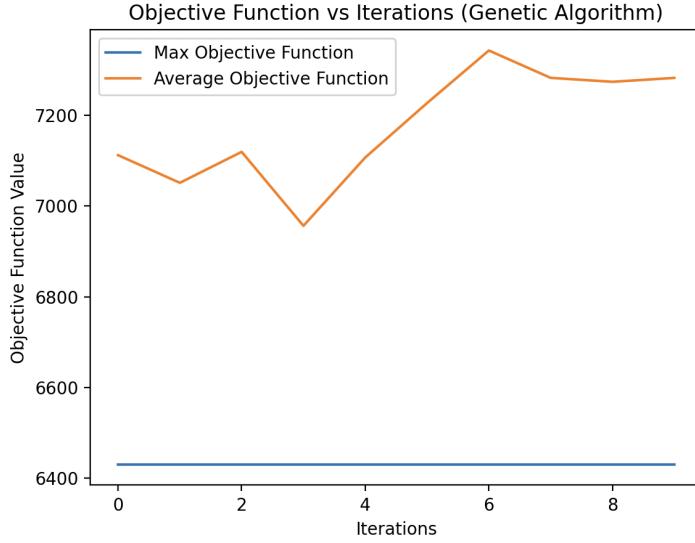
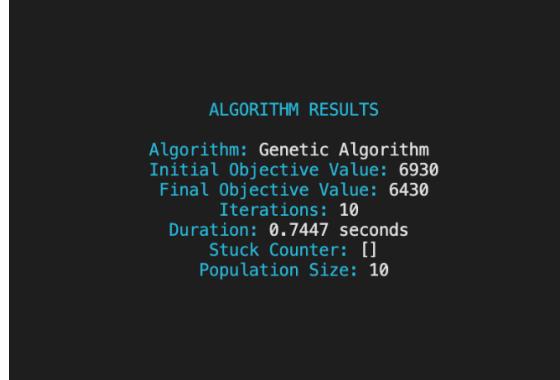
**Tabel 4.3.2.** Pengujian 1 Populasi sebagai Kontrol

State Awal	 <p>Initial Cube State:</p> <table border="1"> <tr><td>102</td><td>119</td><td>38</td><td>6</td><td>9</td></tr> <tr><td>69</td><td>29</td><td>23</td><td>87</td><td>105</td></tr> <tr><td>1</td><td>16</td><td>124</td><td>26</td><td>41</td></tr> <tr><td>92</td><td>103</td><td>25</td><td>118</td><td>49</td></tr> <tr><td>99</td><td>111</td><td>52</td><td>74</td><td>68</td></tr> </table>	102	119	38	6	9	69	29	23	87	105	1	16	124	26	41	92	103	25	118	49	99	111	52	74	68															
102	119	38	6	9																																					
69	29	23	87	105																																					
1	16	124	26	41																																					
92	103	25	118	49																																					
99	111	52	74	68																																					
Objective Awal	7133																																								
State Akhir	 <p>Final Cube State:</p> <table border="1"> <tr><td>116</td><td>103</td><td>6</td><td>110</td><td>78</td></tr> <tr><td>98</td><td>51</td><td>33</td><td>124</td><td>61</td></tr> <tr><td>102</td><td>119</td><td>187</td><td>46</td><td>31</td></tr> <tr><td>56</td><td>13</td><td>81</td><td>1</td><td>91</td></tr> <tr><td>8</td><td>77</td><td>17</td><td>88</td><td>19</td></tr> <tr><td>22</td><td>28</td><td>116</td><td>3</td><td>78</td></tr> <tr><td>187</td><td>2</td><td>121</td><td>71</td><td>42</td></tr> <tr><td>57</td><td>32</td><td>8</td><td>45</td><td>73</td></tr> </table>	116	103	6	110	78	98	51	33	124	61	102	119	187	46	31	56	13	81	1	91	8	77	17	88	19	22	28	116	3	78	187	2	121	71	42	57	32	8	45	73
116	103	6	110	78																																					
98	51	33	124	61																																					
102	119	187	46	31																																					
56	13	81	1	91																																					
8	77	17	88	19																																					
22	28	116	3	78																																					
187	2	121	71	42																																					
57	32	8	45	73																																					
Objective Akhir	6347																																								

Plot nilai objective function terhadap banyak iterasi yang telah dilewati	<p>The graph plots the Objective Function Value (Y-axis, 6400 to 7200) against Iterations (X-axis, 0 to 9). It shows two series: 'Max Objective Function' (blue line) and 'Average Objective Function' (orange line). The Max Objective Function starts at approximately 6850, remains flat until iteration 2, then drops to about 6650 by iteration 3, stays flat until iteration 5, then drops to about 6450 by iteration 7, and ends at approximately 6350. The Average Objective Function starts at approximately 7250, dips to about 7050 at iteration 1, rises to a peak of about 7280 at iteration 3, then drops sharply to about 6750 by iteration 6, fluctuates slightly between 6650 and 6750 until iteration 8, and ends at approximately 6550.</p>
Hasil Eksperimen	<p>ALGORITHM RESULTS</p> <pre>Algorithm: Genetic Algorithm Initial Objective Value: 7133 Final Objective Value: 6347 Iterations: 10 Duration: 0.6631 seconds Stuck Counter: [] Population Size: 10</pre>
Jumlah Populasi	10
Banyak Iterasi	10
Durasi Pencarian	0.6631 detik

**Tabel 4.3.3. Pengujian 1 Populasi sebagai Kontrol**

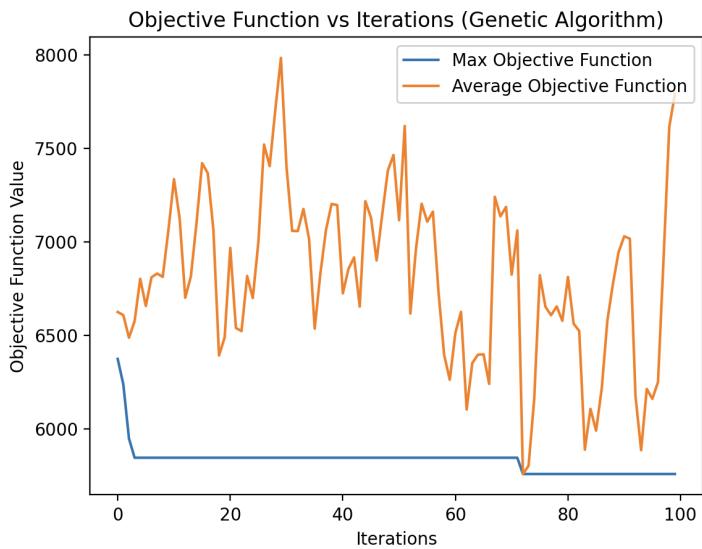
State Awal	
Objective Awal	6930
State Akhir	
Objective Akhir	6430

Plot nilai objective function terhadap banyak iterasi yang telah dilewati	
Hasil Eksperimen	 <pre>ALGORITHM RESULTS Algorithm: Genetic Algorithm Initial Objective Value: 6930 Final Objective Value: 6430 Iterations: 10 Duration: 0.7447 seconds Stuck Counter: [] Population Size: 10</pre>
Jumlah Populasi	10
Banyak Iterasi	10
Durasi Pencarian	0.7447 detik

2. Pengujian 2 - Populasi sebagai Kontrol dengan 100 Iterasi

**Tabel 4.3.4.** Pengujian 2 Populasi sebagai Kontrol

State Awal	<p><b>Initial Cube State:</b></p> <table border="0"> <tr><td>Layer 1:</td><td>9 16 118 59 52 19 10 99 109 26 102 7 89 37 38 8 21 12 79 33 87 11 46 64 106</td></tr> <tr><td>Layer 2:</td><td>43 29 3 90 65 115 50 4 72 25 36 17 83 113 125 62 81 103 6 69 98 124 56 122 5</td></tr> <tr><td>Layer 3:</td><td>110 78 76 92 2 39 84 35 86 123 23 15 30 91 108 121 18 27 100 80 104 53 54 117 20</td></tr> <tr><td>Layer 4:</td><td>97 116 73 96 47 61 111 93 49 88 41 82 66 94 45 105 112 32 57 63 60 74 1 117 22</td></tr> <tr><td>Layer 5:</td><td>58 77 67 68 44 55 40 34 114 42 120 101 51 85 13 107 70 119 95 28 31 71 14 24 48</td></tr> </table>	Layer 1:	9 16 118 59 52 19 10 99 109 26 102 7 89 37 38 8 21 12 79 33 87 11 46 64 106	Layer 2:	43 29 3 90 65 115 50 4 72 25 36 17 83 113 125 62 81 103 6 69 98 124 56 122 5	Layer 3:	110 78 76 92 2 39 84 35 86 123 23 15 30 91 108 121 18 27 100 80 104 53 54 117 20	Layer 4:	97 116 73 96 47 61 111 93 49 88 41 82 66 94 45 105 112 32 57 63 60 74 1 117 22	Layer 5:	58 77 67 68 44 55 40 34 114 42 120 101 51 85 13 107 70 119 95 28 31 71 14 24 48
Layer 1:	9 16 118 59 52 19 10 99 109 26 102 7 89 37 38 8 21 12 79 33 87 11 46 64 106										
Layer 2:	43 29 3 90 65 115 50 4 72 25 36 17 83 113 125 62 81 103 6 69 98 124 56 122 5										
Layer 3:	110 78 76 92 2 39 84 35 86 123 23 15 30 91 108 121 18 27 100 80 104 53 54 117 20										
Layer 4:	97 116 73 96 47 61 111 93 49 88 41 82 66 94 45 105 112 32 57 63 60 74 1 117 22										
Layer 5:	58 77 67 68 44 55 40 34 114 42 120 101 51 85 13 107 70 119 95 28 31 71 14 24 48										
Objective Awal	6353										
State Akhir	<p><b>Final Cube State:</b></p> <table border="0"> <tr><td>Layer 1:</td><td>57 21 123 30 119 62 78 61 113 23 181 5 50 69 35 85 18 56 100 91 77 70 125 46 118</td></tr> <tr><td>Layer 2:</td><td>31 1 19 98 27 51 10 41 75 105 7 124 192 25 88 29 53 13 122 39 47 109 24 112 46</td></tr> <tr><td>Layer 3:</td><td>55 38 8 116 84 87 86 187 95 117 79 44 60 104 36 2 114 108 37 67 120 16 76 4 74</td></tr> <tr><td>Layer 4:</td><td>33 103 92 72 15 81 45 43 115 65 32 3 68 82 63 83 71 52 110 89 54 12 9 66 26</td></tr> <tr><td>Layer 5:</td><td>58 99 94 17 42 106 6 20 64 22 90 28 14 96 80 11 111 49 121 59 48 34 73 93 97</td></tr> </table>	Layer 1:	57 21 123 30 119 62 78 61 113 23 181 5 50 69 35 85 18 56 100 91 77 70 125 46 118	Layer 2:	31 1 19 98 27 51 10 41 75 105 7 124 192 25 88 29 53 13 122 39 47 109 24 112 46	Layer 3:	55 38 8 116 84 87 86 187 95 117 79 44 60 104 36 2 114 108 37 67 120 16 76 4 74	Layer 4:	33 103 92 72 15 81 45 43 115 65 32 3 68 82 63 83 71 52 110 89 54 12 9 66 26	Layer 5:	58 99 94 17 42 106 6 20 64 22 90 28 14 96 80 11 111 49 121 59 48 34 73 93 97
Layer 1:	57 21 123 30 119 62 78 61 113 23 181 5 50 69 35 85 18 56 100 91 77 70 125 46 118										
Layer 2:	31 1 19 98 27 51 10 41 75 105 7 124 192 25 88 29 53 13 122 39 47 109 24 112 46										
Layer 3:	55 38 8 116 84 87 86 187 95 117 79 44 60 104 36 2 114 108 37 67 120 16 76 4 74										
Layer 4:	33 103 92 72 15 81 45 43 115 65 32 3 68 82 63 83 71 52 110 89 54 12 9 66 26										
Layer 5:	58 99 94 17 42 106 6 20 64 22 90 28 14 96 80 11 111 49 121 59 48 34 73 93 97										
Objective Akhir	5758										

Plot nilai objective function terhadap banyak iterasi yang telah dilewati	 <p>The plot shows two data series: 'Max Objective Function' (blue line) and 'Average Objective Function' (orange line). The x-axis represents 'Iterations' from 0 to 100, and the y-axis represents 'Objective Function Value' from 6000 to 8000. The blue line starts at approximately 6400 and quickly drops to around 5900, remaining relatively flat until iteration 75. The orange line starts at approximately 6600, fluctuates between 6500 and 7500, and shows a general downward trend with some fluctuations, ending at approximately 7800.</p>
Hasil Eksperimen	<p style="text-align: center;"><b>ALGORITHM RESULTS</b></p> <pre>Algorithm: Genetic Algorithm Initial Objective Value: 6353 Final Objective Value: 5758 Iterations: 100 Duration: 7.7051 seconds Stuck Counter: [] Population Size: 10</pre>
Jumlah Populasi	10
Banyak Iterasi	100
Durasi Pencarian	7.7051 detik

**Tabel 4.3.5.** Pengujian 2 Populasi sebagai Kontrol

State Awal	<p style="text-align: center;">Initial Cube State:</p> <p>Layer 1:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>24</td><td>89</td><td>42</td><td>94</td><td>15</td></tr> <tr><td>29</td><td>64</td><td>108</td><td>10</td><td>113</td></tr> <tr><td>121</td><td>55</td><td>88</td><td>3</td><td>49</td></tr> <tr><td>33</td><td>1</td><td>44</td><td>86</td><td>58</td></tr> <tr><td>97</td><td>105</td><td>87</td><td>123</td><td>45</td></tr> </table> <p>Layer 2:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>6</td><td>90</td><td>31</td><td>53</td><td>50</td></tr> <tr><td>32</td><td>67</td><td>12</td><td>35</td><td>28</td></tr> <tr><td>11</td><td>85</td><td>115</td><td>59</td><td>81</td></tr> <tr><td>95</td><td>16</td><td>98</td><td>112</td><td>56</td></tr> <tr><td>102</td><td>48</td><td>103</td><td>116</td><td>38</td></tr> </table> <p>Layer 3:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>71</td><td>65</td><td>30</td><td>82</td><td>114</td></tr> <tr><td>5</td><td>77</td><td>75</td><td>109</td><td>41</td></tr> <tr><td>69</td><td>72</td><td>110</td><td>34</td><td>84</td></tr> <tr><td>52</td><td>37</td><td>125</td><td>23</td><td></td></tr> <tr><td>70</td><td>99</td><td>106</td><td>60</td><td>22</td></tr> </table> <p>Layer 4:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>43</td><td>19</td><td>122</td><td>46</td><td>26</td></tr> <tr><td>14</td><td>20</td><td>101</td><td>92</td><td>96</td></tr> <tr><td>2</td><td>111</td><td>78</td><td>36</td><td>117</td></tr> <tr><td>47</td><td>54</td><td>119</td><td>120</td><td>107</td></tr> <tr><td>62</td><td></td><td></td><td></td><td></td></tr> </table> <p>Layer 5:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>118</td><td>8</td><td>68</td><td>100</td><td>17</td></tr> <tr><td>21</td><td>7</td><td>63</td><td>40</td><td>93</td></tr> <tr><td>18</td><td>61</td><td>73</td><td>57</td><td>27</td></tr> <tr><td>79</td><td>104</td><td>25</td><td>9</td><td>124</td></tr> <tr><td>66</td><td>91</td><td>39</td><td>51</td><td>83</td></tr> </table>	24	89	42	94	15	29	64	108	10	113	121	55	88	3	49	33	1	44	86	58	97	105	87	123	45	6	90	31	53	50	32	67	12	35	28	11	85	115	59	81	95	16	98	112	56	102	48	103	116	38	71	65	30	82	114	5	77	75	109	41	69	72	110	34	84	52	37	125	23		70	99	106	60	22	43	19	122	46	26	14	20	101	92	96	2	111	78	36	117	47	54	119	120	107	62					118	8	68	100	17	21	7	63	40	93	18	61	73	57	27	79	104	25	9	124	66	91	39	51	83
24	89	42	94	15																																																																																																																										
29	64	108	10	113																																																																																																																										
121	55	88	3	49																																																																																																																										
33	1	44	86	58																																																																																																																										
97	105	87	123	45																																																																																																																										
6	90	31	53	50																																																																																																																										
32	67	12	35	28																																																																																																																										
11	85	115	59	81																																																																																																																										
95	16	98	112	56																																																																																																																										
102	48	103	116	38																																																																																																																										
71	65	30	82	114																																																																																																																										
5	77	75	109	41																																																																																																																										
69	72	110	34	84																																																																																																																										
52	37	125	23																																																																																																																											
70	99	106	60	22																																																																																																																										
43	19	122	46	26																																																																																																																										
14	20	101	92	96																																																																																																																										
2	111	78	36	117																																																																																																																										
47	54	119	120	107																																																																																																																										
62																																																																																																																														
118	8	68	100	17																																																																																																																										
21	7	63	40	93																																																																																																																										
18	61	73	57	27																																																																																																																										
79	104	25	9	124																																																																																																																										
66	91	39	51	83																																																																																																																										
Objective Awal	6632																																																																																																																													

State Akhir	<p>Final Cube State:</p> <p>Layer 1:</p> <table border="1"><tr><td>4</td><td>100</td><td>113</td><td>36</td><td>34</td></tr><tr><td>65</td><td>64</td><td>38</td><td>42</td><td>30</td></tr><tr><td>57</td><td>72</td><td>122</td><td>62</td><td>115</td></tr><tr><td>85</td><td>13</td><td>93</td><td>51</td><td>53</td></tr></table> <p>Layer 2:</p> <table border="1"><tr><td>107</td><td>60</td><td>119</td><td>80</td><td>52</td></tr><tr><td>102</td><td>74</td><td>40</td><td>110</td><td>68</td></tr><tr><td>23</td><td>16</td><td>61</td><td>76</td><td>6</td></tr><tr><td>92</td><td>103</td><td>82</td><td>69</td><td>63</td></tr></table> <p>Layer 3:</p> <table border="1"><tr><td>48</td><td>70</td><td>79</td><td>20</td><td>15</td></tr><tr><td>116</td><td>41</td><td>19</td><td>50</td><td>8</td></tr><tr><td>33</td><td>22</td><td>105</td><td>49</td><td>9</td></tr><tr><td>97</td><td>75</td><td>77</td><td>7</td><td>32</td></tr><tr><td>86</td><td>94</td><td>58</td><td>26</td><td>121</td></tr></table> <p>Layer 4:</p> <table border="1"><tr><td>124</td><td>17</td><td>118</td><td>84</td><td>1</td></tr><tr><td>56</td><td>88</td><td>73</td><td>46</td><td>37</td></tr><tr><td>12</td><td>45</td><td>27</td><td>14</td><td>117</td></tr><tr><td>44</td><td>21</td><td>120</td><td>81</td><td>59</td></tr></table> <p>Layer 5:</p> <table border="1"><tr><td>39</td><td>111</td><td>43</td><td>54</td><td>35</td></tr><tr><td>87</td><td>2</td><td>98</td><td>71</td><td>112</td></tr><tr><td>29</td><td>109</td><td>99</td><td>24</td><td>106</td></tr><tr><td>125</td><td>10</td><td>18</td><td>114</td><td>90</td></tr><tr><td>31</td><td>104</td><td>28</td><td>83</td><td>55</td></tr></table>	4	100	113	36	34	65	64	38	42	30	57	72	122	62	115	85	13	93	51	53	107	60	119	80	52	102	74	40	110	68	23	16	61	76	6	92	103	82	69	63	48	70	79	20	15	116	41	19	50	8	33	22	105	49	9	97	75	77	7	32	86	94	58	26	121	124	17	118	84	1	56	88	73	46	37	12	45	27	14	117	44	21	120	81	59	39	111	43	54	35	87	2	98	71	112	29	109	99	24	106	125	10	18	114	90	31	104	28	83	55
4	100	113	36	34																																																																																																											
65	64	38	42	30																																																																																																											
57	72	122	62	115																																																																																																											
85	13	93	51	53																																																																																																											
107	60	119	80	52																																																																																																											
102	74	40	110	68																																																																																																											
23	16	61	76	6																																																																																																											
92	103	82	69	63																																																																																																											
48	70	79	20	15																																																																																																											
116	41	19	50	8																																																																																																											
33	22	105	49	9																																																																																																											
97	75	77	7	32																																																																																																											
86	94	58	26	121																																																																																																											
124	17	118	84	1																																																																																																											
56	88	73	46	37																																																																																																											
12	45	27	14	117																																																																																																											
44	21	120	81	59																																																																																																											
39	111	43	54	35																																																																																																											
87	2	98	71	112																																																																																																											
29	109	99	24	106																																																																																																											
125	10	18	114	90																																																																																																											
31	104	28	83	55																																																																																																											
Objective Akhir	5458																																																																																																														
Plot nilai objective function terhadap banyak iterasi yang telah dilewati	<p>Objective Function vs Iterations (Genetic Algorithm)</p> <p>Max Objective Function</p> <p>Average Objective Function</p> <p>Iterations</p> <p>Objective Function Value</p>																																																																																																														

Hasil Eksperimen	<pre> ALGORITHM RESULTS  Algorithm: Genetic Algorithm Initial Objective Value: 6632 Final Objective Value: 5458 Iterations: 100 Duration: 7.5638 seconds Stuck Counter: [] Population Size: 10 </pre>
Jumlah Populasi	10
Banyak Iterasi	100
Durasi Pencarian	7.5683 detik

**Tabel 4.3.6.** Pengujian 2 Populasi sebagai Kontrol

State Awal	<pre> Initial Cube State:  Layer 1:       30   50   106   41   122       83   47   76   124   31       37   99   34   43   72       22   111   52   101   38       69   13   54   53   44  Layer 2:       14   57   25   40   79       66   107   1   39   78       24   112   63   117   75       121   77   46   118   10       87   51   81   119   89  Layer 3:       71   11   29   59   114   4       74   61   26   91   78       115   58   21   64       113   116   90       32   42   55   118   82  Layer 4:       45   80   100   5   97       68   104   67   84   68       35   2   108   8       7   9   95   56   88       103   109   92   102   94  Layer 5:       19   65   20   49   3       36   93   73   62   123       6   98   23   28   125       12   105   120   15   85       16   86   17   27   33 </pre>
Objective Awal	7318

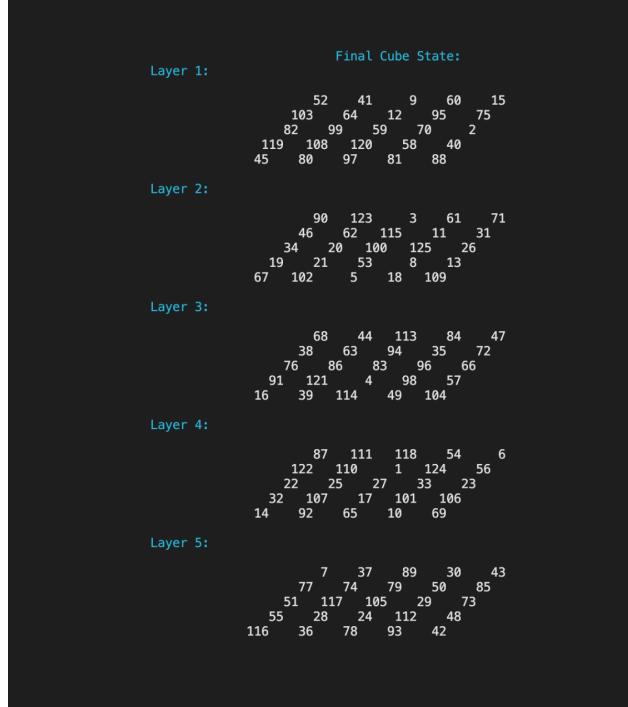
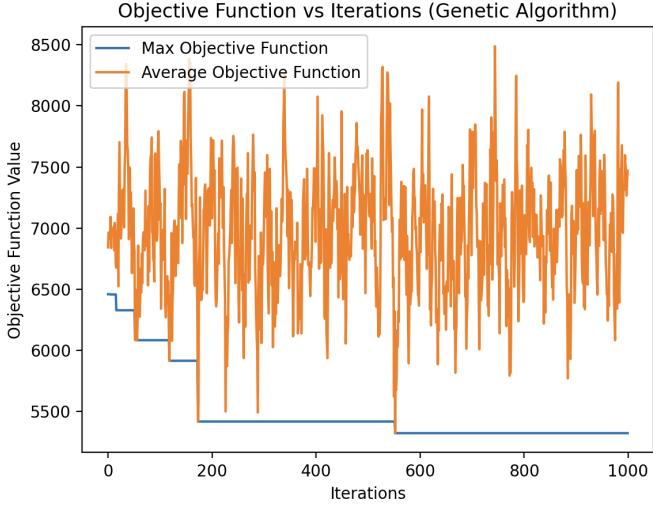
State Akhir	<p>Final Cube State:</p> <p>Layer 1:</p> <table border="1"><tr><td>10</td><td>32</td><td>67</td><td>7</td><td>36</td></tr><tr><td>22</td><td>84</td><td>110</td><td>48</td><td>58</td></tr><tr><td>56</td><td>90</td><td>20</td><td>103</td><td>45</td></tr><tr><td>41</td><td>75</td><td>122</td><td>116</td><td>125</td></tr><tr><td>51</td><td>85</td><td>42</td><td>109</td><td>125</td></tr></table> <p>Layer 2:</p> <table border="1"><tr><td>113</td><td>92</td><td>17</td><td>46</td><td>97</td></tr><tr><td>30</td><td>28</td><td>13</td><td>81</td><td>26</td></tr><tr><td>100</td><td>3</td><td>29</td><td>88</td><td>94</td></tr><tr><td>87</td><td>35</td><td>14</td><td>68</td><td>49</td></tr><tr><td>47</td><td>70</td><td>93</td><td>52</td><td>99</td></tr></table> <p>Layer 3:</p> <table border="1"><tr><td>57</td><td>16</td><td>40</td><td>112</td><td>89</td></tr><tr><td>95</td><td>66</td><td>96</td><td>39</td><td>89</td></tr><tr><td>98</td><td>37</td><td>73</td><td>15</td><td>6</td></tr><tr><td>83</td><td>53</td><td>106</td><td>86</td><td>21</td></tr><tr><td>34</td><td>124</td><td>5</td><td>114</td><td>2</td></tr></table> <p>Layer 4:</p> <table border="1"><tr><td>82</td><td>12</td><td>61</td><td>78</td><td>69</td><td>4</td></tr><tr><td>64</td><td>18</td><td>60</td><td>43</td><td>27</td><td>54</td></tr><tr><td>74</td><td>80</td><td>102</td><td>59</td><td>104</td><td>19</td></tr><tr><td>79</td><td>9</td><td>119</td><td>123</td><td>117</td><td>1</td></tr></table> <p>Layer 5:</p> <table border="1"><tr><td>63</td><td>65</td><td>111</td><td>105</td><td>107</td></tr><tr><td>118</td><td>33</td><td>77</td><td>55</td><td>23</td></tr><tr><td>62</td><td>24</td><td>115</td><td>108</td><td>8</td></tr><tr><td>11</td><td>72</td><td>38</td><td>25</td><td>120</td></tr><tr><td>1</td><td>121</td><td>44</td><td>76</td><td>101</td></tr></table>	10	32	67	7	36	22	84	110	48	58	56	90	20	103	45	41	75	122	116	125	51	85	42	109	125	113	92	17	46	97	30	28	13	81	26	100	3	29	88	94	87	35	14	68	49	47	70	93	52	99	57	16	40	112	89	95	66	96	39	89	98	37	73	15	6	83	53	106	86	21	34	124	5	114	2	82	12	61	78	69	4	64	18	60	43	27	54	74	80	102	59	104	19	79	9	119	123	117	1	63	65	111	105	107	118	33	77	55	23	62	24	115	108	8	11	72	38	25	120	1	121	44	76	101
10	32	67	7	36																																																																																																																									
22	84	110	48	58																																																																																																																									
56	90	20	103	45																																																																																																																									
41	75	122	116	125																																																																																																																									
51	85	42	109	125																																																																																																																									
113	92	17	46	97																																																																																																																									
30	28	13	81	26																																																																																																																									
100	3	29	88	94																																																																																																																									
87	35	14	68	49																																																																																																																									
47	70	93	52	99																																																																																																																									
57	16	40	112	89																																																																																																																									
95	66	96	39	89																																																																																																																									
98	37	73	15	6																																																																																																																									
83	53	106	86	21																																																																																																																									
34	124	5	114	2																																																																																																																									
82	12	61	78	69	4																																																																																																																								
64	18	60	43	27	54																																																																																																																								
74	80	102	59	104	19																																																																																																																								
79	9	119	123	117	1																																																																																																																								
63	65	111	105	107																																																																																																																									
118	33	77	55	23																																																																																																																									
62	24	115	108	8																																																																																																																									
11	72	38	25	120																																																																																																																									
1	121	44	76	101																																																																																																																									
Objective Akhir	5675																																																																																																																												
Plot nilai objective function terhadap banyak iterasi yang telah dilewati	<p>Objective Function vs Iterations (Genetic Algorithm)</p> <p>Max Objective Function</p> <p>Average Objective Function</p> <p>Objective Function Value</p> <p>Iterations</p>																																																																																																																												

Hasil Eksperimen	<pre> ALGORITHM RESULTS  Algorithm: Genetic Algorithm Initial Objective Value: 7318 Final Objective Value: 5675 Iterations: 100 Duration: 7.4402 seconds Stuck Counter: [] Population Size: 10 </pre>
Jumlah Populasi	10
Banyak Iterasi	100
Durasi Pencarian	7.4402 detik

### 3. Pengujian 3 - Populasi sebagai Kontrol dengan 1000 Iterasi

**Tabel 4.3.7.** Pengujian 3 Populasi sebagai Kontrol

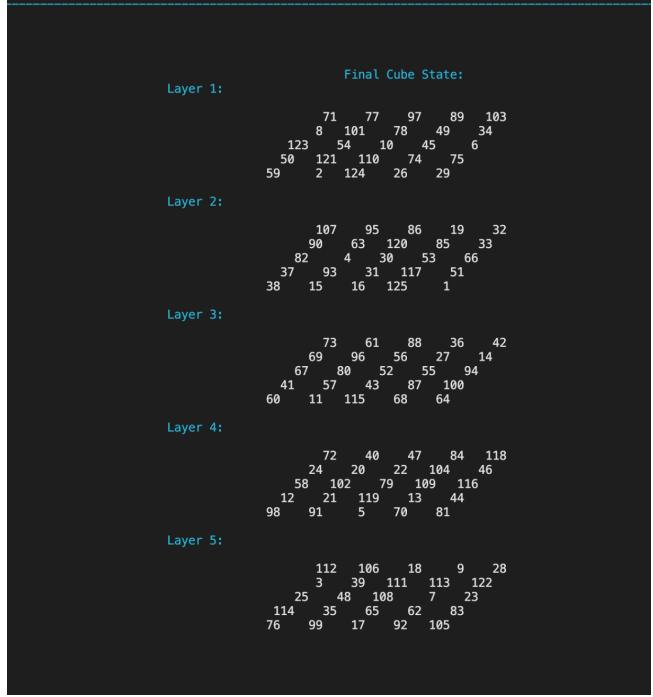
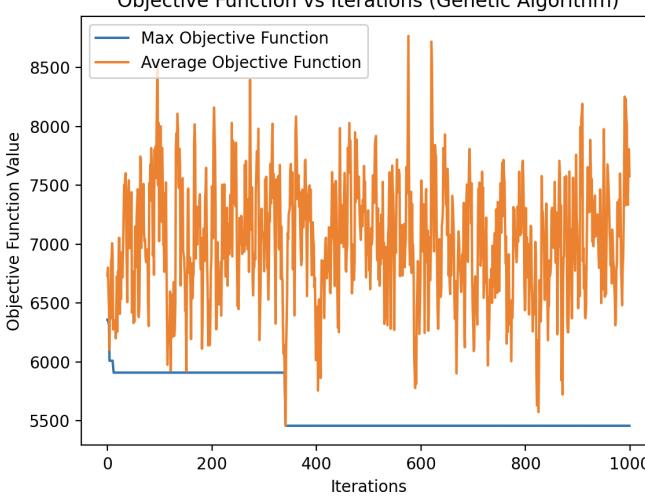
State Awal	<pre> Initial Cube State:  Layer 1:       120 117  7  85  66       95 109 110 16 32       13   75  72  80   6       99  27  90 101 116      107  39  26  36  Layer 2:       78  19  29 114   3       59  43  57 21  56       88  62  28 113 102       84  71  77 106  64       76   9  87  89  86  Layer 3:       50  44 123 112  92       49  70 105  63  24       8  122 118  53  10       2  35  58  60  47      12 119  61  91 115  Layer 4:       83  46 103  73  40       96 124 125 108  17       100 30 111  98  15       37  82  1  41  18       20  31  45  42  22  Layer 5:       14  25  51  54  67  5       38 121  23  81  67       33   4  97  52  94       48  68  65  93  79       55 104  34  74  69 </pre>
Objective Awal	7193

State Akhir	 <p>Final Cube State:</p> <p>Layer 1:</p> <table border="1"><tr><td>52</td><td>41</td><td>9</td><td>60</td><td>15</td></tr><tr><td>103</td><td>64</td><td>12</td><td>95</td><td></td></tr><tr><td>82</td><td>99</td><td>59</td><td>70</td><td>2</td></tr><tr><td>119</td><td>108</td><td>120</td><td>58</td><td>40</td></tr><tr><td>45</td><td>80</td><td>97</td><td>81</td><td>88</td></tr></table> <p>Layer 2:</p> <table border="1"><tr><td>90</td><td>123</td><td>3</td><td>61</td><td>71</td></tr><tr><td>46</td><td>62</td><td>115</td><td>11</td><td>31</td></tr><tr><td>34</td><td>20</td><td>100</td><td>125</td><td>26</td></tr><tr><td>19</td><td>21</td><td>53</td><td>8</td><td>13</td></tr><tr><td>67</td><td>102</td><td>5</td><td>18</td><td>109</td></tr></table> <p>Layer 3:</p> <table border="1"><tr><td>68</td><td>44</td><td>113</td><td>84</td><td>47</td></tr><tr><td>38</td><td>63</td><td>94</td><td>35</td><td>72</td></tr><tr><td>76</td><td>86</td><td>83</td><td>96</td><td>66</td></tr><tr><td>91</td><td>121</td><td>4</td><td>98</td><td>57</td></tr><tr><td>16</td><td>39</td><td>114</td><td>49</td><td>104</td></tr></table> <p>Layer 4:</p> <table border="1"><tr><td>87</td><td>111</td><td>118</td><td>54</td><td>6</td></tr><tr><td>122</td><td>110</td><td>1</td><td>124</td><td>23</td></tr><tr><td>22</td><td>25</td><td>27</td><td>33</td><td>56</td></tr><tr><td>32</td><td>107</td><td>17</td><td>101</td><td>106</td></tr><tr><td>14</td><td>92</td><td>65</td><td>10</td><td>69</td></tr></table> <p>Layer 5:</p> <table border="1"><tr><td>7</td><td>37</td><td>89</td><td>30</td><td>43</td></tr><tr><td>77</td><td>74</td><td>79</td><td>50</td><td>85</td></tr><tr><td>51</td><td>117</td><td>185</td><td>29</td><td>73</td></tr><tr><td>55</td><td>28</td><td>24</td><td>112</td><td>48</td></tr><tr><td>116</td><td>36</td><td>78</td><td>93</td><td>42</td></tr></table>	52	41	9	60	15	103	64	12	95		82	99	59	70	2	119	108	120	58	40	45	80	97	81	88	90	123	3	61	71	46	62	115	11	31	34	20	100	125	26	19	21	53	8	13	67	102	5	18	109	68	44	113	84	47	38	63	94	35	72	76	86	83	96	66	91	121	4	98	57	16	39	114	49	104	87	111	118	54	6	122	110	1	124	23	22	25	27	33	56	32	107	17	101	106	14	92	65	10	69	7	37	89	30	43	77	74	79	50	85	51	117	185	29	73	55	28	24	112	48	116	36	78	93	42
52	41	9	60	15																																																																																																																										
103	64	12	95																																																																																																																											
82	99	59	70	2																																																																																																																										
119	108	120	58	40																																																																																																																										
45	80	97	81	88																																																																																																																										
90	123	3	61	71																																																																																																																										
46	62	115	11	31																																																																																																																										
34	20	100	125	26																																																																																																																										
19	21	53	8	13																																																																																																																										
67	102	5	18	109																																																																																																																										
68	44	113	84	47																																																																																																																										
38	63	94	35	72																																																																																																																										
76	86	83	96	66																																																																																																																										
91	121	4	98	57																																																																																																																										
16	39	114	49	104																																																																																																																										
87	111	118	54	6																																																																																																																										
122	110	1	124	23																																																																																																																										
22	25	27	33	56																																																																																																																										
32	107	17	101	106																																																																																																																										
14	92	65	10	69																																																																																																																										
7	37	89	30	43																																																																																																																										
77	74	79	50	85																																																																																																																										
51	117	185	29	73																																																																																																																										
55	28	24	112	48																																																																																																																										
116	36	78	93	42																																																																																																																										
Objective Akhir	5324																																																																																																																													
Plot nilai objective function terhadap banyak iterasi yang telah dilewati	 <p>Objective Function vs Iterations (Genetic Algorithm)</p> <p>Y-axis: Objective Function Value (5500 to 8500)</p> <p>X-axis: Iterations (0 to 1000)</p> <p>Legend: Max Objective Function (Blue line), Average Objective Function (Orange line)</p> <p>The plot shows the objective function values over 1000 iterations. The blue line represents the maximum objective function, which starts around 6500 and quickly drops to a plateau around 5500. The orange line represents the average objective function, which fluctuates significantly between 5500 and 8500, indicating high variance in the search space.</p>																																																																																																																													

Hasil Eksperimen	<pre> ALGORITHM RESULTS  Algorithm: Genetic Algorithm Initial Objective Value: 7193 Final Objective Value: 5324 Iterations: 1000 Duration: 73.9513 seconds Stuck Counter: [] Population Size: 10 </pre>
Jumlah Populasi	10
Banyak Iterasi	1000
Durasi Pencarian	73.9513 detik

**Tabel 4.3.8.** Pengujian 3 Populasi sebagai Kontrol

State Awal	<pre> Initial Cube State:  Layer 1:       43   57   20   16   79       23   90   83   105  19       69   28   100  59   109       44   38   108  110  27      102   73   74   123  114  Layer 2:       88   40   22   71   41       9    33   15   47   42       80   125  119  72   53       63   66   62   64   82       46   37   118  92   35  Layer 3:       21   25   117  104  124       17   120  52   107  121       14   13   89   31   55       84   87   70   26   101       61   48   50   6    3  Layer 4:       76   96   112  106  94       99   113  11   77   115       98   93   12   68   45       39   8    49   56   56       67   111  78   5    2  Layer 5:       1    60   30   86   65       95   51   24   85   2       32   58   10   116  91       36   122  75   34   103       29   7    81   18   54 </pre>
Objective Awal	7012

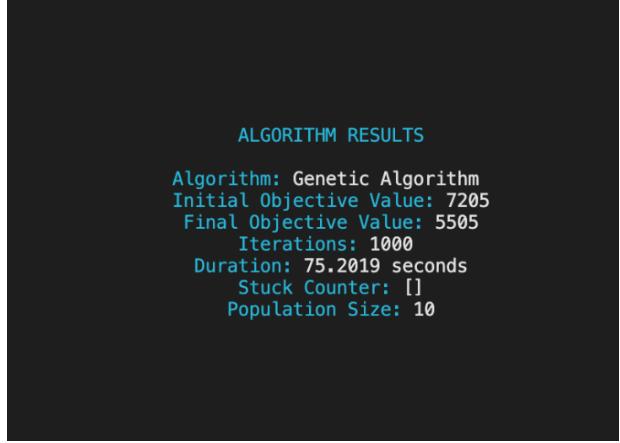
State Akhir	
Objective Akhir	5459
Plot nilai objective function terhadap banyak iterasi yang telah dilewati	

Hasil Eksperimen	<pre> ALGORITHM RESULTS  Algorithm: Genetic Algorithm Initial Objective Value: 7012 Final Objective Value: 5459 Iterations: 1000 Duration: 76.1343 seconds Stuck Counter: [] Population Size: 10 </pre>
Jumlah Populasi	10
Banyak Iterasi	1000
Durasi Pencarian	76.1343 detik

**Tabel 4.3.9.** Pengujian 3 Populasi sebagai Kontrol

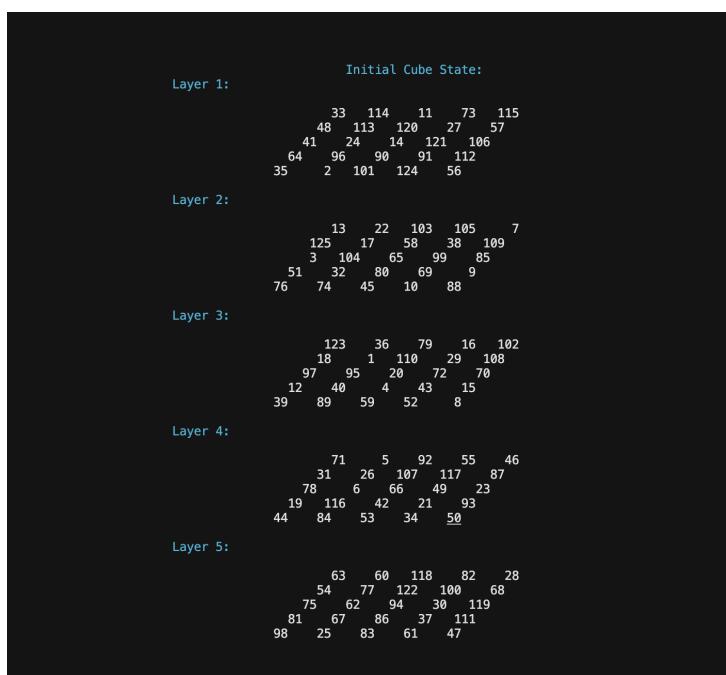
State Awal	<pre> Initial Cube State:  Layer 1:       76   113   50   90   82       20    74   48   14   89       18    33   106   100   51       40    66   64   39   114      102    29   105   109   96  Layer 2:       37   75   112   38   27       95   36   69   41   71       77   86   17   34   45      119   99   103   108   30       53   80   101   122   88  Layer 3:       21   13   73   110   98       78   55   28   57   25       115   16   121   58   12       60    3   111    7   65        8   10   123   120   81  Layer 4:       79   52    5   35   62       49   70   67   84   61       104    2    1   107   63       94   44   92   42   11       43    4   47   23   68  Layer 5:       83   15   72   22   19       32    9   26   118   56       125   93   97   59   117       24   54   46   85   87      116 </pre>
------------	---

Objective Awal	7205																																																																																																																																							
State Akhir	<p style="text-align: center;">Final Cube State:</p> <p>Layer 1:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>77</td><td>88</td><td>93</td><td>100</td><td>48</td></tr> <tr><td>43</td><td>45</td><td>42</td><td>37</td><td>87</td></tr> <tr><td>41</td><td>47</td><td>14</td><td>54</td><td>59</td></tr> <tr><td>63</td><td>120</td><td>28</td><td>20</td><td>21</td></tr> <tr><td>51</td><td></td><td>91</td><td></td><td>78</td></tr> </table> <p>Layer 2:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>68</td><td>117</td><td>39</td><td>7</td><td>2</td></tr> <tr><td>125</td><td>94</td><td>90</td><td>108</td><td>110</td></tr> <tr><td>1</td><td>73</td><td>111</td><td>62</td><td>114</td></tr> <tr><td>53</td><td>66</td><td>103</td><td>30</td><td>8</td></tr> <tr><td></td><td></td><td></td><td></td><td></td></tr> </table> <p>Layer 3:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>101</td><td>16</td><td>4</td><td>67</td><td>84</td></tr> <tr><td>115</td><td>40</td><td>122</td><td>102</td><td>72</td></tr> <tr><td>15</td><td>82</td><td>38</td><td>17</td><td>79</td></tr> <tr><td>32</td><td>69</td><td>5</td><td>71</td><td>107</td></tr> <tr><td></td><td></td><td></td><td></td><td></td></tr> </table> <p>Layer 4:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>99</td><td>104</td><td>65</td><td>44</td><td>124</td></tr> <tr><td>35</td><td>61</td><td>34</td><td>70</td><td>52</td></tr> <tr><td>18</td><td>116</td><td>74</td><td>121</td><td>3</td></tr> <tr><td>22</td><td>19</td><td>112</td><td>31</td><td>33</td></tr> <tr><td>36</td><td>123</td><td>56</td><td>96</td><td>24</td></tr> <tr><td></td><td></td><td></td><td></td><td></td></tr> </table> <p>Layer 5:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>29</td><td>26</td><td>58</td><td>86</td><td>81</td></tr> <tr><td>11</td><td>92</td><td>113</td><td>13</td><td>105</td></tr> <tr><td>50</td><td>118</td><td>57</td><td>97</td><td>6</td></tr> <tr><td>98</td><td>9</td><td>80</td><td>49</td><td>27</td></tr> <tr><td>83</td><td>23</td><td>119</td><td>60</td><td>12</td></tr> <tr><td></td><td></td><td></td><td></td><td></td></tr> </table>	77	88	93	100	48	43	45	42	37	87	41	47	14	54	59	63	120	28	20	21	51		91		78	68	117	39	7	2	125	94	90	108	110	1	73	111	62	114	53	66	103	30	8						101	16	4	67	84	115	40	122	102	72	15	82	38	17	79	32	69	5	71	107						99	104	65	44	124	35	61	34	70	52	18	116	74	121	3	22	19	112	31	33	36	123	56	96	24						29	26	58	86	81	11	92	113	13	105	50	118	57	97	6	98	9	80	49	27	83	23	119	60	12					
77	88	93	100	48																																																																																																																																				
43	45	42	37	87																																																																																																																																				
41	47	14	54	59																																																																																																																																				
63	120	28	20	21																																																																																																																																				
51		91		78																																																																																																																																				
68	117	39	7	2																																																																																																																																				
125	94	90	108	110																																																																																																																																				
1	73	111	62	114																																																																																																																																				
53	66	103	30	8																																																																																																																																				
101	16	4	67	84																																																																																																																																				
115	40	122	102	72																																																																																																																																				
15	82	38	17	79																																																																																																																																				
32	69	5	71	107																																																																																																																																				
99	104	65	44	124																																																																																																																																				
35	61	34	70	52																																																																																																																																				
18	116	74	121	3																																																																																																																																				
22	19	112	31	33																																																																																																																																				
36	123	56	96	24																																																																																																																																				
29	26	58	86	81																																																																																																																																				
11	92	113	13	105																																																																																																																																				
50	118	57	97	6																																																																																																																																				
98	9	80	49	27																																																																																																																																				
83	23	119	60	12																																																																																																																																				
Objective Akhir	5505																																																																																																																																							
Plot nilai objective function terhadap banyak iterasi yang telah dilewati	<p style="text-align: center;">Objective Function vs Iterations (Genetic Algorithm)</p> <p>Objective Function Value</p> <p>Iterations</p> <p>Max Objective Function</p> <p>Average Objective Function</p>																																																																																																																																							

Hasil Eksperimen	
Jumlah Populasi	10
Banyak Iterasi	1000
Durasi Pencarian	75.2019 detik

#### 4. Pengujian 4 - Iterasi sebagai Kontrol dengan 10 Populasi

**Tabel 4.3.10.** Pengujian 4 Iterasi sebagai Kontrol

State Awal	
------------	--

Objective Awal	6615																																																																																																																													
State Akhir	<p style="text-align: center;">Final Cube State:</p> <p>Layer 1:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>99</td><td>88</td><td>32</td><td>65</td><td>20</td></tr> <tr><td>38</td><td>92</td><td>39</td><td>62</td><td>78</td></tr> <tr><td>9</td><td>37</td><td>13</td><td>15</td><td>114</td></tr> <tr><td>52</td><td>45</td><td>60</td><td>85</td><td>26</td></tr> <tr><td>115</td><td>83</td><td>5</td><td>102</td><td>24</td></tr> </table> <p>Layer 2:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>112</td><td>54</td><td>25</td><td>58</td><td>74</td></tr> <tr><td>55</td><td>98</td><td>11</td><td>91</td><td>109</td></tr> <tr><td>51</td><td>27</td><td>66</td><td>68</td><td>1</td></tr> <tr><td>44</td><td>19</td><td>30</td><td>7</td><td>22</td></tr> <tr><td>75</td><td>124</td><td>36</td><td>16</td><td>107</td></tr> </table> <p>Layer 3:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>72</td><td>121</td><td>118</td><td>10</td><td>80</td></tr> <tr><td>34</td><td>97</td><td>67</td><td>111</td><td>8</td></tr> <tr><td>95</td><td>42</td><td>183</td><td>48</td><td>122</td></tr> <tr><td>101</td><td>12</td><td>108</td><td>77</td><td>33</td></tr> <tr><td>6</td><td>41</td><td>116</td><td>89</td><td>82</td></tr> </table> <p>Layer 4:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>69</td><td>81</td><td>71</td><td>70</td><td>53</td></tr> <tr><td>59</td><td>50</td><td>76</td><td>125</td><td>113</td></tr> <tr><td>93</td><td>35</td><td>94</td><td>29</td><td>47</td></tr> <tr><td>28</td><td>105</td><td>63</td><td>18</td><td>56</td></tr> <tr><td>46</td><td>120</td><td>31</td><td>90</td><td>106</td></tr> </table> <p>Layer 5:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>3</td><td>73</td><td>57</td><td>61</td><td>21</td></tr> <tr><td>87</td><td>96</td><td>123</td><td>119</td><td>17</td></tr> <tr><td>23</td><td>64</td><td>79</td><td>2</td><td>110</td></tr> <tr><td>4</td><td>49</td><td>104</td><td>117</td><td>43</td></tr> <tr><td>100</td><td>40</td><td>86</td><td>84</td><td>14</td></tr> </table>	99	88	32	65	20	38	92	39	62	78	9	37	13	15	114	52	45	60	85	26	115	83	5	102	24	112	54	25	58	74	55	98	11	91	109	51	27	66	68	1	44	19	30	7	22	75	124	36	16	107	72	121	118	10	80	34	97	67	111	8	95	42	183	48	122	101	12	108	77	33	6	41	116	89	82	69	81	71	70	53	59	50	76	125	113	93	35	94	29	47	28	105	63	18	56	46	120	31	90	106	3	73	57	61	21	87	96	123	119	17	23	64	79	2	110	4	49	104	117	43	100	40	86	84	14
99	88	32	65	20																																																																																																																										
38	92	39	62	78																																																																																																																										
9	37	13	15	114																																																																																																																										
52	45	60	85	26																																																																																																																										
115	83	5	102	24																																																																																																																										
112	54	25	58	74																																																																																																																										
55	98	11	91	109																																																																																																																										
51	27	66	68	1																																																																																																																										
44	19	30	7	22																																																																																																																										
75	124	36	16	107																																																																																																																										
72	121	118	10	80																																																																																																																										
34	97	67	111	8																																																																																																																										
95	42	183	48	122																																																																																																																										
101	12	108	77	33																																																																																																																										
6	41	116	89	82																																																																																																																										
69	81	71	70	53																																																																																																																										
59	50	76	125	113																																																																																																																										
93	35	94	29	47																																																																																																																										
28	105	63	18	56																																																																																																																										
46	120	31	90	106																																																																																																																										
3	73	57	61	21																																																																																																																										
87	96	123	119	17																																																																																																																										
23	64	79	2	110																																																																																																																										
4	49	104	117	43																																																																																																																										
100	40	86	84	14																																																																																																																										
Objective Akhir	5932																																																																																																																													
Plot nilai objective function terhadap banyak iterasi yang telah dilewati	<p style="text-align: center;">Objective Function vs Iterations (Genetic Algorithm)</p> <table border="1"> <caption>Data for Objective Function vs Iterations</caption> <thead> <tr> <th>Iterations</th> <th>Max Objective Function</th> <th>Average Objective Function</th> </tr> </thead> <tbody> <tr><td>0</td><td>6000</td><td>6750</td></tr> <tr><td>1</td><td>6000</td><td>7000</td></tr> <tr><td>2</td><td>6000</td><td>7000</td></tr> <tr><td>3</td><td>6000</td><td>6900</td></tr> <tr><td>4</td><td>6000</td><td>7100</td></tr> <tr><td>5</td><td>6000</td><td>7500</td></tr> <tr><td>6</td><td>6000</td><td>7500</td></tr> <tr><td>7</td><td>6000</td><td>7450</td></tr> <tr><td>8</td><td>6000</td><td>7000</td></tr> </tbody> </table>	Iterations	Max Objective Function	Average Objective Function	0	6000	6750	1	6000	7000	2	6000	7000	3	6000	6900	4	6000	7100	5	6000	7500	6	6000	7500	7	6000	7450	8	6000	7000																																																																																															
Iterations	Max Objective Function	Average Objective Function																																																																																																																												
0	6000	6750																																																																																																																												
1	6000	7000																																																																																																																												
2	6000	7000																																																																																																																												
3	6000	6900																																																																																																																												
4	6000	7100																																																																																																																												
5	6000	7500																																																																																																																												
6	6000	7500																																																																																																																												
7	6000	7450																																																																																																																												
8	6000	7000																																																																																																																												
Objective Awal	6615																																																																																																																													
State Akhir	<p style="text-align: center;">Final Cube State:</p> <p>Layer 1:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>99</td><td>88</td><td>32</td><td>65</td><td>20</td></tr> <tr><td>38</td><td>92</td><td>39</td><td>62</td><td>78</td></tr> <tr><td>9</td><td>37</td><td>13</td><td>15</td><td>114</td></tr> <tr><td>52</td><td>45</td><td>60</td><td>85</td><td>26</td></tr> <tr><td>115</td><td>83</td><td>5</td><td>102</td><td>24</td></tr> </table> <p>Layer 2:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>112</td><td>54</td><td>25</td><td>58</td><td>74</td></tr> <tr><td>55</td><td>98</td><td>11</td><td>91</td><td>109</td></tr> <tr><td>51</td><td>27</td><td>66</td><td>68</td><td>1</td></tr> <tr><td>44</td><td>19</td><td>30</td><td>7</td><td>22</td></tr> <tr><td>75</td><td>124</td><td>36</td><td>16</td><td>107</td></tr> </table> <p>Layer 3:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>72</td><td>121</td><td>118</td><td>10</td><td>80</td></tr> <tr><td>34</td><td>97</td><td>67</td><td>111</td><td>8</td></tr> <tr><td>95</td><td>42</td><td>183</td><td>48</td><td>122</td></tr> <tr><td>101</td><td>12</td><td>108</td><td>77</td><td>33</td></tr> <tr><td>6</td><td>41</td><td>116</td><td>89</td><td>82</td></tr> </table> <p>Layer 4:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>69</td><td>81</td><td>71</td><td>70</td><td>53</td></tr> <tr><td>59</td><td>50</td><td>76</td><td>125</td><td>113</td></tr> <tr><td>93</td><td>35</td><td>94</td><td>29</td><td>47</td></tr> <tr><td>28</td><td>105</td><td>63</td><td>18</td><td>56</td></tr> <tr><td>46</td><td>120</td><td>31</td><td>90</td><td>106</td></tr> </table> <p>Layer 5:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>3</td><td>73</td><td>57</td><td>61</td><td>21</td></tr> <tr><td>87</td><td>96</td><td>123</td><td>119</td><td>17</td></tr> <tr><td>23</td><td>64</td><td>79</td><td>2</td><td>110</td></tr> <tr><td>4</td><td>49</td><td>104</td><td>117</td><td>43</td></tr> <tr><td>100</td><td>40</td><td>86</td><td>84</td><td>14</td></tr> </table>	99	88	32	65	20	38	92	39	62	78	9	37	13	15	114	52	45	60	85	26	115	83	5	102	24	112	54	25	58	74	55	98	11	91	109	51	27	66	68	1	44	19	30	7	22	75	124	36	16	107	72	121	118	10	80	34	97	67	111	8	95	42	183	48	122	101	12	108	77	33	6	41	116	89	82	69	81	71	70	53	59	50	76	125	113	93	35	94	29	47	28	105	63	18	56	46	120	31	90	106	3	73	57	61	21	87	96	123	119	17	23	64	79	2	110	4	49	104	117	43	100	40	86	84	14
99	88	32	65	20																																																																																																																										
38	92	39	62	78																																																																																																																										
9	37	13	15	114																																																																																																																										
52	45	60	85	26																																																																																																																										
115	83	5	102	24																																																																																																																										
112	54	25	58	74																																																																																																																										
55	98	11	91	109																																																																																																																										
51	27	66	68	1																																																																																																																										
44	19	30	7	22																																																																																																																										
75	124	36	16	107																																																																																																																										
72	121	118	10	80																																																																																																																										
34	97	67	111	8																																																																																																																										
95	42	183	48	122																																																																																																																										
101	12	108	77	33																																																																																																																										
6	41	116	89	82																																																																																																																										
69	81	71	70	53																																																																																																																										
59	50	76	125	113																																																																																																																										
93	35	94	29	47																																																																																																																										
28	105	63	18	56																																																																																																																										
46	120	31	90	106																																																																																																																										
3	73	57	61	21																																																																																																																										
87	96	123	119	17																																																																																																																										
23	64	79	2	110																																																																																																																										
4	49	104	117	43																																																																																																																										
100	40	86	84	14																																																																																																																										
Objective Akhir	5932																																																																																																																													
Plot nilai objective function terhadap banyak iterasi yang telah dilewati	<p style="text-align: center;">Objective Function vs Iterations (Genetic Algorithm)</p> <table border="1"> <caption>Data for Objective Function vs Iterations</caption> <thead> <tr> <th>Iterations</th> <th>Max Objective Function</th> <th>Average Objective Function</th> </tr> </thead> <tbody> <tr><td>0</td><td>6000</td><td>6750</td></tr> <tr><td>1</td><td>6000</td><td>7000</td></tr> <tr><td>2</td><td>6000</td><td>7000</td></tr> <tr><td>3</td><td>6000</td><td>6900</td></tr> <tr><td>4</td><td>6000</td><td>7100</td></tr> <tr><td>5</td><td>6000</td><td>7500</td></tr> <tr><td>6</td><td>6000</td><td>7500</td></tr> <tr><td>7</td><td>6000</td><td>7450</td></tr> <tr><td>8</td><td>6000</td><td>7000</td></tr> </tbody> </table>	Iterations	Max Objective Function	Average Objective Function	0	6000	6750	1	6000	7000	2	6000	7000	3	6000	6900	4	6000	7100	5	6000	7500	6	6000	7500	7	6000	7450	8	6000	7000																																																																																															
Iterations	Max Objective Function	Average Objective Function																																																																																																																												
0	6000	6750																																																																																																																												
1	6000	7000																																																																																																																												
2	6000	7000																																																																																																																												
3	6000	6900																																																																																																																												
4	6000	7100																																																																																																																												
5	6000	7500																																																																																																																												
6	6000	7500																																																																																																																												
7	6000	7450																																																																																																																												
8	6000	7000																																																																																																																												

Hasil Eksperimen	<b>ALGORITHM RESULTS</b> Algorithm: Genetic Algorithm Initial Objective Value: 6615 Final Objective Value: 5932 Iterations: 10 Duration: 0.7908 seconds Stuck Counter: [] Population Size: 10
Jumlah Populasi	10
Banyak Iterasi	10
Durasi Pencarian	0.7908 detik

**Tabel 4.3.11.** Pengujian 4 Iterasi sebagai Kontrol

State Awal	Initial Cube State: Layer 1: <table> <tr><td>52</td><td>21</td><td>85</td><td>64</td><td>29</td></tr> <tr><td>20</td><td>105</td><td>1</td><td>57</td><td>51</td></tr> <tr><td>56</td><td>9</td><td>49</td><td>93</td><td>114</td></tr> <tr><td>40</td><td>63</td><td>74</td><td>100</td><td>116</td></tr> <tr><td>125</td><td>71</td><td>92</td><td>82</td><td>33</td></tr> </table> Layer 2: <table> <tr><td>43</td><td>103</td><td>59</td><td>101</td><td>108</td></tr> <tr><td>2</td><td>58</td><td>88</td><td>87</td><td>3</td></tr> <tr><td>86</td><td>53</td><td>36</td><td>115</td><td>119</td></tr> <tr><td>24</td><td>37</td><td>68</td><td>77</td><td>107</td></tr> <tr><td>91</td><td>48</td><td>97</td><td>65</td><td>62</td></tr> </table> Layer 3: <table> <tr><td>16</td><td>90</td><td>66</td><td>18</td><td>60</td></tr> <tr><td>39</td><td>5</td><td>113</td><td>73</td><td>80</td></tr> <tr><td>84</td><td>8</td><td>31</td><td>121</td><td>112</td></tr> <tr><td>89</td><td>61</td><td>47</td><td>25</td><td>104</td></tr> <tr><td>14</td><td>70</td><td>30</td><td>117</td><td>122</td></tr> </table> Layer 4: <table> <tr><td>28</td><td>102</td><td>38</td><td>42</td><td>78</td></tr> <tr><td>72</td><td>32</td><td>109</td><td>41</td><td>46</td></tr> <tr><td>118</td><td>110</td><td>6</td><td>96</td><td>54</td></tr> <tr><td>34</td><td>111</td><td>15</td><td>99</td><td>98</td></tr> <tr><td>12</td><td>19</td><td>13</td><td>44</td><td>10</td></tr> </table> Layer 5: <table> <tr><td>76</td><td>27</td><td>123</td><td>4</td><td>11</td></tr> <tr><td>45</td><td>7</td><td>124</td><td>23</td><td>81</td></tr> <tr><td>22</td><td>50</td><td>120</td><td>94</td><td>69</td></tr> <tr><td>67</td><td>55</td><td>106</td><td>26</td><td>95</td></tr> <tr><td>35</td><td>83</td><td>79</td><td></td><td></td></tr> </table>	52	21	85	64	29	20	105	1	57	51	56	9	49	93	114	40	63	74	100	116	125	71	92	82	33	43	103	59	101	108	2	58	88	87	3	86	53	36	115	119	24	37	68	77	107	91	48	97	65	62	16	90	66	18	60	39	5	113	73	80	84	8	31	121	112	89	61	47	25	104	14	70	30	117	122	28	102	38	42	78	72	32	109	41	46	118	110	6	96	54	34	111	15	99	98	12	19	13	44	10	76	27	123	4	11	45	7	124	23	81	22	50	120	94	69	67	55	106	26	95	35	83	79		
52	21	85	64	29																																																																																																																										
20	105	1	57	51																																																																																																																										
56	9	49	93	114																																																																																																																										
40	63	74	100	116																																																																																																																										
125	71	92	82	33																																																																																																																										
43	103	59	101	108																																																																																																																										
2	58	88	87	3																																																																																																																										
86	53	36	115	119																																																																																																																										
24	37	68	77	107																																																																																																																										
91	48	97	65	62																																																																																																																										
16	90	66	18	60																																																																																																																										
39	5	113	73	80																																																																																																																										
84	8	31	121	112																																																																																																																										
89	61	47	25	104																																																																																																																										
14	70	30	117	122																																																																																																																										
28	102	38	42	78																																																																																																																										
72	32	109	41	46																																																																																																																										
118	110	6	96	54																																																																																																																										
34	111	15	99	98																																																																																																																										
12	19	13	44	10																																																																																																																										
76	27	123	4	11																																																																																																																										
45	7	124	23	81																																																																																																																										
22	50	120	94	69																																																																																																																										
67	55	106	26	95																																																																																																																										
35	83	79																																																																																																																												

Objective Awal	7349																																																																																																																													
State Akhir	<p>Final Cube State:</p> <table border="1"> <tr><td>86</td><td>68</td><td>11</td><td>44</td><td>54</td></tr> <tr><td>9</td><td>119</td><td>102</td><td>98</td><td>81</td></tr> <tr><td>125</td><td>60</td><td>111</td><td>6</td><td>2</td></tr> <tr><td>115</td><td>101</td><td>56</td><td>76</td><td>118</td></tr> <tr><td></td><td>19</td><td>35</td><td>35</td><td>112</td></tr> </table> <table border="1"> <tr><td>89</td><td>4</td><td>121</td><td>1</td><td>84</td></tr> <tr><td>22</td><td>64</td><td>85</td><td>79</td><td>82</td></tr> <tr><td>20</td><td>36</td><td>15</td><td>94</td><td>107</td></tr> <tr><td>46</td><td>55</td><td>29</td><td>34</td><td>45</td></tr> <tr><td>52</td><td>59</td><td>99</td><td>43</td><td>45</td></tr> </table> <table border="1"> <tr><td>50</td><td>38</td><td>47</td><td>70</td><td>61</td></tr> <tr><td>87</td><td>114</td><td>51</td><td>103</td><td>110</td></tr> <tr><td>108</td><td>3</td><td>90</td><td>104</td><td>112</td></tr> <tr><td>71</td><td>83</td><td>73</td><td>63</td><td>39</td></tr> <tr><td>113</td><td>25</td><td>92</td><td>12</td><td>39</td></tr> </table> <table border="1"> <tr><td>97</td><td>67</td><td>117</td><td>33</td><td>58</td></tr> <tr><td>95</td><td>16</td><td>31</td><td>109</td><td>57</td></tr> <tr><td>14</td><td>32</td><td>75</td><td>18</td><td>26</td></tr> <tr><td>93</td><td>77</td><td>53</td><td>80</td><td>17</td></tr> <tr><td>30</td><td>96</td><td>65</td><td>40</td><td>17</td></tr> </table> <table border="1"> <tr><td>10</td><td>88</td><td>120</td><td>13</td><td>28</td></tr> <tr><td>105</td><td>21</td><td>5</td><td>8</td><td>37</td></tr> <tr><td>48</td><td>116</td><td>75</td><td>106</td><td>69</td></tr> <tr><td>41</td><td>124</td><td>42</td><td>72</td><td>49</td></tr> <tr><td>122</td><td>23</td><td>27</td><td>91</td><td>49</td></tr> </table>	86	68	11	44	54	9	119	102	98	81	125	60	111	6	2	115	101	56	76	118		19	35	35	112	89	4	121	1	84	22	64	85	79	82	20	36	15	94	107	46	55	29	34	45	52	59	99	43	45	50	38	47	70	61	87	114	51	103	110	108	3	90	104	112	71	83	73	63	39	113	25	92	12	39	97	67	117	33	58	95	16	31	109	57	14	32	75	18	26	93	77	53	80	17	30	96	65	40	17	10	88	120	13	28	105	21	5	8	37	48	116	75	106	69	41	124	42	72	49	122	23	27	91	49
86	68	11	44	54																																																																																																																										
9	119	102	98	81																																																																																																																										
125	60	111	6	2																																																																																																																										
115	101	56	76	118																																																																																																																										
	19	35	35	112																																																																																																																										
89	4	121	1	84																																																																																																																										
22	64	85	79	82																																																																																																																										
20	36	15	94	107																																																																																																																										
46	55	29	34	45																																																																																																																										
52	59	99	43	45																																																																																																																										
50	38	47	70	61																																																																																																																										
87	114	51	103	110																																																																																																																										
108	3	90	104	112																																																																																																																										
71	83	73	63	39																																																																																																																										
113	25	92	12	39																																																																																																																										
97	67	117	33	58																																																																																																																										
95	16	31	109	57																																																																																																																										
14	32	75	18	26																																																																																																																										
93	77	53	80	17																																																																																																																										
30	96	65	40	17																																																																																																																										
10	88	120	13	28																																																																																																																										
105	21	5	8	37																																																																																																																										
48	116	75	106	69																																																																																																																										
41	124	42	72	49																																																																																																																										
122	23	27	91	49																																																																																																																										
Objective Akhir	6297																																																																																																																													
Plot nilai objective function terhadap banyak iterasi yang telah dilewati	<p>Objective Function vs Iterations (Genetic Algorithm)</p> <table border="1"> <caption>Data points estimated from the graph</caption> <thead> <tr> <th>Iterations</th> <th>Max Objective Function</th> <th>Average Objective Function</th> </tr> </thead> <tbody> <tr><td>0</td><td>6500</td><td>7100</td></tr> <tr><td>1</td><td>6400</td><td>6900</td></tr> <tr><td>2</td><td>6400</td><td>7100</td></tr> <tr><td>3</td><td>6400</td><td>6900</td></tr> <tr><td>4</td><td>6300</td><td>6900</td></tr> <tr><td>5</td><td>6300</td><td>6700</td></tr> <tr><td>6</td><td>6300</td><td>7200</td></tr> <tr><td>7</td><td>6300</td><td>7550</td></tr> <tr><td>8</td><td>6300</td><td>6850</td></tr> <tr><td>9</td><td>6300</td><td>6650</td></tr> </tbody> </table>	Iterations	Max Objective Function	Average Objective Function	0	6500	7100	1	6400	6900	2	6400	7100	3	6400	6900	4	6300	6900	5	6300	6700	6	6300	7200	7	6300	7550	8	6300	6850	9	6300	6650																																																																																												
Iterations	Max Objective Function	Average Objective Function																																																																																																																												
0	6500	7100																																																																																																																												
1	6400	6900																																																																																																																												
2	6400	7100																																																																																																																												
3	6400	6900																																																																																																																												
4	6300	6900																																																																																																																												
5	6300	6700																																																																																																																												
6	6300	7200																																																																																																																												
7	6300	7550																																																																																																																												
8	6300	6850																																																																																																																												
9	6300	6650																																																																																																																												

Hasil Eksperimen	<pre> ALGORITHM RESULTS  Algorithm: Genetic Algorithm Initial Objective Value: 7349 Final Objective Value: 6297 Iterations: 10 Duration: 0.8160 seconds Stuck Counter: [] Population Size: 10 </pre>
Jumlah Populasi	10
Banyak Iterasi	10
Durasi Pencarian	0.8160 detik

**Tabel 4.3.12.** Pengujian 4 Populasi sebagai Kontrol

State Awal	<pre> Initial Cube State:  Layer 1:       29   113   20   13   108       100   33   36   115   75       1   122   85   27   49       54   119   99   5   109       31   37   82   17   97  Layer 2:       40   3   6   55   106       74   50   39   89   63       44   120   32   10   121       114   23   116   8   4       28   12   11   123   16  Layer 3:       52   22   73   61   91       56   72   95   111   105       26   90   2   65   24       68   81   47   118   102       78   70   25   18   124  Layer 4:       80   59   51   66   98       34   57   103   14   77       45   21   79   117   107       41   125   58   69   62       9   112   48   35   92  Layer 5:       101   87   67   104   60       71   53   93   84   76       86   38   64   88   7       42   94   15   30   19       110   96   43   46   83 </pre>
Objective Awal	7058

State Akhir	<pre> Final Cube State:  Layer 1:       110 104  9  59  76  42       46   22   89       17   107  34  20  95       23   70   45  88  41       58   115  106 74  114  Layer 2:       30  36  28  69  75       112 31  83       99  90  39  117 27       65  62  67  3  12       125 109 24  94  77  Layer 3:       124 43  91  105 85       79   98  16  108       119 13  80  19  18       120 35  68  61  78       102 1  81  40  64  Layer 4:       29  21  123 111 63       72  47  33  55  38       49  87  60  56       73  84  92  118 50       14  1  81  40  64  Layer 5:       2  10  122 8  54       116 113 52  100       15   4   32  51  101       86  82  71  25  96       121 37  26  57  56     </pre>																																	
Objective Akhir	6141																																	
Plot nilai objective function terhadap banyak iterasi yang telah dilewati	<p>Objective Function vs Iterations (Genetic Algorithm)</p> <table border="1"> <caption>Data for Objective Function vs Iterations</caption> <thead> <tr> <th>Iterations</th> <th>Max Objective Function</th> <th>Average Objective Function</th> </tr> </thead> <tbody> <tr><td>0</td><td>6200</td><td>6900</td></tr> <tr><td>1</td><td>6200</td><td>6600</td></tr> <tr><td>2</td><td>6200</td><td>6550</td></tr> <tr><td>3</td><td>6200</td><td>6550</td></tr> <tr><td>4</td><td>6200</td><td>6550</td></tr> <tr><td>5</td><td>6200</td><td>6800</td></tr> <tr><td>6</td><td>6200</td><td>7350</td></tr> <tr><td>7</td><td>6200</td><td>7300</td></tr> <tr><td>8</td><td>6200</td><td>7100</td></tr> <tr><td>9</td><td>6200</td><td>7250</td></tr> </tbody> </table>	Iterations	Max Objective Function	Average Objective Function	0	6200	6900	1	6200	6600	2	6200	6550	3	6200	6550	4	6200	6550	5	6200	6800	6	6200	7350	7	6200	7300	8	6200	7100	9	6200	7250
Iterations	Max Objective Function	Average Objective Function																																
0	6200	6900																																
1	6200	6600																																
2	6200	6550																																
3	6200	6550																																
4	6200	6550																																
5	6200	6800																																
6	6200	7350																																
7	6200	7300																																
8	6200	7100																																
9	6200	7250																																

Hasil Eksperimen	<pre> ALGORITHM RESULTS  Algorithm: Genetic Algorithm Initial Objective Value: 7058 Final Objective Value: 6141 Iterations: 10 Duration: 0.7819 seconds Stuck Counter: [] Population Size: 10 </pre>
Jumlah Populasi	10
Banyak Iterasi	10
Durasi Pencarian	0.7819 detik

### 5. Pengujian 5 - Iterasi sebagai Kontrol dengan 100 Populasi

**Tabel 4.3.13.** Pengujian 5 Iterasi sebagai Kontrol

State Awal	<pre> Initial Cube State:  Layer 1:       14   65   48   56   40       17   18    9   54   83       16   106   94   7   42       85   15   43   60   64       19   70   22   110  107  Layer 2:       28   49   101  123  122       82   76   74   116  52       90   1    45   47   53       108  89   58   72   79       69   44   29   73   84  Layer 3:       67   91   92   77   112       104  103  33   111  109       8    86   30   115  2       57   11   37   34   118       51   62   20   46   38  Layer 4:       105  117  114  95   98       71   121  81   93   35       50   25   119  12   78       23   5    120  66   68       24   4    21   88   59  Layer 5:       26   63   55   10   39       27   99   102  96   97       125  36   3    113  75       100  124  61   87   31       41   6    88   32   13 </pre>
------------	---

Objective Awal	7386																																	
State Akhir	<p>Final Cube State:</p> <pre> Layer 1:      Final Cube State:       119   88   12   42   17       56   97   76   43   63       87   102  94   77   61       99   29   18   13   45       28   93   31   78   22 Layer 2:       74   113  65   90   115       1   106   3   83   121       112   106  15   91   38       32   52   98   58   84       26   35    8   125  117 Layer 3:       50   79   59   46   14       33   55   124  75   64       6   36   85   69   34       107  109  24   89   105       81   109  20   9   71 Layer 4:       37   7   80   122  11       82   39   67   25   111       48   70   54   72   120       21   123  47   95   103       10   66   41   60   4 Layer 5:       23   118  44   51   49       62   73   16   86   19       116  96   53   5   68       101  92   108  100  2       114  27   104  57   30 </pre>																																	
Objective Akhir	5545																																	
Plot nilai objective function terhadap banyak iterasi yang telah dilewati	<p>Objective Function vs Iterations (Genetic Algorithm)</p> <table border="1"> <caption>Data for Objective Function vs Iterations</caption> <thead> <tr> <th>Iterations</th> <th>Max Objective Function</th> <th>Average Objective Function</th> </tr> </thead> <tbody> <tr><td>0</td><td>5545</td><td>6880</td></tr> <tr><td>1</td><td>5545</td><td>6820</td></tr> <tr><td>2</td><td>5545</td><td>6850</td></tr> <tr><td>3</td><td>5545</td><td>6830</td></tr> <tr><td>4</td><td>5545</td><td>6820</td></tr> <tr><td>5</td><td>5545</td><td>6880</td></tr> <tr><td>6</td><td>5545</td><td>6850</td></tr> <tr><td>7</td><td>5545</td><td>6880</td></tr> <tr><td>8</td><td>5545</td><td>6950</td></tr> <tr><td>9</td><td>5545</td><td>6750</td></tr> </tbody> </table>	Iterations	Max Objective Function	Average Objective Function	0	5545	6880	1	5545	6820	2	5545	6850	3	5545	6830	4	5545	6820	5	5545	6880	6	5545	6850	7	5545	6880	8	5545	6950	9	5545	6750
Iterations	Max Objective Function	Average Objective Function																																
0	5545	6880																																
1	5545	6820																																
2	5545	6850																																
3	5545	6830																																
4	5545	6820																																
5	5545	6880																																
6	5545	6850																																
7	5545	6880																																
8	5545	6950																																
9	5545	6750																																

Hasil Eksperimen	<pre> ALGORITHM RESULTS  Algorithm: Genetic Algorithm Initial Objective Value: 7386 Final Objective Value: 5545 Iterations: 10 Duration: 6.4968 seconds Stuck Counter: [] Population Size: 100 </pre>
Jumlah Populasi	100
Banyak Iterasi	10
Durasi Pencarian	6.4968 detik

**Tabel 4.3.14.** Pengujian 5 Iterasi sebagai Kontrol

State Awal	<pre> Initial Cube State:  Layer 1:       89   41   112   93   48      107   110   21   123   91      102   104   99   58   125       70   118   12   33   82       28   92   67   35   106  Layer 2:       6   20   79   101   84      83   66   65   113   4      32   108   90   56   51      37   43    3   86   60      69   55   22   59   51  Layer 3:       68   109   78   120   100      121   115   36   124   44      39   97   75   40   54      122   19   18   23   54      25   77   10   53   96  Layer 4:       98   14   34   76   26       52   114   117   72   42       5   116    7   17   29      47   16   74   15   81      111   61   24   57   95  Layer 5:       50   73   45   119   63       46   13   11   71   27       94    8   103   80   88       62   31   87   30   64       38    1   49   105   85 </pre>
------------	--

Objective Awal	7114																																																																																																				
State Akhir	<p style="text-align: center;">Final Cube State:</p> <p>Layer 1:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>47</td><td>62</td><td>104</td><td>109</td><td>10</td></tr> <tr><td>19</td><td>76</td><td>110</td><td>90</td><td>66</td></tr> <tr><td>102</td><td>97</td><td>61</td><td>16</td><td>96</td></tr> <tr><td>118</td><td>23</td><td>5</td><td>15</td><td>63</td></tr> </table> <p>Layer 2:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>71</td><td>81</td><td>14</td><td>13</td><td>55</td></tr> <tr><td>53</td><td>24</td><td>27</td><td>91</td><td>79</td></tr> <tr><td>95</td><td>38</td><td>73</td><td>125</td><td>69</td></tr> <tr><td>22</td><td>70</td><td>34</td><td>52</td><td>7</td></tr> </table> <p>Layer 3:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>31</td><td>98</td><td>108</td><td>4</td><td>44</td></tr> <tr><td>60</td><td>49</td><td>20</td><td>123</td><td>36</td></tr> <tr><td>43</td><td>41</td><td>33</td><td>78</td><td>116</td></tr> <tr><td>121</td><td>35</td><td>50</td><td>120</td><td>37</td></tr> </table> <p>Layer 4:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>48</td><td>107</td><td>26</td><td>40</td><td>25</td></tr> <tr><td>86</td><td>80</td><td>122</td><td>39</td><td>58</td></tr> <tr><td>92</td><td>112</td><td>84</td><td>65</td><td>12</td></tr> <tr><td>119</td><td>89</td><td>117</td><td>94</td><td>18</td></tr> </table> <p>Layer 5:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>32</td><td>105</td><td>101</td><td>114</td><td>1</td></tr> <tr><td>124</td><td>106</td><td>45</td><td>83</td><td>85</td></tr> <tr><td>100</td><td>59</td><td>64</td><td>30</td><td>6</td></tr> <tr><td>51</td><td>28</td><td>82</td><td>57</td><td>111</td></tr> </table>	47	62	104	109	10	19	76	110	90	66	102	97	61	16	96	118	23	5	15	63	71	81	14	13	55	53	24	27	91	79	95	38	73	125	69	22	70	34	52	7	31	98	108	4	44	60	49	20	123	36	43	41	33	78	116	121	35	50	120	37	48	107	26	40	25	86	80	122	39	58	92	112	84	65	12	119	89	117	94	18	32	105	101	114	1	124	106	45	83	85	100	59	64	30	6	51	28	82	57	111
47	62	104	109	10																																																																																																	
19	76	110	90	66																																																																																																	
102	97	61	16	96																																																																																																	
118	23	5	15	63																																																																																																	
71	81	14	13	55																																																																																																	
53	24	27	91	79																																																																																																	
95	38	73	125	69																																																																																																	
22	70	34	52	7																																																																																																	
31	98	108	4	44																																																																																																	
60	49	20	123	36																																																																																																	
43	41	33	78	116																																																																																																	
121	35	50	120	37																																																																																																	
48	107	26	40	25																																																																																																	
86	80	122	39	58																																																																																																	
92	112	84	65	12																																																																																																	
119	89	117	94	18																																																																																																	
32	105	101	114	1																																																																																																	
124	106	45	83	85																																																																																																	
100	59	64	30	6																																																																																																	
51	28	82	57	111																																																																																																	
Objective Akhir	5883																																																																																																				
Plot nilai objective function terhadap banyak iterasi yang telah dilewati	<p style="text-align: center;">Objective Function vs Iterations (Genetic Algorithm)</p> <table border="1"> <caption>Data for Objective Function vs Iterations</caption> <thead> <tr> <th>Iterations</th> <th>Max Objective Function</th> <th>Average Objective Function</th> </tr> </thead> <tbody> <tr><td>0</td><td>6100</td><td>7050</td></tr> <tr><td>1</td><td>6100</td><td>7050</td></tr> <tr><td>2</td><td>6100</td><td>7100</td></tr> <tr><td>3</td><td>6100</td><td>7150</td></tr> <tr><td>4</td><td>6100</td><td>7250</td></tr> <tr><td>5</td><td>6100</td><td>7150</td></tr> <tr><td>6</td><td>6100</td><td>7150</td></tr> <tr><td>7</td><td>6100</td><td>7250</td></tr> <tr><td>8</td><td>6100</td><td>7150</td></tr> </tbody> </table>	Iterations	Max Objective Function	Average Objective Function	0	6100	7050	1	6100	7050	2	6100	7100	3	6100	7150	4	6100	7250	5	6100	7150	6	6100	7150	7	6100	7250	8	6100	7150																																																																						
Iterations	Max Objective Function	Average Objective Function																																																																																																			
0	6100	7050																																																																																																			
1	6100	7050																																																																																																			
2	6100	7100																																																																																																			
3	6100	7150																																																																																																			
4	6100	7250																																																																																																			
5	6100	7150																																																																																																			
6	6100	7150																																																																																																			
7	6100	7250																																																																																																			
8	6100	7150																																																																																																			

Hasil Eksperimen	<pre> ALGORITHM RESULTS  Algorithm: Genetic Algorithm Initial Objective Value: 7114 Final Objective Value: 5883 Iterations: 10 Duration: 7.0886 seconds Stuck Counter: [] Population Size: 100 </pre>
Jumlah Populasi	100
Banyak Iterasi	10
Durasi Pencarian	7.0886 detik

**Tabel 4.3.15.** Pengujian 5 Iterasi sebagai Kontrol

State Awal	<pre> Initial Cube State:  Layer 1:       8   110   70   27   104      46   49   93   66   113      31   84   22   45      5      26   106   16   10   19      25   86      4   101      2  Layer 2:       121   73   39   109   60       78   41   124   11   64       59   87      3   63   108       92   54   62   85   112      13   74   80   95      98  Layer 3:       119   76   42   38   82       125   57   88   35   117       34   118   89   44   116       18   99   12   37   116      28   20   15   67      30  Layer 4:       83   122   50   71   96       7   115   6   43   69       17   94   29   103   32       53   114   68   1   48      24   123   55   21      9  Layer 5:       120   52   33   100   14       23   58   105   77   56       40   36   75   51   81       47   79   97   107   102      91   61   111   65      90 </pre>
Objective Awal	8237

State Akhir	<p>Final Cube State:</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr><td>37</td><td>60</td><td>63</td><td>86</td><td>51</td></tr> <tr><td>49</td><td>48</td><td>42</td><td>106</td><td>14</td></tr> <tr><td>12</td><td>7</td><td>98</td><td>114</td><td>95</td></tr> <tr><td>75</td><td>115</td><td>46</td><td>3</td><td>68</td></tr> <tr><td>27</td><td>119</td><td>107</td><td>43</td><td>32</td></tr> </table> <p>Layer 1:</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr><td>71</td><td>121</td><td>104</td><td>90</td><td>50</td></tr> <tr><td>88</td><td>74</td><td>59</td><td>65</td><td>116</td></tr> <tr><td>17</td><td>8</td><td>55</td><td>56</td><td>4</td></tr> <tr><td>96</td><td>33</td><td>85</td><td>77</td><td>76</td></tr> <tr><td>72</td><td>87</td><td>22</td><td>57</td><td>30</td></tr> </table> <p>Layer 2:</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr><td>11</td><td>73</td><td>35</td><td>61</td><td>84</td></tr> <tr><td>117</td><td>29</td><td>93</td><td>19</td><td>105</td></tr> <tr><td>89</td><td>41</td><td>36</td><td>38</td><td>118</td></tr> <tr><td>125</td><td>1</td><td>18</td><td>83</td><td>40</td></tr> <tr><td>99</td><td>78</td><td>39</td><td>62</td><td>101</td></tr> </table> <p>Layer 3:</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr><td>67</td><td>123</td><td>25</td><td>20</td><td>52</td></tr> <tr><td>9</td><td>81</td><td>6</td><td>110</td><td>44</td></tr> <tr><td>100</td><td>69</td><td>26</td><td>31</td><td>91</td></tr> <tr><td>10</td><td>15</td><td>103</td><td>108</td><td>70</td></tr> <tr><td>34</td><td>113</td><td>102</td><td>82</td><td>122</td></tr> </table> <p>Layer 4:</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr><td>94</td><td>21</td><td>120</td><td>23</td><td>80</td></tr> <tr><td>64</td><td>45</td><td>92</td><td>58</td><td>124</td></tr> <tr><td>111</td><td>24</td><td>13</td><td>112</td><td>66</td></tr> <tr><td>16</td><td>79</td><td>54</td><td>28</td><td>53</td></tr> <tr><td>109</td><td>97</td><td>5</td><td>47</td><td>2</td></tr> </table> <p>Layer 5:</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <tr><td>67</td><td>123</td><td>25</td><td>20</td><td>52</td></tr> <tr><td>9</td><td>81</td><td>6</td><td>110</td><td>44</td></tr> <tr><td>100</td><td>69</td><td>26</td><td>31</td><td>91</td></tr> <tr><td>10</td><td>15</td><td>103</td><td>108</td><td>70</td></tr> <tr><td>34</td><td>113</td><td>102</td><td>82</td><td>122</td></tr> </table>	37	60	63	86	51	49	48	42	106	14	12	7	98	114	95	75	115	46	3	68	27	119	107	43	32	71	121	104	90	50	88	74	59	65	116	17	8	55	56	4	96	33	85	77	76	72	87	22	57	30	11	73	35	61	84	117	29	93	19	105	89	41	36	38	118	125	1	18	83	40	99	78	39	62	101	67	123	25	20	52	9	81	6	110	44	100	69	26	31	91	10	15	103	108	70	34	113	102	82	122	94	21	120	23	80	64	45	92	58	124	111	24	13	112	66	16	79	54	28	53	109	97	5	47	2	67	123	25	20	52	9	81	6	110	44	100	69	26	31	91	10	15	103	108	70	34	113	102	82	122
37	60	63	86	51																																																																																																																																																			
49	48	42	106	14																																																																																																																																																			
12	7	98	114	95																																																																																																																																																			
75	115	46	3	68																																																																																																																																																			
27	119	107	43	32																																																																																																																																																			
71	121	104	90	50																																																																																																																																																			
88	74	59	65	116																																																																																																																																																			
17	8	55	56	4																																																																																																																																																			
96	33	85	77	76																																																																																																																																																			
72	87	22	57	30																																																																																																																																																			
11	73	35	61	84																																																																																																																																																			
117	29	93	19	105																																																																																																																																																			
89	41	36	38	118																																																																																																																																																			
125	1	18	83	40																																																																																																																																																			
99	78	39	62	101																																																																																																																																																			
67	123	25	20	52																																																																																																																																																			
9	81	6	110	44																																																																																																																																																			
100	69	26	31	91																																																																																																																																																			
10	15	103	108	70																																																																																																																																																			
34	113	102	82	122																																																																																																																																																			
94	21	120	23	80																																																																																																																																																			
64	45	92	58	124																																																																																																																																																			
111	24	13	112	66																																																																																																																																																			
16	79	54	28	53																																																																																																																																																			
109	97	5	47	2																																																																																																																																																			
67	123	25	20	52																																																																																																																																																			
9	81	6	110	44																																																																																																																																																			
100	69	26	31	91																																																																																																																																																			
10	15	103	108	70																																																																																																																																																			
34	113	102	82	122																																																																																																																																																			
Objective Akhir	5677																																																																																																																																																						
Plot nilai objective function terhadap banyak iterasi yang telah dilewati	<p>Objective Function vs Iterations (Genetic Algorithm)</p> <table border="1"> <caption>Data for Objective Function vs Iterations</caption> <thead> <tr> <th>Iterations</th> <th>Average Objective Function</th> <th>Max Objective Function</th> </tr> </thead> <tbody> <tr><td>0</td><td>6900</td><td>5850</td></tr> <tr><td>1</td><td>6850</td><td>5850</td></tr> <tr><td>2</td><td>6950</td><td>5850</td></tr> <tr><td>3</td><td>6850</td><td>5850</td></tr> <tr><td>4</td><td>7050</td><td>5850</td></tr> <tr><td>5</td><td>7150</td><td>5850</td></tr> <tr><td>6</td><td>7000</td><td>5850</td></tr> <tr><td>7</td><td>7200</td><td>5850</td></tr> <tr><td>8</td><td>7050</td><td>5850</td></tr> <tr><td>9</td><td>7100</td><td>5850</td></tr> </tbody> </table>	Iterations	Average Objective Function	Max Objective Function	0	6900	5850	1	6850	5850	2	6950	5850	3	6850	5850	4	7050	5850	5	7150	5850	6	7000	5850	7	7200	5850	8	7050	5850	9	7100	5850																																																																																																																					
Iterations	Average Objective Function	Max Objective Function																																																																																																																																																					
0	6900	5850																																																																																																																																																					
1	6850	5850																																																																																																																																																					
2	6950	5850																																																																																																																																																					
3	6850	5850																																																																																																																																																					
4	7050	5850																																																																																																																																																					
5	7150	5850																																																																																																																																																					
6	7000	5850																																																																																																																																																					
7	7200	5850																																																																																																																																																					
8	7050	5850																																																																																																																																																					
9	7100	5850																																																																																																																																																					

Hasil Eksperimen	<pre> ALGORITHM RESULTS  Algorithm: Genetic Algorithm Initial Objective Value: 8237 Final Objective Value: 5677 Iterations: 10 Duration: 7.3332 seconds Stuck Counter: [] Population Size: 100 </pre>
Jumlah Populasi	100
Banyak Iterasi	10
Durasi Pencarian	7.3332 detik

## 6. Pengujian 6 - Iterasi sebagai Kontrol dengan 1000 Populasi

**Tabel 4.3.16.** Pengujian 6 Iterasi sebagai Kontrol

State Awal	<pre> Initial Cube State:  Layer 1:       74   82   26   19   14      118   70    2   97  100       48   35   51    3   18       96   91   119   40  102       65   38   121   57   78  Layer 2:       15   60   79   53   124       44   33    7   63  113      120   71   77   80   76        9   107   81   123  104       31   68   43   66   11  Layer 3:       75   109   99   37   10       73   67   101   56   32       24   23   117   20   69       95   83   17   62   54       46   106   29   85   45  Layer 4:        8   25   27   39   52        6   84   30   112  159       115   92   28   22   42        88   98    5   72  122       93   34   86   49   41  Layer 5:       110   47   111   16   50       108   13   36   55   114       125   105   87   58   61         1   94   89   103   64       90   116    4   21   12 </pre>
------------	--

Objective Awal	6390																																																																																																																													
State Akhir	<p>Final Cube State:</p> <p>Layer 1:</p> <table> <tbody> <tr><td>50</td><td>111</td><td>34</td><td>4</td><td>2</td></tr> <tr><td>86</td><td>36</td><td>43</td><td>113</td><td>44</td></tr> <tr><td>31</td><td>95</td><td>51</td><td>61</td><td>25</td></tr> <tr><td>118</td><td>48</td><td>9</td><td>14</td><td>87</td></tr> <tr><td>16</td><td>15</td><td>117</td><td>60</td><td>123</td></tr> </tbody> </table> <p>Layer 2:</p> <table> <tbody> <tr><td>90</td><td>8</td><td>33</td><td>101</td><td>49</td></tr> <tr><td>38</td><td>54</td><td>42</td><td>78</td><td>62</td></tr> <tr><td>109</td><td>107</td><td>20</td><td>99</td><td>88</td></tr> <tr><td>104</td><td>56</td><td>46</td><td>41</td><td>58</td></tr> <tr><td>72</td><td>94</td><td>110</td><td>3</td><td>64</td></tr> </tbody> </table> <p>Layer 3:</p> <table> <tbody> <tr><td>63</td><td>102</td><td>98</td><td>23</td><td>1</td></tr> <tr><td>53</td><td>45</td><td>12</td><td>71</td><td>69</td></tr> <tr><td>120</td><td>39</td><td>108</td><td>121</td><td>85</td></tr> <tr><td>119</td><td>74</td><td>26</td><td>66</td><td>79</td></tr> <tr><td>116</td><td>70</td><td>76</td><td>52</td><td>7</td></tr> </tbody> </table> <p>Layer 4:</p> <table> <tbody> <tr><td>27</td><td>47</td><td>80</td><td>106</td><td>67</td></tr> <tr><td>32</td><td>122</td><td>89</td><td>10</td><td>11</td></tr> <tr><td>5</td><td>29</td><td>65</td><td>18</td><td>97</td></tr> <tr><td>75</td><td>82</td><td>92</td><td>22</td><td>125</td></tr> <tr><td>83</td><td>40</td><td>6</td><td>103</td><td>100</td></tr> </tbody> </table> <p>Layer 5:</p> <table> <tbody> <tr><td>115</td><td>68</td><td>55</td><td>112</td><td>93</td></tr> <tr><td>73</td><td>105</td><td>35</td><td>24</td><td>57</td></tr> <tr><td>28</td><td>30</td><td>77</td><td>124</td><td>114</td></tr> <tr><td>21</td><td>96</td><td>19</td><td>17</td><td>81</td></tr> <tr><td>37</td><td>59</td><td>13</td><td>91</td><td>84</td></tr> </tbody> </table>	50	111	34	4	2	86	36	43	113	44	31	95	51	61	25	118	48	9	14	87	16	15	117	60	123	90	8	33	101	49	38	54	42	78	62	109	107	20	99	88	104	56	46	41	58	72	94	110	3	64	63	102	98	23	1	53	45	12	71	69	120	39	108	121	85	119	74	26	66	79	116	70	76	52	7	27	47	80	106	67	32	122	89	10	11	5	29	65	18	97	75	82	92	22	125	83	40	6	103	100	115	68	55	112	93	73	105	35	24	57	28	30	77	124	114	21	96	19	17	81	37	59	13	91	84
50	111	34	4	2																																																																																																																										
86	36	43	113	44																																																																																																																										
31	95	51	61	25																																																																																																																										
118	48	9	14	87																																																																																																																										
16	15	117	60	123																																																																																																																										
90	8	33	101	49																																																																																																																										
38	54	42	78	62																																																																																																																										
109	107	20	99	88																																																																																																																										
104	56	46	41	58																																																																																																																										
72	94	110	3	64																																																																																																																										
63	102	98	23	1																																																																																																																										
53	45	12	71	69																																																																																																																										
120	39	108	121	85																																																																																																																										
119	74	26	66	79																																																																																																																										
116	70	76	52	7																																																																																																																										
27	47	80	106	67																																																																																																																										
32	122	89	10	11																																																																																																																										
5	29	65	18	97																																																																																																																										
75	82	92	22	125																																																																																																																										
83	40	6	103	100																																																																																																																										
115	68	55	112	93																																																																																																																										
73	105	35	24	57																																																																																																																										
28	30	77	124	114																																																																																																																										
21	96	19	17	81																																																																																																																										
37	59	13	91	84																																																																																																																										
Objective Akhir	5423																																																																																																																													
Plot nilai objective function terhadap banyak iterasi yang telah dilewati	<p>Objective Function vs Iterations (Genetic Algorithm)</p> <table border="1"> <caption>Data for Objective Function vs Iterations</caption> <thead> <tr> <th>Iterations</th> <th>Max Objective Function</th> <th>Average Objective Function</th> </tr> </thead> <tbody> <tr><td>0</td><td>5400</td><td>6900</td></tr> <tr><td>1</td><td>5400</td><td>6920</td></tr> <tr><td>2</td><td>5400</td><td>6950</td></tr> <tr><td>3</td><td>5400</td><td>6980</td></tr> <tr><td>4</td><td>5400</td><td>7000</td></tr> <tr><td>5</td><td>5400</td><td>6950</td></tr> <tr><td>6</td><td>5400</td><td>6980</td></tr> <tr><td>7</td><td>5400</td><td>6950</td></tr> <tr><td>8</td><td>5400</td><td>6850</td></tr> </tbody> </table>	Iterations	Max Objective Function	Average Objective Function	0	5400	6900	1	5400	6920	2	5400	6950	3	5400	6980	4	5400	7000	5	5400	6950	6	5400	6980	7	5400	6950	8	5400	6850																																																																																															
Iterations	Max Objective Function	Average Objective Function																																																																																																																												
0	5400	6900																																																																																																																												
1	5400	6920																																																																																																																												
2	5400	6950																																																																																																																												
3	5400	6980																																																																																																																												
4	5400	7000																																																																																																																												
5	5400	6950																																																																																																																												
6	5400	6980																																																																																																																												
7	5400	6950																																																																																																																												
8	5400	6850																																																																																																																												

Hasil Eksperimen	<pre> ALGORITHM RESULTS  Algorithm: Genetic Algorithm Initial Objective Value: 6390 Final Objective Value: 5423 Iterations: 10 Duration: 70.2945 seconds Stuck Counter: [] Population Size: 1000 </pre>
Jumlah Populasi	1000
Banyak Iterasi	10
Durasi Pencarian	70.2945 detik

**Tabel 4.3.17.** Pengujian 6 Iterasi sebagai Kontrol

State Awal	<pre> Initial Cube State:  Layer 1:       72   101   7   10   106       18   78    14   79   104       114   60    99   90   62       12    93     4   105   15       123   44    74   42   22  Layer 2:       69   85   67   77   81       92    9   13   109   71       100   76   26   17   58       117   125   29   73   45       112   23    19   31   51  Layer 3:       11   88   59   41   110       27   68   47   107   63       55   40   53   108   3       1    32   20   35   83       28   75   84   98   97  Layer 4:       52   50   57   46   115       39   95   120   5   16       86   91   36   54   38       124   24   43   118   33       111   49    6    2   37  Layer 5:       61   48   96   66   70       113   64   102   82   65       116   80   87   25   21       34   121   119   30    8       56   103   94   89   122 </pre>
------------	--

Objective Awal	6792																																																																																																																																																						
State Akhir	<p style="text-align: center;">Final Cube State:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>83</td><td>23</td><td>80</td><td>41</td><td>101</td></tr> <tr><td>114</td><td>106</td><td>24</td><td>104</td><td>77</td></tr> <tr><td>21</td><td>70</td><td>59</td><td>55</td><td>90</td></tr> <tr><td>13</td><td>89</td><td>123</td><td>22</td><td>10</td></tr> <tr><td>32</td><td>35</td><td>98</td><td>96</td><td>49</td></tr> </table> <p style="text-align: center;">Layer 1:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>39</td><td>75</td><td>48</td><td>16</td><td>116</td></tr> <tr><td>60</td><td>66</td><td>108</td><td>44</td><td>125</td></tr> <tr><td>109</td><td>120</td><td>87</td><td>43</td><td>71</td></tr> <tr><td>107</td><td>115</td><td>37</td><td>68</td><td>19</td></tr> <tr><td>6</td><td>20</td><td>47</td><td>122</td><td>46</td></tr> </table> <p style="text-align: center;">Layer 2:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>94</td><td>78</td><td>14</td><td>72</td><td>74</td></tr> <tr><td>88</td><td>18</td><td>67</td><td>33</td><td>95</td></tr> <tr><td>34</td><td>85</td><td>118</td><td>97</td><td>64</td></tr> <tr><td>40</td><td>65</td><td>38</td><td>7</td><td>4</td></tr> <tr><td>63</td><td>28</td><td>99</td><td>31</td><td>81</td></tr> </table> <p style="text-align: center;">Layer 3:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>15</td><td>121</td><td>56</td><td>82</td><td>8</td></tr> <tr><td>113</td><td>61</td><td>53</td><td>25</td><td>9</td></tr> <tr><td>50</td><td>51</td><td>52</td><td>112</td><td>91</td></tr> <tr><td>124</td><td>57</td><td>1</td><td>92</td><td>73</td></tr> <tr><td>45</td><td>76</td><td>5</td><td>110</td><td>73</td></tr> </table> <p style="text-align: center;">Layer 4:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>62</td><td>102</td><td>111</td><td>86</td><td>42</td></tr> <tr><td>58</td><td>69</td><td>2</td><td>105</td><td>36</td></tr> <tr><td>79</td><td>119</td><td>3</td><td>26</td><td>93</td></tr> <tr><td>54</td><td>30</td><td>84</td><td>103</td><td>29</td></tr> <tr><td>117</td><td>11</td><td>12</td><td>27</td><td>100</td></tr> </table> <p style="text-align: center;">Layer 5:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>62</td><td>102</td><td>111</td><td>86</td><td>42</td></tr> <tr><td>58</td><td>69</td><td>2</td><td>105</td><td>36</td></tr> <tr><td>79</td><td>119</td><td>3</td><td>26</td><td>93</td></tr> <tr><td>54</td><td>30</td><td>84</td><td>103</td><td>29</td></tr> <tr><td>117</td><td>11</td><td>12</td><td>27</td><td>100</td></tr> </table>	83	23	80	41	101	114	106	24	104	77	21	70	59	55	90	13	89	123	22	10	32	35	98	96	49	39	75	48	16	116	60	66	108	44	125	109	120	87	43	71	107	115	37	68	19	6	20	47	122	46	94	78	14	72	74	88	18	67	33	95	34	85	118	97	64	40	65	38	7	4	63	28	99	31	81	15	121	56	82	8	113	61	53	25	9	50	51	52	112	91	124	57	1	92	73	45	76	5	110	73	62	102	111	86	42	58	69	2	105	36	79	119	3	26	93	54	30	84	103	29	117	11	12	27	100	62	102	111	86	42	58	69	2	105	36	79	119	3	26	93	54	30	84	103	29	117	11	12	27	100
83	23	80	41	101																																																																																																																																																			
114	106	24	104	77																																																																																																																																																			
21	70	59	55	90																																																																																																																																																			
13	89	123	22	10																																																																																																																																																			
32	35	98	96	49																																																																																																																																																			
39	75	48	16	116																																																																																																																																																			
60	66	108	44	125																																																																																																																																																			
109	120	87	43	71																																																																																																																																																			
107	115	37	68	19																																																																																																																																																			
6	20	47	122	46																																																																																																																																																			
94	78	14	72	74																																																																																																																																																			
88	18	67	33	95																																																																																																																																																			
34	85	118	97	64																																																																																																																																																			
40	65	38	7	4																																																																																																																																																			
63	28	99	31	81																																																																																																																																																			
15	121	56	82	8																																																																																																																																																			
113	61	53	25	9																																																																																																																																																			
50	51	52	112	91																																																																																																																																																			
124	57	1	92	73																																																																																																																																																			
45	76	5	110	73																																																																																																																																																			
62	102	111	86	42																																																																																																																																																			
58	69	2	105	36																																																																																																																																																			
79	119	3	26	93																																																																																																																																																			
54	30	84	103	29																																																																																																																																																			
117	11	12	27	100																																																																																																																																																			
62	102	111	86	42																																																																																																																																																			
58	69	2	105	36																																																																																																																																																			
79	119	3	26	93																																																																																																																																																			
54	30	84	103	29																																																																																																																																																			
117	11	12	27	100																																																																																																																																																			
Objective Akhir	5375																																																																																																																																																						
Plot nilai objective function terhadap banyak iterasi yang telah dilewati	<p style="text-align: center;">Objective Function vs Iterations (Genetic Algorithm)</p> <table border="1"> <caption>Data for Objective Function vs Iterations</caption> <thead> <tr> <th>Iterations</th> <th>Max Objective Function</th> <th>Average Objective Function</th> </tr> </thead> <tbody> <tr><td>0</td><td>5600</td><td>6900</td></tr> <tr><td>1</td><td>5600</td><td>6900</td></tr> <tr><td>2</td><td>5400</td><td>6900</td></tr> <tr><td>3</td><td>5400</td><td>6900</td></tr> <tr><td>4</td><td>5400</td><td>6850</td></tr> <tr><td>5</td><td>5400</td><td>6850</td></tr> <tr><td>6</td><td>5400</td><td>6900</td></tr> <tr><td>7</td><td>5400</td><td>6900</td></tr> <tr><td>8</td><td>5400</td><td>6950</td></tr> </tbody> </table>	Iterations	Max Objective Function	Average Objective Function	0	5600	6900	1	5600	6900	2	5400	6900	3	5400	6900	4	5400	6850	5	5400	6850	6	5400	6900	7	5400	6900	8	5400	6950																																																																																																																								
Iterations	Max Objective Function	Average Objective Function																																																																																																																																																					
0	5600	6900																																																																																																																																																					
1	5600	6900																																																																																																																																																					
2	5400	6900																																																																																																																																																					
3	5400	6900																																																																																																																																																					
4	5400	6850																																																																																																																																																					
5	5400	6850																																																																																																																																																					
6	5400	6900																																																																																																																																																					
7	5400	6900																																																																																																																																																					
8	5400	6950																																																																																																																																																					

Hasil Eksperimen	<pre> ALGORITHM RESULTS  Algorithm: Genetic Algorithm Initial Objective Value: 6792 Final Objective Value: 5375 Iterations: 10 Duration: 71.8978 seconds Stuck Counter: [] Population Size: 1000 </pre>
Jumlah Populasi	1000
Banyak Iterasi	10
Durasi Pencarian	71.8978 detik

**Tabel 4.3.18.** Pengujian 6 Iterasi sebagai Kontrol

State Awal	<pre> Initial Cube State:  Layer 1:       20   107   88   124   83       5    84   58   104   36       98   91   27   51   108       35   43   61   41   125  Layer 2:       24   96   65   40   45       94   67   46   31   114       34   92   80   10   22       110   119   123   109   66       11    17   120   39   86  Layer 3:       18   76   70   37   50       29   57   93   28   95       106   81   103   71   90       48   85   79   117   6       19   13   64   33   97  Layer 4:       101   118   102   32   23       54   62   72   99   44       53   115   113   112   122       52   100    4   14   15       25   49   56   55   63  Layer 5:       42   75   38   12    7       77   89   116   121   1       21   69    8   78   26       16   30   59   60   26       74   87    3   105   82 </pre>
------------	--

Objective Awal	7168																																																																																																																													
State Akhir	<p style="text-align: center;">Final Cube State:</p> <p>Layer 1:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>116</td><td>37</td><td>82</td><td>98</td><td>103</td></tr> <tr><td>22</td><td>117</td><td>125</td><td>35</td><td>118</td></tr> <tr><td>9</td><td>91</td><td>45</td><td>96</td><td>58</td></tr> <tr><td>115</td><td>8</td><td>94</td><td>88</td><td>87</td></tr> <tr><td>122</td><td>51</td><td>55</td><td>111</td><td>34</td></tr> </table> <p>Layer 2:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>47</td><td>49</td><td>121</td><td>102</td><td>74</td></tr> <tr><td>52</td><td>73</td><td>78</td><td>23</td><td>20</td></tr> <tr><td>83</td><td>64</td><td>24</td><td>67</td><td>63</td></tr> <tr><td>99</td><td>13</td><td>54</td><td>11</td><td>109</td></tr> <tr><td>95</td><td>57</td><td>53</td><td>101</td><td></td></tr> </table> <p>Layer 3:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>46</td><td>93</td><td>40</td><td>92</td><td>1</td></tr> <tr><td>72</td><td>19</td><td>3</td><td>25</td><td>10</td></tr> <tr><td>30</td><td>32</td><td>85</td><td>16</td><td>123</td></tr> <tr><td>48</td><td>15</td><td>108</td><td>81</td><td>101</td></tr> <tr><td>44</td><td>12</td><td>79</td><td>62</td><td>106</td></tr> </table> <p>Layer 4:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>69</td><td>14</td><td>124</td><td>18</td><td>59</td></tr> <tr><td>107</td><td>38</td><td>75</td><td>97</td><td>50</td></tr> <tr><td>84</td><td>39</td><td>17</td><td>29</td><td>61</td></tr> <tr><td>56</td><td>113</td><td>90</td><td>31</td><td>68</td></tr> <tr><td>6</td><td>120</td><td>33</td><td>71</td><td>86</td></tr> </table> <p>Layer 5:</p> <table style="margin-left: auto; margin-right: auto;"> <tr><td>42</td><td>36</td><td>41</td><td>66</td><td>76</td></tr> <tr><td>89</td><td>65</td><td>26</td><td>70</td><td>43</td></tr> <tr><td>5</td><td>112</td><td>80</td><td>119</td><td>110</td></tr> <tr><td>4</td><td>104</td><td>114</td><td>60</td><td>21</td></tr> <tr><td>7</td><td>77</td><td>28</td><td>105</td><td>2</td></tr> </table>	116	37	82	98	103	22	117	125	35	118	9	91	45	96	58	115	8	94	88	87	122	51	55	111	34	47	49	121	102	74	52	73	78	23	20	83	64	24	67	63	99	13	54	11	109	95	57	53	101		46	93	40	92	1	72	19	3	25	10	30	32	85	16	123	48	15	108	81	101	44	12	79	62	106	69	14	124	18	59	107	38	75	97	50	84	39	17	29	61	56	113	90	31	68	6	120	33	71	86	42	36	41	66	76	89	65	26	70	43	5	112	80	119	110	4	104	114	60	21	7	77	28	105	2
116	37	82	98	103																																																																																																																										
22	117	125	35	118																																																																																																																										
9	91	45	96	58																																																																																																																										
115	8	94	88	87																																																																																																																										
122	51	55	111	34																																																																																																																										
47	49	121	102	74																																																																																																																										
52	73	78	23	20																																																																																																																										
83	64	24	67	63																																																																																																																										
99	13	54	11	109																																																																																																																										
95	57	53	101																																																																																																																											
46	93	40	92	1																																																																																																																										
72	19	3	25	10																																																																																																																										
30	32	85	16	123																																																																																																																										
48	15	108	81	101																																																																																																																										
44	12	79	62	106																																																																																																																										
69	14	124	18	59																																																																																																																										
107	38	75	97	50																																																																																																																										
84	39	17	29	61																																																																																																																										
56	113	90	31	68																																																																																																																										
6	120	33	71	86																																																																																																																										
42	36	41	66	76																																																																																																																										
89	65	26	70	43																																																																																																																										
5	112	80	119	110																																																																																																																										
4	104	114	60	21																																																																																																																										
7	77	28	105	2																																																																																																																										
Objective Akhir	5464																																																																																																																													
Plot nilai objective function terhadap banyak iterasi yang telah dilewati	<p style="text-align: center;">Objective Function vs Iterations (Genetic Algorithm)</p> <p>The graph illustrates the performance of a Genetic Algorithm over 9 iterations. The X-axis represents the Iterations (0 to 9), and the Y-axis represents the Objective Function Value (5400 to 7000). The blue line, labeled 'Max Objective Function', shows a constant value of approximately 5450 across all iterations. The orange line, labeled 'Average Objective Function', starts at approximately 6950, fluctuates slightly between 6850 and 7000, and stabilizes around 6950 by the final iteration.</p> <table border="1"> <caption>Data extracted from the Objective Function vs Iterations plot</caption> <thead> <tr> <th>Iterations</th> <th>Max Objective Function</th> <th>Average Objective Function</th> </tr> </thead> <tbody> <tr><td>0</td><td>5450</td><td>6950</td></tr> <tr><td>1</td><td>5450</td><td>6950</td></tr> <tr><td>2</td><td>5450</td><td>6950</td></tr> <tr><td>3</td><td>5450</td><td>6950</td></tr> <tr><td>4</td><td>5450</td><td>6950</td></tr> <tr><td>5</td><td>5450</td><td>6950</td></tr> <tr><td>6</td><td>5450</td><td>6850</td></tr> <tr><td>7</td><td>5450</td><td>6950</td></tr> <tr><td>8</td><td>5450</td><td>6950</td></tr> <tr><td>9</td><td>5450</td><td>6950</td></tr> </tbody> </table>	Iterations	Max Objective Function	Average Objective Function	0	5450	6950	1	5450	6950	2	5450	6950	3	5450	6950	4	5450	6950	5	5450	6950	6	5450	6850	7	5450	6950	8	5450	6950	9	5450	6950																																																																																												
Iterations	Max Objective Function	Average Objective Function																																																																																																																												
0	5450	6950																																																																																																																												
1	5450	6950																																																																																																																												
2	5450	6950																																																																																																																												
3	5450	6950																																																																																																																												
4	5450	6950																																																																																																																												
5	5450	6950																																																																																																																												
6	5450	6850																																																																																																																												
7	5450	6950																																																																																																																												
8	5450	6950																																																																																																																												
9	5450	6950																																																																																																																												

Hasil Eksperimen	<pre>ALGORITHM RESULTS  Algorithm: Genetic Algorithm Initial Objective Value: 7168 Final Objective Value: 5464 Iterations: 10 Duration: 72.0271 seconds Stuck Counter: [] Population Size: 1000</pre>
Jumlah Populasi	1000
Banyak Iterasi	10
Durasi Pencarian	72.0271 detik

## 5. Analisis

### 5.1 Tingkat Kedekatan Tiap Algoritma terhadap Global Optima

**Simulated Annealing** dan **Steepest Ascent Hill Climbing** adalah dua algoritma yang menghasilkan nilai objektif paling dekat dengan global optima. Namun, Simulated Annealing memiliki kelebihan dalam mencari ruang kemungkinan solusi yang lebih luas karena pengaturan suhu tinggi dan *cooling rate* yang lambat, tetapi membutuhkan waktu yang lama. Lalu, Sideways Move menunjukkan hasil yang baik dalam mendekati global optima karena dapat memilih *neighbor* yang memiliki *objective function* bernilai sama ketika terjebak di local optima. Dengan maksimum *sideways* yang cukup besar, Sideways Move mendekati global optima lebih baik dibanding algoritma lainnya, tetapi membutuhkan waktu yang lebih lama. Random Restart, Stochastic, dan Genetic Algorithm cenderung kurang optimal dalam mendekati global optima dalam kondisi yang ditentukan, tetapi Genetic Algorithm memiliki potensi lebih baik jika diterapkan dengan ukuran populasi dan iterasi yang cukup besar.

### 5.2 Perbandingan Hasil Pencarian Tiap-Tiap Algoritma Dengan Algoritma Local Search yang Lain

- a. **Simulated Annealing** dan **Steepest Ascent Hill Climbing** lebih optimal dalam mencapai nilai objektif terendah, terutama ketika jumlah iterasi banyak. **Simulated Annealing** lebih fleksibel karena dapat memilih *neighbor* dengan nilai *objective* yang lebih buruk untuk mengeksplorasi kemungkinan yang lebih luas, sedangkan **Steepest Ascent** memerlukan lebih banyak waktu dalam proses *searching* tetapi menghasilkan nilai *objective* rendah yang lebih pasti.
- b. **Sideways Move** berada di antara Simulated Annealing dan Steepest Ascent Hill Climbing dengan kemampuan memilih *neighbor* yang memiliki nilai *objective* sama dengan *current*, sehingga dapat menghindari *stuck* di *local minima* tetapi cenderung menghabiskan waktu yang lebih lama dan tidak menemukan solusi yang lebih baik dibandingkan Steepest Ascent.
- c. **Random Restart** dan **Stochastic Hill Climbing** membutuhkan waktu yang lebih sedikit dalam proses *searching*, tetapi tidak optimal untuk mendapatkan solusi terbaik dibandingkan algoritma Hill Climbing yang lain.

- d. **Genetic Algorithm** membutuhkan waktu yang lebih sedikit tetapi menghasilkan nilai *objective* yang lebih tinggi sehingga algoritma ini lebih mengutamakan kecepatan daripada nilai *objective* yang optimal.

### 5.3 Perbandingan Durasi Proses Pencarian Tiap Algoritma Terhadap Algoritma Lainnya

- a. **Steepest Ascent** berkisar antara 76.5 detik sampai 186.9 detik. Dengan peningkatan jumlah iterasi, waktu yang dibutuhkan semakin lama. Iterasi 50 membutuhkan 76.5 detik, sedangkan iterasi 120 membutuhkan waktu hampir tiga kali lipat lebih lama (186.9 detik).
- b. **Random Restart** berkisar antara 26.3 detik sampai 86.3 detik. Waktu pencarian meningkat seiring dengan peningkatan jumlah restarts dan iterasi di setiap *restart*. Contohnya, 10 *restart* dengan 5 iterasi membutuhkan 86.3 detik, sedangkan 3 *restart* dengan 5 iterasi hanya membutuhkan 26.3 detik. Random Restart membutuhkan durasi yang relatif panjang dibandingkan dengan Simulated Annealing, namun lebih cepat dibandingkan Steepest Ascent ketika iterasi rendah.
- c. **Sideways Move** berkisar antara 181.8 detik sampai 217.1 Sideways move merupakan algoritma dengan durasi paling lama karena adanya kemampuan memilih *neighbor* yang memiliki nilai *objective function* sama sehingga eksplorasi solusi lebih luas.
- d. **Stochastic** berkisar antara 21.9 detik sampai 70.6 detik. Meningkatkan jumlah iterasi akan menambah durasi. Contohnya, iterasi 100 membutuhkan 21.9 detik dan iterasi 300 membutuhkan waktu 70.6 detik. Stochastic termasuk cepat dibandingkan dengan Steepest Ascent, Sideways Move, dan Simulated Annealing, terutama ketika jumlah iterasi sedikit.
- e. **Simulated Annealing** berkisar antara 2 detik hingga 79.8 detik, bergantung pada parameter suhu, *cooling rate*, dan iterasi maksimum. Algoritma dengan suhu awal yang tinggi (20000) dan *cooling rate* yang lambat (0.999) memiliki durasi terlama (79.8 detik) karena eksplorasi solusi yang lebih luas dan proses pendinginan yang lambat. Sedangkan algoritma dengan suhu awal lebih rendah (1000) dan *cooling rate* lebih cepat (0.98) memiliki durasi paling singkat (2 detik).
- f. **Genetic Algorithm** berkisar dari 0.6631 detik sampai 72.027 detik, tergantung pada jumlah iterasi dan ukuran populasi. Durasi paling singkat (0.6631 detik) yaitu pada iterasi 10 dan populasi 10. Sedangkan durasi

paling lama (72.027 detik) terjadi pada iterasi 10 dengan populasi 1000. Ketika jumlah iterasi sedikit dan populasi kecil, Genetic Algorithm membutuhkan waktu yang jauh lebih singkat daripada algoritma yang lain.

Dapat disimpulkan bahwa:

- Algoritma tercepat: **Genetic Algorithm** dengan iterasi rendah dan populasi kecil.
- Algoritma yang cukup cepat: **Stochastic** pada iterasi rendah dan **Random Restart** dengan sedikit *restart* dan iterasi rendah.
- Algoritma dengan durasi sedang: **Simulated Annealing** tergantung pada pengaturan suhu dan *cooling rate*.
- Algoritma lambat: **Steepest Ascent** dan **Sideways Move**, terutama pada iterasi tinggi atau jumlah pergerakan *sideways* yang besar.

#### 5.4 Konsistensi Hasil Akhir yang Didapatkan dari Tiap-Tiap Eksperimen yang Dilakukan

- a. **Steepest Ascent Hill Climbing** menunjukkan konsistensi dalam menghasilkan nilai *objective function* yang rendah, terutama ketika jumlah iterasi tinggi. Hasil eksperimen menunjukkan bahwa algoritma ini terus menurunkan nilai *objective function* secara signifikan dengan peningkatan iterasi, misalnya pada iterasi ke-120, nilai *objective function* dapat turun hingga 272. Namun, algoritma ini membutuhkan waktu yang lebih lama untuk mencapai hasil tersebut, karena mengevaluasi semua *neighbor* di setiap iterasi. Steepest Ascent cenderung konsisten dalam mencapai nilai yang rendah, tetapi karena kurangnya fleksibilitas dalam menerima solusi sementara yang lebih buruk, algoritma ini rentan terjebak di local minima.
- b. **Sideways Move Hill Climbing** menunjukkan tingkat konsistensi yang cukup baik dalam menurunkan nilai *objective function*, khususnya saat batas maksimum *sideways* yang diperbolehkan lebih tinggi. Pada eksperimen dengan batas *sideways* 10, algoritma berhasil mencapai nilai *objective* 267, menunjukkan kemampuan untuk menghindari *local* minima yang lebih dalam. Konsistensi hasil akhir juga dipengaruhi oleh jumlah *sideways* yang diperbolehkan untuk meningkatkan kemungkinan mencapai nilai *objective function* yang lebih rendah. Secara keseluruhan, Sideways Move menunjukkan konsistensi dalam mencapai solusi yang baik meskipun dengan waktu *searching* yang lebih lama.

- c. **Random Restart Hill Climbing** menunjukkan hasil akhir yang lebih bervariasi dan kurang konsisten dalam mencapai nilai *objective function* yang rendah. Konsistensi algoritma ini sangat bergantung pada jumlah *restart* dan iterasi per *restart* yang diterapkan. Misalnya, pada eksperimen dengan 10 *restart* dan 5 iterasi per *restart*, nilai *objective function* tercatat di angka 4345. Walaupun *restart* membantu algoritma keluar dari *local* minima, hasil akhirnya masih kurang optimal dibandingkan algoritma lain seperti Simulated Annealing atau Steepest Ascent. Dengan demikian, Random Restart kurang konsisten dalam menghasilkan solusi terbaik, terutama jika *restart* atau iterasi per *restart* terlalu sedikit.
- d. **Stochastic Hill Climbing** memperlihatkan tingkat konsistensi yang lebih rendah dibandingkan algoritma lainnya, dengan nilai *objective function* akhir yang cenderung tetap tinggi meskipun iterasi ditingkatkan. Karena *neighbor* dipilih secara acak tanpa memperhatikan seluruh ruang pencarian, algoritma ini menghasilkan hasil akhir yang bervariasi dan kurang optimal. Misalnya, meskipun dijalankan hingga 300 iterasi, nilai *objective function* hanya turun hingga 3128. Hal ini menunjukkan bahwa pencarian secara acak pada Stochastic Hill Climbing mengakibatkan kurangnya konsistensi dalam mencapai solusi terbaik dan lebih rentan terhadap ketidakstabilan antar iterasi.
- e. **Simulated Annealing** menunjukkan hasil yang relatif konsisten dalam mencapai nilai *objective function* yang rendah. Algoritma ini memanfaatkan parameter temperatur dan *cooling rate* sehingga lebih fleksibel dalam menerima solusi sementara yang lebih buruk di awal pencarian. Konsistensi ini terlihat dari eksperimen dengan variasi temperatur tinggi dan *cooling rate* lambat sehingga menghasilkan penurunan nilai *objective function* yang signifikan meskipun dengan waktu *searching* yang lebih lama. Misalnya, pada eksperimen dengan temperatur awal 20000 dan cooling rate 0.999, nilai *objective* dapat diturunkan menjadi 375. Penyesuaian ini menunjukkan bahwa Simulated Annealing efektif dalam mencapai hasil yang lebih konsisten dan mendekati global optima dengan waktu yang lebih cepat dari Steepest Ascent.
- f. **Genetic Algorithm** menunjukkan konsistensi yang cukup baik dalam menghasilkan nilai akhir. Algoritma ini memberikan hasil yang konsisten dalam jangka pendek, tetapi kualitas solusinya terbatas oleh ukuran populasi dan jumlah iterasi. Genetic Algorithm lebih berfokus pada kecepatan dibandingkan eksplorasi ruang pencarian yang optimal,

sehingga hasil akhirnya konsisten pada nilai akhir yang buruk dibandingkan algoritma lainnya.

## 5.5 Pengaruh Jumlah Iterasi dan Jumlah Populasi Terhadap Hasil Akhir Pencarian pada Genetic Algorithm

### 1. Pengaruh Iterasi

Berdasarkan hasil yang diperoleh, terlihat bahwa peningkatan jumlah iterasi pada Genetic Algorithm memberikan perubahan positif terhadap penurunan nilai *objective function*, namun juga berbanding lurus dengan durasi pencarian. Pada variasi iterasi pertama, yaitu 10 kali iterasi, nilai *objective function* turun dari 7133 menjadi 6297 dengan durasi pencarian sebesar 0,6631 detik. Ini menunjukkan bahwa Genetic Algorithm mampu melakukan eksplorasi awal untuk mencari solusi yang lebih baik dalam waktu singkat, tetapi penurunan nilai *objective function* yang dihasilkan tidak signifikan. Saat variasi iterasi dinaikkan menjadi 100, nilai *objective function* berkurang lebih jauh menjadi 5458, dengan durasi pencarian yang juga meningkat menjadi 7,5638 detik. Hal ini menunjukkan bahwa jumlah iterasi yang lebih banyak memberi kesempatan algoritma untuk melakukan seleksi, crossover, dan mutasi lebih intensif, sehingga berpotensi menghasilkan solusi yang lebih optimal dengan *trade-off* berupa waktu yang lebih lama.

Pada iterasi ke-1000, nilai *objective function* berhasil diturunkan lebih signifikan lagi menjadi 5324 dengan durasi yang meningkat menjadi 73,9 detik. Hasil ini memperlihatkan bahwa semakin banyak iterasi yang dijalankan, algoritma memiliki peluang lebih besar untuk mengoptimalkan kualitas solusi tetapi tetap tidak signifikan. Peningkatan jumlah iterasi juga memperpanjang waktu yang dibutuhkan sehingga terjadi *trade-off* antara kualitas solusi dan efisiensi waktu.

### 2. Pengaruh Populasi

Berdasarkan hasil yang diperoleh, terlihat bahwa peningkatan jumlah populasi pada Genetic Algorithm memberikan perubahan positif terhadap penurunan nilai *objective function*, namun juga diiringi dengan peningkatan durasi pencarian. Pada variasi pertama, yaitu dengan populasi sebesar 10 individu dan 10 kali iterasi, nilai *objective function* turun dari 6615 menjadi 5932, dengan durasi pencarian sebesar 0,7908 detik. Hasil ini menunjukkan bahwa populasi yang lebih kecil

memungkinkan algoritma untuk mencapai solusi yang lebih baik dalam waktu singkat, meskipun ruang pencarian terbatas sehingga kualitas solusi mungkin belum optimal. Ketika ukuran populasi dinaikkan menjadi 100 individu dengan jumlah iterasi yang sama, nilai *objective function* berhasil turun lebih jauh menjadi 5545 dengan durasi pencarian yang meningkat menjadi 6,4968 detik. Hal ini mengindikasikan bahwa populasi yang lebih besar menyediakan lebih banyak variasi solusi, yang meningkatkan kemungkinan algoritma menemukan solusi yang lebih optimal, meskipun membutuhkan waktu yang lebih lama.

Pada variasi populasi terbesar, yaitu 1000 individu dengan iterasi yang sama, nilai *objective function* berhasil diturunkan lebih signifikan menjadi 5375, namun durasi pencarian meningkat drastis menjadi 71,8978 detik. Hasil ini memperlihatkan bahwa semakin besar ukuran populasi, algoritma memiliki peluang lebih tinggi untuk menghasilkan solusi yang lebih baik karena lebih banyak individu yang dapat diolah dalam proses seleksi, crossover, dan mutasi. Akan tetapi, semakin besar ukuran populasi juga berbanding lurus dengan peningkatan durasi pencarian, sehingga terjadi *trade-off* antara kualitas hasil dan efisiensi waktu.

## 6. Kesimpulan dan Saran

### 6.1 Kesimpulan

Berdasarkan analisis yang dilakukan pada berbagai algoritma optimasi, disimpulkan bahwa setiap algoritma memiliki keunggulan dan kelemahan dalam mencapai global optima, durasi eksekusi, dan konsistensi hasil. Algoritma seperti Simulated Annealing dan Steepest Ascent Hill Climbing menunjukkan kinerja yang baik dalam mendekati global optima, terutama dalam kondisi dengan banyak iterasi. Simulated Annealing fleksibel dalam menerima solusi sementara yang kurang baik di awal pencarian, sehingga lebih efektif dalam eksplorasi solusi yang luas meskipun memerlukan waktu lebih lama. Steepest Ascent, meskipun cenderung konsisten tetapi rentan terhadap jebakan local minima, sehingga diperlukan iterasi yang banyak untuk mencapai global optima.

Sementara itu, Sideways Move mampu mencapai solusi yang cukup baik dengan kemampuan menghindari jebakan *local* minima melalui pemilihan *neighbor* yang sama, tetapi membutuhkan waktu yang lebih lama. Random Restart dan Stochastic Hill Climbing, di sisi lain, cenderung lebih cepat, tetapi kurang konsisten dalam mencapai solusi yang optimal.

Genetic Algorithm memberikan hasil yang kurang baik dalam durasi yang lebih singkat, terutama pada populasi dan iterasi rendah. Kemudian, kualitas hasil akhir bergantung pada ukuran populasi dan iterasi, di mana jumlah yang lebih besar meningkatkan peluang mendapatkan solusi yang lebih baik tetapi memperpanjang waktu *searching*.

Secara keseluruhan, performa algoritma ini dipengaruhi oleh faktor-faktor seperti jumlah iterasi, ukuran populasi, dan parameter spesifik pada masing-masing algoritma. *Trade-off* antara kecepatan dan kualitas hasil perlu dipertimbangkan dalam pemilihan algoritma yang sesuai dengan kebutuhan masalah.

### 6.2 Saran

1. **Pilih algoritma berdasarkan kebutuhan optimalisasi.** Jika kualitas hasil lebih diutamakan daripada waktu, Simulated Annealing atau Steepest Ascent Hill Climbing dengan jumlah iterasi tinggi dapat dipertimbangkan untuk dipilih. Dari sisi kecepatan, Genetic Algorithm dengan populasi dan iterasi rendah dapat dipilih.

2. **Atur parameter algoritma secara tepat.** Untuk mendapatkan hasil terbaik, lakukan tuning parameter seperti suhu dan *cooling rate* pada Simulated Annealing atau jumlah populasi dan iterasi pada Genetic Algorithm. Eksperimen dengan berbagai parameter dapat membantu menemukan solusi yang optimal.
3. **Sesuaikan jumlah iterasi dengan batasan waktu yang ada.** Jika waktu *searching* menjadi kendala, algoritma yang lebih cepat seperti Stochastic Hill Climbing atau Random Restart dapat dipertimbangkan dengan jumlah iterasi yang cukup rendah.

## 7. Pembagian Tugas Tiap Anggota Kelompok

Berikut adalah tabel rincian mengenai pendistribusian kerja oleh anggota kelompok:

**Tabel 7.1.** Pembagian Tugas

No	NIM Anggota	Nama Anggota	Deskripsi
1	18222106	Audra Zelvania Putri Harjanto	Algoritma Hill Climbing, Laporan
2	18222118	Rizqi Andhika Pratama	Algoritma Genetic Algorithm, Laporan
3	18222125	Sekar Anindita Nurjadini	Algoritma Simulated Annealing, Laporan
4	18222138	Khayla Belva Annandira	Algoritma Hill Climbing, Laporan

## 8. Lampiran

Repository tempat penyimpanan program dapat diakses melalui [link](#) berikut.

## Referensi

- [Features of the magic cube - Magisch vierkant](#)
- [Perfect Magic Cubes \(trump.de\)](#)
- [Magic cube - Wikipedia](#)
- [Rubiks Professors Cube 5 x 5 x 5 \(grubiks.com\)](#)
- [Beyond Classical Search - Local Search \(Edunex\)](#)
- [Beyond Classical Search - Hill Climbing \(Edunex\)](#)
- [Beyong Classical Search - Simulated Annealing \(Edunex\)](#)
- [Beyond Classical Search - Genetic Algorithm \(Edunex\)](#)