

LAPORAN TUGAS BESAR 1
IF3170 Inteligensi Artifisial
Pencarian Solusi Penjadwalan Kelas Mingguan
dengan Local Search



Disusun Oleh:
Orvin Andika Ikhsan Abhista - 13523017
Fajar Kurniawan - 13523027
Reza Ahmad Syarif - 13523119

DAFTAR ISI

DAFTAR ISI.....	2
BAB 1 DESKRIPSI PERSOALAN.....	3
BAB 2 PEMBAHASAN.....	4
1. Pemilihan Objective Function.....	4
2. Penjelasan Implementasi Algoritma Local Search.....	4
3. Hasil Eksperimen dan Analisis.....	17
3.1.1. Hill Climbing.....	21
3.1.2. Simulated Annealing.....	45
3.1.3. Genetic Algorithm.....	53
3.1.4. Heuristik Yang Digunakan.....	56
3.1.5. Analisis.....	57
BAB 3 KESIMPULAN DAN SARAN.....	61
3.1 Kesimpulan.....	61
3.2 Saran.....	62
PEMBAGIAN TUGAS TIAP ANGGOTA KELOMPOK.....	63
REFERENSI.....	64

BAB 1 DESKRIPSI PROBLEMAN

Persoalan yang harus diselesaikan dalam tugas besar ini adalah persoalan penjadwalan kuliah mingguan. Proses ini mengharuskan algoritma untuk melakukan alokasi ruangan dan waktu tertentu ke suatu mata kuliah. Proses ini juga mengharuskan program untuk memenuhi beberapa batasan untuk menentukan alokasi terbaik.

Persoalan ini dapat didefinisikan secara formal. Persoalan ini akan menerima beberapa input yaitu daftar mata kuliah, daftar ruangan, dan daftar mahasiswa. Output yang diharapkan dari program ini adalah sebuah jadwal mingguan yang valid. Setiap mata kuliah harus dialokasikan ke sebuah ruangan dan slot waktu tertentu. Jadwal yang valid juga harus memenuhi beberapa batasan. Batasan pertama adalah seorang mahasiswa tidak dapat dijadwalkan untuk mengikuti lebih dari satu mata kuliah pada waktu yang sama. Batasan kedua adalah sebuah ruangan tidak dapat digunakan oleh lebih dari 1 mata kuliah pada waktu yang sama. Terakhir, sebuah ruangan tidak bisa digunakan oleh suatu mata kuliah jika kapasitas ruangan tersebut kurang dari jumlah peserta kuliah.

Persoalan ini termasuk persoalan kategori NP-hard. Ruang pencarian solusi persoalan ini akan tumbuh secara eksponensial seiring dengan jumlah masukan. Mencari solusi optimal dengan algoritma pencarian *brute-force* akan menjadi tidak ideal. Oleh karena itu, algoritma-algoritma local search dapat menjadi pilihan yang baik untuk mencari solusi mendekati optimal dalam waktu yang wajar.

BAB 2 PEMBAHASAN

1. Pemilihan Objective Function

Objective function adalah sebuah fungsi untuk mengukur kualitas dari sebuah kandidat solusi. Tujuan pembuatan fungsi ini adalah untuk memberikan nilai kuantitatif yang dapat digunakan algoritma untuk menuju solusi yang lebih baik.

Implementasi objective function pada tugas besar ini berbasis pada penalti. Kualitas sebuah jadwal ditentukan oleh seberapa sedikit konflik yang terjadi pada jadwal tersebut. Jadwal yang terbaik adalah jadwal yang tidak memiliki konflik sama sekali. Total konflik dihitung dengan menjumlahkan konflik jadwal mahasiswa, konflik jadwal ruangan, dan konflik kapasitas. Setiap jumlah waktu yang bertabrakan untuk tiap mahasiswa akan berkontribusi 10 poin konflik, setiap mata kuliah yang dijadwalkan pada ruangan dan waktu yang sama akan berkontribusi durasi kuliah dikali dengan prioritas mata kuliah tersebut bagi mahasiswa, dan setiap kuliah yang dijadwalkan pada ruangan yang berkapasitas kurang dari peserta kuliah akan berkontribusi selisih antara peserta kuliah dan kapasitas ruangan.

2. Penjelasan Implementasi Algoritma Local Search

2.1. Algoritma Hill Climbing

2.1.1. Steepest Ascent

Varian hill climbing ini adalah varian hill climbing standar. Fungsi ini akan memeriksa semua tetangga dari jadwal saat ini. Tetangga yang dipilih sebagai suksesor adalah tetangga dengan nilai penalti terendah. Proses ini akan berlanjut hingga tidak ada lagi tetangga yang lebih baik.

```
def hill_climbing_steepest_ascent(jadwal_awal,
    kelas_mata_kuliah, ruangan, mahasiswa, hari,
    objective_function, max_neighbors=500):
    jadwal_sekarang = copy.deepcopy(jadwal_awal)
    penalti_sekarang =
    objective_function(jadwal_sekarang,
    kelas_mata_kuliah, ruangan, mahasiswa)

    history = {'iterasi': [], 'penalti': []}
    iterasi = 0

    while True:
        semua_tetangga =
        generate_all_neighbors(jadwal_sekarang, ruangan,
        hari, max_neighbors=max_neighbors)

        if not semua_tetangga:
```

```

        break

        tetangga_terbaik = None
        penalti_terbaik_tetangga = float('inf')

        for tetangga in semua_tetangga:
            penalti_tetangga =
objective_function(tetangga, kelas_mata_kuliah,
ruangan, mahasiswa)

            if penalti_tetangga <
penalti_terbaik_tetangga:
                penalti_terbaik_tetangga =
penalti_tetangga
                tetangga_terbaik = tetangga

            if penalti_terbaik_tetangga <
penalti_sekarang:
                jadwal_sekarang = tetangga_terbaik
                penalti_sekarang =
penalti_terbaik_tetangga

            history['iterasi'].append(iterasi)

history['penalti'].append(penalti_sekarang)
        iterasi += 1
    else:
        break

    return jadwal_sekarang, penalti_sekarang,
history, iterasi

```

2.1.2. Sideways Move

Varian hill climbing ini adalah varian yang memperbolehkan gerakan menyamping. Artinya, algoritma ini akan memperbolehkan perpindahan ke tetangga dengan nilai penalti sama jika tidak ada lagi tetangga yang lebih baik. Harapannya, algoritma bisa keluar dari local optima dan mencapai global optima. Gerakan menyamping akan dibatasi jumlahnya untuk menghindari perulangan tak terbatas.

```

def hill_climbing_sideways_procedure(jadwal_awal,
kelas_mata_kuliah, ruangan, mahasiswa, hari,

objective_function, max_neighbors=500,
max_sideways=100):

    jadwal_sekarang = copy.deepcopy(jadwal_awal)

```

```

    penalti_awal =
objective_function(jadwal_sekarang,
kelas_mata_kuliah, ruangan, mahasiswa)
    penalti_sekarang = penalti_awal

    history = {'iterasi': [0], 'penalti':
[penalti_awal]}
    iterasi = 0
    sideways_count = 0

    while True:
        semua_tetangga =
generate_all_neighbors(jadwal_sekarang, ruangan,
hari, max_neighbors=max_neighbors)

        if not semua_tetangga:
            break

        tetangga_terbaik = None
        penalti_terbaik_tetangga = float('inf')

        for tetangga in semua_tetangga:
            penalti_tetangga =
objective_function(tetangga, kelas_mata_kuliah,
ruangan, mahasiswa)

            if penalti_tetangga <
penalti_terbaik_tetangga:
                penalti_terbaik_tetangga =
penalti_tetangga
                tetangga_terbaik = tetangga

            if penalti_terbaik_tetangga <
penalti_sekarang:
                jadwal_sekarang = tetangga_terbaik
                penalti_sekarang =
penalti_terbaik_tetangga
                sideways_count = 0

            iterasi += 1
            history['iterasi'].append(iterasi)

        history['penalti'].append(penalti_sekarang)

        if penalti_sekarang == 0:
            break
        elif penalti_terbaik_tetangga ==
penalti_sekarang and penalti_sekarang != 0 and
sideways_count < max_sideways:
            jadwal_sekarang = tetangga_terbaik
            sideways_count += 1

```

```

        iterasi += 1
        history['iterasi'].append(iterasi)

    history['penalti'].append(penalti_sekarang)
    else:
        break

    penalti_akhir = penalti_sekarang

    return jadwal_sekarang, penalti_akhir, history,
    iterasi

```

2.1.3. Random Restart

Varian hill climbing ini adalah varian hill climbing yang menjalankan hill climbing standar berulang kali dari titik awal yang berbeda. Tujuannya adalah keluar dari local optima. Dengan memulai pencarian dari beberapa titik berbeda, diharapkan kemungkinan menemukan global optima dapat meningkat secara signifikan. Fungsi ini akan dibatasi jumlah pengulangannya dan akan menyimpan solusi terbaik yang pernah ditemukan dari semua restart.

```

def hill_climbing_random_restart_procedure(
    kelas_mata_kuliah, ruangan, mahasiswa, hari,
    objective_function, max_neighbors, max_restart
):
    jadwal_terbaik_global = None
    penalti_terbaik_global = float('inf')
    restart_stats = []
    restart_histories = []

    for restart in range(max_restart):
        print(f"\n--- RESTART {restart + 1}/{max_restart} ---")

        jadwal_restart =
generate_random_schedule(kelas_mata_kuliah,
ruangan, hari)
        penalti_restart =
objective_function(jadwal_restart,
kelas_mata_kuliah, ruangan, mahasiswa)
        print(f"Penalty awal: {penalti_restart}")

        jadwal_hasil, penalti_hasil, iterasi,
history = hill_climbing_steepest_once(
            jadwal_restart, kelas_mata_kuliah,
            ruangan, mahasiswa, hari,

```

```

        objective_function, max_neighbors
    )
    print(f"Penalty akhir: {penalti_hasil}
(iterasi: {iterasi})")

    restart_stats.append({
        'restart': restart + 1,
        'initial_penalty': penalti_restart,
        'final_penalty': penalti_hasil,
        'iterations': iterasi
    })
    restart_histories.append(history)

    if penalti_hasil < penalti_terbaik_global:
        penalti_terbaik_global = penalti_hasil
        jadwal_terbaik_global = jadwal_hasil

    if penalti_hasil == 0:
        break

    return jadwal_terbaik_global,
penalti_terbaik_global, restart_stats,
restart_histories

```

2.1.4. Stochastic

Varian ini adalah varian probabilistik dari hill climbing. Berbeda dengan hill climbing standar yang memeriksa seluruh tetangga, varian ini akan memilih tetangga secara acak pada setiap iterasi. Jika tetangga tersebut lebih baik dari solusi saat ini, maka algoritma akan pindah ke tetangga tersebut. Varian ini tidak menjamin langkah terbaik di setiap iterasi, tetapi jauh lebih ringan secara komputasi.

```

def stochastic_hill_climbing(initial_state,
kelas_mata_kuliah, ruangan, mahasiswa, hari,
max_iter=1000):
    current_state = copy.deepcopy(initial_state)
    current_penalty =
objective_function(current_state,
kelas_mata_kuliah, ruangan, mahasiswa)

    history = {'iterasi': [0], 'penalti':
[current_penalty]}
    iterasi_total = 0
    for _ in range(max_iter):
        iterasi_total += 1
        neighbor = generate_neighbor(current_state,
ruangan, hari)

```



```

        neighbor_penalty =
objective_function(neighbor, kelas_mata_kuliah,
ruangan, mahasiswa)

        if neighbor_penalty == 0:
history['iterasi'].append(iterasi_total)

history['penalti'].append(neighbor_penalty)
        print("\nInfo: Solusi optimal (penalti
= 0) ditemukan. Menghentikan pencarian lebih
awal.")
        return neighbor, neighbor_penalty,
history, iterasi_total

        if neighbor_penalty < current_penalty:
history['iterasi'].append(iterasi_total)

history['penalti'].append(neighbor_penalty)
        current_state = neighbor
        current_penalty = neighbor_penalty

        print("\nInfo: Maksimum iterasi tercapai.
Menghentikan pencarian.")
        return current_state, current_penalty, history,
history['iterasi'][-1]

```

2.2. Algoritma Simulated Annealing

Simulated Annealing (SA) adalah algoritma optimasi yang dirancang untuk menghindari terjebak di *local optimum*. Algoritma SA mirip Random-Restart Hill Climbing. Perbedaannya, alih-alih memulai ulang pencarian dari titik acak berulang kali, SA mencoba "melompat" keluar dari *local optimum* dalam satu kali proses pencarian yang berkelanjutan.

SA terinspirasi dari proses pendinginan logam (annealing). Algoritma ini dimulai dengan suhu (T) yang tinggi. Kemudian, diturunkan secara bertahap seiring waktu. Di setiap langkah, SA membuat perubahan acak kecil pada solusi saat ini untuk menghasilkan solusi "tetangga". Keputusan untuk pindah ke tetangga ini bergantung pada kualitasnya. Jika tetangga lebih baik (memiliki penalti lebih rendah), SA selalu pindah. Namun, jika tetangga lebih buruk (penalti lebih tinggi), SA masih mungkin pindah berdasarkan probabilitas $e^{\{-\Delta E/T\}}$ di mana ΔE adalah perbedaan penalti dan T adalah suhu saat ini. Probabilitas ini lebih tinggi saat suhu tinggi sehingga memungkinkan algoritma "melompat" dari solusi lokal yang kurang optimal di awal. Kemudian, menurun saat suhu mendingin sehingga membuat algoritma lebih fokus pada solusi yang lebih baik di akhir proses.

```

def simulated_annealing(jadwal_awal, suhu_awal,
laju_pendinginan, suhu_akhir, kelas_mata_kuliah,
ruangan, mahasiswa, hari):
    suhu_sekarang = suhu_awal
    jadwal_sekarang = copy.deepcopy(jadwal_awal)
    penalti_sekarang =
objective_function(jadwal_sekarang,
kelas_mata_kuliah, ruangan, mahasiswa)

    jadwal_terbaik = copy.deepcopy(jadwal_awal)
    penalti_terbaik = penalti_sekarang

    history = {
        'penalti_sekarang_per_iterasi':
[penalti_sekarang],
        'penalti_terbaik_per_iterasi':
[penalti_terbaik],
        'probabilitas_penerimaan': [],
        'periode_stagnasi': []
    }
    iterasi_tanpa_peningkatan = 0

    while suhu_sekarang > suhu_akhir:
        jadwal_tetangga =
generate_neighbor(jadwal_sekarang, ruangan, hari)
        penalti_tetangga =
objective_function(jadwal_tetangga,
kelas_mata_kuliah, ruangan, mahasiswa)
        selisih_penalti = penalti_tetangga -
penalti_sekarang

        prob_val = None
        if selisih_penalti < 0 or (selisih_penalti
>= 0 and random.random() < (prob_val :=
math.exp(-selisih_penalti / suhu_sekarang))):
            jadwal_sekarang = jadwal_tetangga
            penalti_sekarang = penalti_tetangga

    history['probabilitas_penerimaan'].append(prob_val)

    if penalti_sekarang < penalti_terbaik:
        penalti_terbaik = penalti_sekarang
        jadwal_terbaik =
copy.deepcopy(jadwal_sekarang)
        if iterasi_tanpa_peningkatan > 0:

history['periode_stagnasi'].append(iterasi_tanpa_pe
ningkatan)

```

```

        iterasi_tanpa_peningkatan = 0
    else:
        iterasi_tanpa_peningkatan += 1

    history['penalti_sekarang_per_iterasi'].append(penalti_sekarang)

    history['penalti_terbaik_per_iterasi'].append(penalti_terbaik)

    if penalti_terbaik == 0:
        print("\nInfo: Solusi optimal (penalti = 0) ditemukan.")
        break

    suhu_sekarang *= laju_pendinginan

    if iterasi_tanpa_peningkatan > 0:
        history['periode_stagnasi'].append(iterasi_tanpa_peningkatan)

    return jadwal_terbaik, penalti_terbaik, history

```

2.3. Genetic Algorithm

Algoritma genetik adalah rumpun model komputasi yang terinspirasi dari evolusi. Algoritma ini mengkodekan solusi potensial untuk masalah tertentu pada sebuah struktur data sederhana yang menyerupai kromosom dan menerapkan operator rekombinasi pada struktur tersebut. Algoritma genetik diimplementasikan dalam beberapa tahapan utama. Setiap individu dalam populasi merepresentasikan sebuah jadwal lengkap. Jadwal ini direpresentasikan dengan *list of dictionary*. Setiap *dictionary* akan berisi informasi satu kelas yaitu kode mata kuliah, ruangan, hari, waktu mulai, dan waktu selesai.

Untuk menghitung tingkat kecocokan sebuah jadwal dengan solusi, digunakan fungsi *fitness_function()*. Fungsi ini akan menghitung konflik kemudian menghitung kecocokan dengan rumus $1/(1+\text{konflik})$. Setiap jumlah waktu yang bertabrakan untuk tiap mahasiswa akan berkontribusi 10 poin konflik, setiap mata kuliah yang dijadwalkan pada ruangan dan waktu yang sama akan berkontribusi durasi kuliah dikali dengan prioritas mata kuliah tersebut bagi mahasiswa, dan setiap kuliah yang dijadwalkan pada ruangan yang berkapasitas kurang dari peserta kuliah akan berkontribusi selisih antara peserta kuliah dan kapasitas ruangan.

```

def fitness_function(population, kelas_mata_kuliah,
                    ruangan, mahasiswa):
    conflict = 0

```

[illegible]

```

        kelas_sesi = kelas
    for ruang in ruangan:
        if ruang['kode'] == sesi['ruangan']:
            ruangan_sesi = ruang
            if kelas_sesi['jumlah_mahasiswa'] >
ruangan_sesi['kuota']:
                conflict += (kelas_sesi['jumlah_mahasiswa'] -
ruangan_sesi['kuota'])
                # print(f"Capacity conflict in room
{sesi['ruangan']} for class {sesi['kode']}")
    return 1/(1+conflict)

```

Untuk melakukan pemilihan *parent*, algoritma ini menggunakan *tournament selection*. Proses ini dilakukan dengan mengambil 2 atau kandidat secara acak. Kemudian, dari kandidat-kandidat itu ditentukan kandidat terbaik. Ada 75 persen peluang bahwa kandidat terbaik tersebut akan terpilih menjadi pemenang. Sisanya, pemenang dipilih secara acak.

```

def selection(population, kelas_mata_kuliah, ruangan,
mahasiswa, num_parents):
    fitness_scores = [fitness_function(individual,
kelas_mata_kuliah, ruangan, mahasiswa) for individual in
population]
    parents = []
    index = list(range(len(population)))
    tournament_size = min(3, max(2, len(population) //
10))
    tournament_p = 0.75
    for _ in range(num_parents):
        contenders = random.sample(index,
k=min(tournament_size, len(index)))
        contenders.sort(key=lambda i: fitness_scores[i],
reverse=True)
        if random.random() < tournament_p:
            winner_idx = contenders[0]
        else:
            winner_idx = random.choice(contenders)
        parents.append(population[winner_idx])
    return parents

```

Untuk crossover dan mutation sepenuhnya dilakukan secara acak. Pada *crossover*, *crossover point* ditentukan secara acak dulu kemudian dilakukan persilangan antara 2 parent. Untuk *mutation*, bagian yang dimutasi dipilih secara acak. Setelah terpilih bagian yang akan dimutasi, nilai barunya juga ditentukan secara acak.

```

def crossover(parent1, parent2):
    point = random.randint(1, len(parent1) - 1)
    child1 = parent1[:point] + parent2[point:]
    child2 = parent2[:point] + parent1[point:]
    return child1, child2

def mutate(individu, ruangan, hari, waktu_mulai,
mutation_rate=0.1, fitness=None):
    individu_baru = []
    for gen in individu:
        gen_baru = gen.copy()
        if random.random() < mutation_rate:
            mutated = random.randint(0, 2)
            if mutated == 0:
                choice = random.choice(ruangan)
                if isinstance(choice, dict):
                    gen_baru["ruangan"] =
choice.get('kode', choice)
                else:
                    gen_baru["ruangan"] = choice
            elif mutated == 1:
                gen_baru["hari"] = random.choice(hari)
            else:
                durasi = gen_baru["waktu_selesai"] -
gen_baru["waktu_mulai"]
                waktu_mulai_baru =
random.choice(waktu_mulai)
                waktu_selesai_baru = waktu_mulai_baru +
durasi

                if waktu_selesai_baru > 18:
                    waktu_mulai_baru = 18 - durasi
                    waktu_selesai_baru = 18

                gen_baru["waktu_mulai"] =
waktu_mulai_baru
                gen_baru["waktu_selesai"] =
waktu_selesai_baru

        individu_baru.append(gen_baru)
    return individu_baru

```

Implementasi algoritma genetik ini menggunakan beberapa heuristik. Heuristik pertama adalah mutasi adaptif bergantung pada jumlah stagnasi. Semakin banyak stagnasi yang terjadi, semakin tinggi pula tingkat mutasi. Heuristik kedua adalah elitisme. Heuristik ini akan membawa 10 persen kandidat terbaik ke generasi selanjutnya. Terakhir adalah *random immigrant*. Algoritma ini memiliki kemungkinan kecil ada individu acak baru dalam setiap iterasinya.

```

def genetic_algorithm(population, kelas_mata_kuliah,
ruangan, mahasiswa, hari,waktu_mulai, iterations):
    population_size = len(population)
    max_iter = []
    avg_iter = []
    total_ind = 0
    elite_count = max(1, population_size // 10)
    mutation_rate = 0.1
    stagnan = 0
    current_best = None

    for gen in range(1, iterations + 1):
        fitness_scores = [fitness_function(individual,
kelas_mata_kuliah, ruangan, mahasiswa) for individual in
population]
        max_fitness = max(fitness_scores)
        avg_fitness = sum(fitness_scores) /
len(fitness_scores)
        max_iter.append(max_fitness)
        avg_iter.append(avg_fitness)
        best_idx = fitness_scores.index(max_fitness)
        best_individual = population[best_idx]

        if max_fitness == 1:
            return max_fitness, best_individual, gen,
max_iter, avg_iter, total_ind

        # deteksi stagnasi
        if current_best is None or max_fitness >
current_best + 1e-12:
            current_best = max_fitness
            stagnan = 0
        else:
            stagnan += 1

        # adaptive mutation
        mutation_multiplier = 1.0
        if stagnan >= 10:
            mutation_multiplier = 1.5
        if stagnan >= 30:
            mutation_multiplier = 2.5

        # elitism
        sorted_population = sorted(zip(population,
fitness_scores), key=lambda p: p[1], reverse=True)
        new_population = [p[0] for p in
sorted_population[:elite_count]]

        while len(new_population) < population_size:

```

```

        parents = selection(population,
kelas_mata_kuliah, ruangan, mahasiswa, 2)
        parent1, parent2 = parents[0], parents[1]
        child1, child2 = crossover(parent1, parent2)
        # kemungkinan kecil ada individu baru
        immigrant_rate = 0.03
        def randomize_child(child):
            child_copy = [g.copy() for g in child]
            for _ in range(max(1, len(child_copy) //
5)):
                g = random.choice(child_copy)
                rc = random.choice(ruangan)
                if isinstance(rc, dict):
                    g['ruangan'] = rc.get('kode',
g.get('ruangan'))
                else:
                    g['ruangan'] = rc
                    g['hari'] = random.choice(hari)
                    dur = g.get('waktu_selesai', 1) -
g.get('waktu_mulai', 0)
                    start = random.choice(waktu_mulai)
                    g['waktu_mulai'] = start
                    g['waktu_selesai'] = min(18, start +
max(1, dur))
            return child_copy

        if random.random() < immigrant_rate:
            child1 = randomize_child(child1)
        if random.random() < immigrant_rate:
            child2 = randomize_child(child2)

        f1 = fitness_function(child1,
kelas_mata_kuliah, ruangan, mahasiswa)
        f2 = fitness_function(child2,
kelas_mata_kuliah, ruangan, mahasiswa)
        child1 = mutate(child1, ruangan, hari,
waktu_mulai, mutation_rate * mutation_multiplier,
fitness=f1)
        child2 = mutate(child2, ruangan, hari,
waktu_mulai, mutation_rate * mutation_multiplier,
fitness=f2)
        new_population.append(child1)
        if len(new_population) < population_size:
            new_population.append(child2)
        total_ind += 2

    population = new_population[:population_size]

    fitness_scores = [fitness_function(individual,
kelas_mata_kuliah, ruangan, mahasiswa) for individual in
population]

```



```
max_fitness = max(fitness_scores)
best_individual =
population[fitness_scores.index(max_fitness)]
return max_fitness, best_individual, iterations,
max_iter, avg_iter, total_ind
```

3. Hasil Eksperimen dan Analisis

Eksperimen dilakukan dengan test case yang memiliki 30 mata kuliah, 12 ruangan, dan 50 mahasiswa.

```
{
  "kelas_mata_kuliah": [
    {"kode": "IF3071_K01", "jumlah_mahasiswa": 60, "sks": 3},
    {"kode": "IF3071_K02", "jumlah_mahasiswa": 55, "sks": 3},
    {"kode": "IF3130_K01", "jumlah_mahasiswa": 45, "sks": 2},
    {"kode": "IF3130_K02", "jumlah_mahasiswa": 50, "sks": 2},
    {"kode": "IF3110_K01", "jumlah_mahasiswa": 70, "sks": 3},
    {"kode": "IF3110_K02", "jumlah_mahasiswa": 65, "sks": 3},
    {"kode": "IF3140_K01", "jumlah_mahasiswa": 55, "sks": 2},
    {"kode": "IF3140_K02", "jumlah_mahasiswa": 48, "sks": 2},
    {"kode": "IF3150_K01", "jumlah_mahasiswa": 40, "sks": 4},
    {"kode": "IF3150_K02", "jumlah_mahasiswa": 38, "sks": 4},
    {"kode": "IF3160_K01", "jumlah_mahasiswa": 52, "sks": 3},
    {"kode": "IF3160_K02", "jumlah_mahasiswa": 48, "sks": 3},
    {"kode": "IF3170_K01", "jumlah_mahasiswa": 35, "sks": 2},
    {"kode": "IF3170_K02", "jumlah_mahasiswa": 30, "sks": 2},
    {"kode": "IF3180_K01", "jumlah_mahasiswa": 58, "sks": 3},
    {"kode": "IF3180_K02", "jumlah_mahasiswa": 62, "sks": 3},
    {"kode": "IF3190_K01", "jumlah_mahasiswa": 44, "sks": 2},
    {"kode": "IF3190_K02", "jumlah_mahasiswa": 46, "sks": 2},
    {"kode": "IF3200_K01", "jumlah_mahasiswa": 54, "sks": 3},
    {"kode": "IF3200_K02", "jumlah_mahasiswa": 56, "sks": 3},
    {"kode": "IF3210_K01", "jumlah_mahasiswa": 42, "sks": 2},
    {"kode": "IF3210_K02", "jumlah_mahasiswa": 40, "sks": 2},
    {"kode": "IF3220_K01", "jumlah_mahasiswa": 50, "sks": 4},
    {"kode": "IF3220_K02", "jumlah_mahasiswa": 48, "sks": 4},
    {"kode": "IF3230_K01", "jumlah_mahasiswa": 36, "sks": 2},
    {"kode": "IF3230_K02", "jumlah_mahasiswa": 34, "sks": 2},
    {"kode": "IF3240_K01", "jumlah_mahasiswa": 64, "sks": 3},
    {"kode": "IF3240_K02", "jumlah_mahasiswa": 60, "sks": 3},
    {"kode": "IF3250_K01", "jumlah_mahasiswa": 38, "sks": 2},
    {"kode": "IF3250_K02", "jumlah_mahasiswa": 42, "sks": 2}
  ],
  "ruangan": [
    {"kode": "7602", "kuota": 40},
```

```

    {"kode": "7603", "kuota": 50},
    {"kode": "7604", "kuota": 45},
    {"kode": "7605", "kuota": 60},
    {"kode": "7606", "kuota": 80},
    {"kode": "7607", "kuota": 55},
    {"kode": "7608", "kuota": 65},
    {"kode": "7609", "kuota": 60},
    {"kode": "7610", "kuota": 70},
    {"kode": "multimedia", "kuota": 40},
    {"kode": "lab_pemrograman", "kuota": 50},
    {"kode": "lab_jaringan", "kuota": 45}
  ],
  "mahasiswa": [
    {"nim": "13523001", "daftar_mk": ["IF3071_K01",
    "IF3130_K01", "IF3150_K01", "IF3180_K01"], "prioritas": [1, 2,
    3, 4]},
    {"nim": "13523002", "daftar_mk": ["IF3071_K02",
    "IF3130_K02", "IF3160_K01"], "prioritas": [1, 2, 3]},
    {"nim": "13523003", "daftar_mk": ["IF3110_K01",
    "IF3140_K01", "IF3170_K01"], "prioritas": [1, 2, 3]},
    {"nim": "13523004", "daftar_mk": ["IF3110_K02",
    "IF3140_K02", "IF3190_K01"], "prioritas": [1, 2, 3]},
    {"nim": "13523005", "daftar_mk": ["IF3150_K01",
    "IF3180_K01", "IF3200_K01"], "prioritas": [1, 2, 3]},
    {"nim": "13523006", "daftar_mk": ["IF3150_K02",
    "IF3180_K02", "IF3200_K02"], "prioritas": [1, 2, 3]},
    {"nim": "13523007", "daftar_mk": ["IF3160_K01",
    "IF3190_K01", "IF3210_K01"], "prioritas": [1, 2, 3]},
    {"nim": "13523008", "daftar_mk": ["IF3160_K02",
    "IF3190_K02", "IF3210_K02"], "prioritas": [1, 2, 3]},
    {"nim": "13523009", "daftar_mk": ["IF3071_K01",
    "IF3110_K01", "IF3220_K01"], "prioritas": [1, 2, 3]},
    {"nim": "13523010", "daftar_mk": ["IF3071_K02",
    "IF3110_K02", "IF3220_K02"], "prioritas": [1, 2, 3]},
    {"nim": "13523011", "daftar_mk": ["IF3130_K01",
    "IF3170_K01", "IF3230_K01"], "prioritas": [1, 2, 3]},
    {"nim": "13523012", "daftar_mk": ["IF3130_K02",
    "IF3170_K02", "IF3230_K02"], "prioritas": [1, 2, 3]},
    {"nim": "13523013", "daftar_mk": ["IF3140_K01",
    "IF3200_K01", "IF3240_K01"], "prioritas": [1, 2, 3]},
    {"nim": "13523014", "daftar_mk": ["IF3140_K02",
    "IF3200_K02", "IF3240_K02"], "prioritas": [1, 2, 3]},
    {"nim": "13523015", "daftar_mk": ["IF3150_K01",
    "IF3210_K01", "IF3250_K01"], "prioritas": [1, 2, 3]},
    {"nim": "13523016", "daftar_mk": ["IF3150_K02",
    "IF3210_K02", "IF3250_K02"], "prioritas": [1, 2, 3]},
    {"nim": "13523017", "daftar_mk": ["IF3071_K01",
    "IF3140_K01", "IF3180_K01"], "prioritas": [1, 2, 3]},
    {"nim": "13523018", "daftar_mk": ["IF3071_K02",
    "IF3140_K02", "IF3180_K02"], "prioritas": [1, 2, 3]},

```

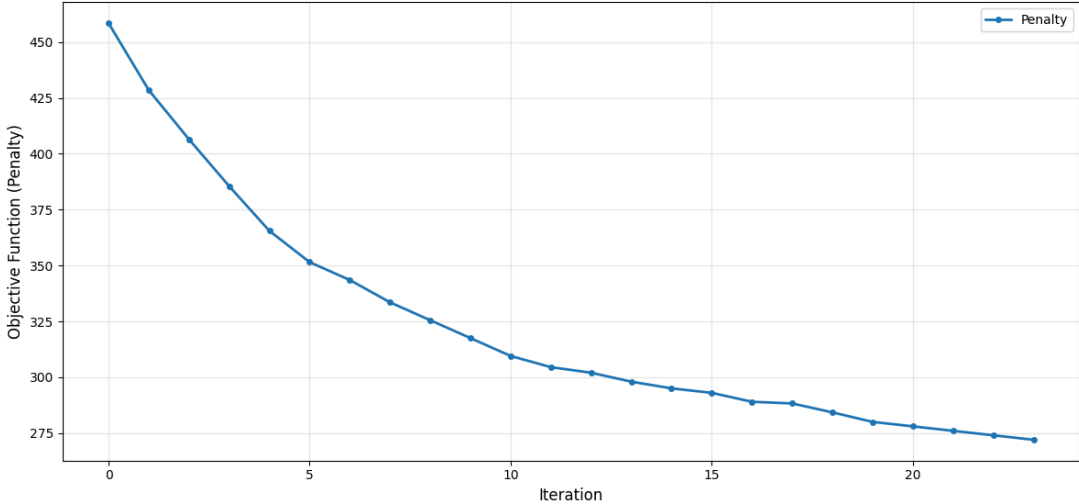
```
{
  "nim": "13523019", "daftar_mk": ["IF3110_K01",
  "IF3160_K01", "IF3200_K01"], "prioritas": [1, 2, 3]},
  {"nim": "13523020", "daftar_mk": ["IF3110_K02",
  "IF3160_K02", "IF3200_K02"], "prioritas": [1, 2, 3]},
  {"nim": "13523021", "daftar_mk": ["IF3130_K01",
  "IF3190_K01", "IF3220_K01"], "prioritas": [1, 2, 3]},
  {"nim": "13523022", "daftar_mk": ["IF3130_K02",
  "IF3190_K02", "IF3220_K02"], "prioritas": [1, 2, 3]},
  {"nim": "13523023", "daftar_mk": ["IF3170_K01",
  "IF3210_K01", "IF3240_K01"], "prioritas": [1, 2, 3]},
  {"nim": "13523024", "daftar_mk": ["IF3170_K02",
  "IF3210_K02", "IF3240_K02"], "prioritas": [1, 2, 3]},
  {"nim": "13523025", "daftar_mk": ["IF3071_K01",
  "IF3150_K01", "IF3230_K01"], "prioritas": [1, 2, 3]},
  {"nim": "13523026", "daftar_mk": ["IF3071_K02",
  "IF3150_K02", "IF3230_K02"], "prioritas": [1, 2, 3]},
  {"nim": "13523027", "daftar_mk": ["IF3110_K01",
  "IF3180_K01", "IF3250_K01"], "prioritas": [1, 2, 3]},
  {"nim": "13523028", "daftar_mk": ["IF3110_K02",
  "IF3180_K02", "IF3250_K02"], "prioritas": [1, 2, 3]},
  {"nim": "13523029", "daftar_mk": ["IF3130_K01",
  "IF3160_K01", "IF3240_K01"], "prioritas": [1, 2, 3]},
  {"nim": "13523030", "daftar_mk": ["IF3130_K02",
  "IF3160_K02", "IF3240_K02"], "prioritas": [1, 2, 3]},
  {"nim": "13523031", "daftar_mk": ["IF3071_K01",
  "IF3110_K01", "IF3130_K01", "IF3150_K01"], "prioritas": [1, 2,
  3, 4]},
  {"nim": "13523032", "daftar_mk": ["IF3071_K02",
  "IF3110_K02", "IF3130_K02", "IF3150_K02"], "prioritas": [1, 2,
  3, 4]},
  {"nim": "13523033", "daftar_mk": ["IF3140_K01",
  "IF3160_K01", "IF3180_K01"], "prioritas": [1, 2, 3]},
  {"nim": "13523034", "daftar_mk": ["IF3140_K02",
  "IF3160_K02", "IF3180_K02"], "prioritas": [1, 2, 3]},
  {"nim": "13523035", "daftar_mk": ["IF3170_K01",
  "IF3190_K01", "IF3200_K01"], "prioritas": [1, 2, 3]},
  {"nim": "13523036", "daftar_mk": ["IF3170_K02",
  "IF3190_K02", "IF3200_K02"], "prioritas": [1, 2, 3]},
  {"nim": "13523037", "daftar_mk": ["IF3210_K01",
  "IF3220_K01", "IF3230_K01"], "prioritas": [1, 2, 3]},
  {"nim": "13523038", "daftar_mk": ["IF3210_K02",
  "IF3220_K02", "IF3230_K02"], "prioritas": [1, 2, 3]},
  {"nim": "13523039", "daftar_mk": ["IF3071_K01",
  "IF3200_K01", "IF3240_K01"], "prioritas": [1, 2, 3]},
  {"nim": "13523040", "daftar_mk": ["IF3071_K02",
  "IF3200_K02", "IF3240_K02"], "prioritas": [1, 2, 3]},
  {"nim": "13523041", "daftar_mk": ["IF3110_K01",
  "IF3140_K01", "IF3250_K01"], "prioritas": [1, 2, 3]},
  {"nim": "13523042", "daftar_mk": ["IF3110_K02",
  "IF3140_K02", "IF3250_K02"], "prioritas": [1, 2, 3]},
}
```

```
    {"nim": "13523043", "daftar_mk": ["IF3130_K01",  
"IF3180_K01", "IF3220_K01"], "prioritas": [1, 2, 3]},  
    {"nim": "13523044", "daftar_mk": ["IF3130_K02",  
"IF3180_K02", "IF3220_K02"], "prioritas": [1, 2, 3]},  
    {"nim": "13523045", "daftar_mk": ["IF3150_K01",  
"IF3190_K01", "IF3230_K01"], "prioritas": [1, 2, 3]},  
    {"nim": "13523046", "daftar_mk": ["IF3150_K02",  
"IF3190_K02", "IF3230_K02"], "prioritas": [1, 2, 3]},  
    {"nim": "13523047", "daftar_mk": ["IF3160_K01",  
"IF3210_K01", "IF3240_K01"], "prioritas": [1, 2, 3]},  
    {"nim": "13523048", "daftar_mk": ["IF3160_K02",  
"IF3210_K02", "IF3240_K02"], "prioritas": [1, 2, 3]},  
    {"nim": "13523049", "daftar_mk": ["IF3170_K01",  
"IF3220_K01", "IF3250_K01"], "prioritas": [1, 2, 3]},  
    {"nim": "13523050", "daftar_mk": ["IF3170_K02",  
"IF3220_K02", "IF3250_K02"], "prioritas": [1, 2, 3]}  
  ]  
}
```

3.1.1. Hill Climbing

3.1.1.1. Steepest Ascent (maximum neighbors to sample = 500)

Iterasi	State Awal dan Akhir	
1	<pre>===== INITIAL STATE ===== Kode Ruangan Hari Jam ----- IF3071_K01 7605 Jumat 14-16 IF3071_K01 7609 Senin 17-18 IF3071_K02 lab_jaringan Rabu 9-12 IF3130_K01 7602 Jumat 7-8 IF3130_K01 7609 Selasa 13-14 IF3130_K02 7602 Selasa 7-8 IF3130_K02 7608 Selasa 8-9 IF3110_K01 multimedia Senin 8-9 IF3110_K01 multimedia Jumat 13-15 IF3110_K02 7609 Rabu 12-14 IF3110_K02 7610 Rabu 7-8 IF3140_K01 7607 Rabu 10-11 IF3140_K01 7610 Jumat 15-16 IF3140_K02 lab_pemrograman Selasa 10-11 IF3140_K02 7605 Jumat 10-11 IF3150_K01 7605 Jumat 12-13 IF3150_K01 lab_jaringan Selasa 17-18 IF3150_K01 7607 Senin 14-15 IF3150_K01 7602 Senin 15-16 IF3150_K02 lab_pemrograman Selasa 13-17 IF3160_K01 7605 Kamis 11-12 IF3160_K01 lab_jaringan Rabu 13-15 IF3160_K02 7610 Kamis 13-15 IF3160_K02 7604 Selasa 16-17 IF3170_K01 7608 Selasa 8-9 IF3170_K01 multimedia Rabu 17-18 IF3170_K02 lab_pemrograman Jumat 15-16 IF3170_K02 multimedia Selasa 7-8 IF3180_K01 7606 Rabu 12-14 IF3180_K01 7606 Senin 14-15 IF3180_K02 7602 Jumat 11-14 IF3190_K01 7603 Kamis 13-14 IF3190_K01 lab_jaringan Jumat 17-18 IF3190_K02 7604 Senin 8-10 IF3200_K01 7608 Selasa 10-13 IF3200_K02 multimedia Jumat 14-17 IF3210_K01 7610 Rabu 15-17 IF3210_K02 7608 Kamis 14-16 IF3220_K01 7603 Kamis 16-17 IF3220_K01 7603 Rabu 12-15 IF3220_K02 lab_jaringan Kamis 12-15 IF3220_K02 7608 Senin 16-17 IF3230_K01 7608 Rabu 9-11 IF3230_K02 7604 Jumat 13-15 IF3240_K01 7606 Senin 15-16 IF3240_K01 7608 Kamis 9-11 IF3240_K02 multimedia Rabu 13-16 IF3250_K01 7603 Rabu 14-15 IF3250_K01 7608 Rabu 9-10 IF3250_K02 lab_jaringan Kamis 9-11 ===== Initial Objective Function: 471.25</pre>	<pre>===== FINAL STATE ===== Schedule has conflicts - displaying in list: ===== SCHEDULE ===== Kode Ruangan Hari Jam ----- IF3071_K01 7605 Jumat 14-16 IF3071_K01 7609 Senin 17-18 IF3071_K02 lab_jaringan Rabu 9-12 IF3130_K01 lab_pemrograman Selasa 10-11 IF3130_K01 7604 Selasa 16-17 IF3130_K02 7607 Senin 14-15 IF3130_K02 lab_pemrograman Jumat 15-16 IF3110_K01 7606 Senin 14-15 IF3110_K01 7610 Rabu 15-17 IF3110_K02 7606 Rabu 12-14 IF3110_K02 7610 Rabu 7-8 IF3140_K01 7607 Rabu 10-11 IF3140_K01 7610 Jumat 15-16 IF3140_K02 7605 Jumat 12-13 IF3140_K02 7605 Jumat 10-11 IF3150_K01 7602 Jumat 7-8 IF3150_K01 lab_jaringan Selasa 17-18 IF3150_K01 7602 Selasa 7-8 IF3150_K01 7602 Senin 15-16 IF3150_K02 lab_pemrograman Selasa 13-17 IF3160_K01 7605 Kamis 11-12 IF3160_K01 7609 Rabu 12-14 IF3160_K02 7610 Kamis 13-15 IF3160_K02 7609 Selasa 13-14 IF3170_K01 multimedia Senin 8-9 IF3170_K01 multimedia Rabu 17-18 IF3170_K02 7606 Senin 15-16 IF3170_K02 multimedia Selasa 7-8 IF3180_K01 7608 Kamis 14-16 IF3180_K01 7608 Senin 16-17 IF3180_K02 7608 Selasa 10-13 IF3190_K01 7603 Kamis 13-14 IF3190_K01 lab_jaringan Jumat 17-18 IF3190_K02 lab_jaringan Rabu 13-15 IF3200_K01 7602 Jumat 11-14 IF3200_K02 multimedia Jumat 14-17 IF3210_K01 7604 Jumat 13-15 IF3210_K02 7604 Senin 8-10 IF3220_K01 7603 Kamis 16-17 IF3220_K01 7603 Rabu 12-15 IF3220_K02 lab_jaringan Kamis 12-15 IF3220_K02 7608 Selasa 8-9 IF3230_K01 7608 Rabu 9-11 IF3230_K02 multimedia Jumat 13-15 IF3240_K01 7608 Selasa 8-9 IF3240_K01 7608 Kamis 9-11 IF3240_K02 multimedia Rabu 13-16 IF3250_K01 7603 Rabu 14-15 IF3250_K01 7608 Rabu 9-10 IF3250_K02 lab_jaringan Kamis 9-11 =====</pre>
	<pre>===== RESULTS ===== Final Objective Function: 278.5 Number of Iterations: 16 Duration: 4.0206 seconds =====</pre>	
	Visualisasi	

	<div>=====</div> <div>RESULTS</div> <div>=====</div> <div>Final Objective Function: 272.0</div> <div>Number of Iterations: 24</div> <div>Duration: 6.5387 seconds</div> <div>=====</div>																																																			
	Visualisasi																																																			
	<div>Steepest Ascent Hill Climbing - Objective Function</div>  <table border="1"><thead><tr><th>Iteration</th><th>Objective Function (Penalty)</th></tr></thead><tbody><tr><td>0</td><td>455</td></tr><tr><td>1</td><td>428</td></tr><tr><td>2</td><td>405</td></tr><tr><td>3</td><td>385</td></tr><tr><td>4</td><td>365</td></tr><tr><td>5</td><td>352</td></tr><tr><td>6</td><td>342</td></tr><tr><td>7</td><td>332</td></tr><tr><td>8</td><td>325</td></tr><tr><td>9</td><td>318</td></tr><tr><td>10</td><td>310</td></tr><tr><td>11</td><td>305</td></tr><tr><td>12</td><td>302</td></tr><tr><td>13</td><td>298</td></tr><tr><td>14</td><td>295</td></tr><tr><td>15</td><td>292</td></tr><tr><td>16</td><td>288</td></tr><tr><td>17</td><td>288</td></tr><tr><td>18</td><td>285</td></tr><tr><td>19</td><td>280</td></tr><tr><td>20</td><td>278</td></tr><tr><td>21</td><td>276</td></tr><tr><td>22</td><td>275</td></tr><tr><td>23</td><td>274</td></tr><tr><td>24</td><td>272</td></tr></tbody></table>	Iteration	Objective Function (Penalty)	0	455	1	428	2	405	3	385	4	365	5	352	6	342	7	332	8	325	9	318	10	310	11	305	12	302	13	298	14	295	15	292	16	288	17	288	18	285	19	280	20	278	21	276	22	275	23	274	24
Iteration	Objective Function (Penalty)																																																			
0	455																																																			
1	428																																																			
2	405																																																			
3	385																																																			
4	365																																																			
5	352																																																			
6	342																																																			
7	332																																																			
8	325																																																			
9	318																																																			
10	310																																																			
11	305																																																			
12	302																																																			
13	298																																																			
14	295																																																			
15	292																																																			
16	288																																																			
17	288																																																			
18	285																																																			
19	280																																																			
20	278																																																			
21	276																																																			
22	275																																																			
23	274																																																			
24	272																																																			
3	State Awal dan Akhir																																																			

INITIAL STATE			
Kode	Ruangan	Hari	Jam
IF3071_K01	7605	Senin	10-12
IF3071_K01	7602	Selasa	8-9
IF3071_K02	lab_pemrograman	Jumat	13-16
IF3130_K01	7609	Senin	15-16
IF3130_K01	7602	Senin	9-10
IF3130_K02	7604	Rabu	15-17
IF3110_K01	lab_jaringan	Senin	13-16
IF3110_K02	lab_pemrograman	Jumat	9-11
IF3110_K02	7604	Selasa	11-12
IF3140_K01	7606	Rabu	8-9
IF3140_K01	lab_pemrograman	Senin	8-9
IF3140_K02	7606	Jumat	16-17
IF3140_K02	7609	Kamis	7-8
IF3150_K01	7608	Jumat	17-18
IF3150_K01	7609	Kamis	12-15
IF3150_K02	7602	Kamis	15-16
IF3150_K02	lab_pemrograman	Senin	7-8
IF3150_K02	7610	Senin	12-14
IF3160_K01	multimedia	Senin	8-9
IF3160_K01	7607	Rabu	15-17
IF3160_K02	lab_jaringan	Jumat	8-9
IF3160_K02	7606	Jumat	17-18
IF3160_K02	7603	Kamis	14-15
IF3170_K01	7608	Senin	16-17
IF3170_K01	7602	Kamis	15-16
IF3170_K02	7606	Rabu	16-18
IF3180_K01	7606	Selasa	11-14
IF3180_K02	lab_pemrograman	Senin	13-15
IF3180_K02	lab_pemrograman	Rabu	9-10
IF3190_K01	7607	Jumat	17-18
IF3190_K01	7609	Kamis	11-12
IF3190_K02	7610	Senin	11-12
IF3190_K02	7606	Kamis	14-15
IF3200_K01	7607	Kamis	11-13
IF3200_K01	lab_jaringan	Selasa	12-13
IF3200_K02	7602	Senin	11-12
IF3200_K02	multimedia	Selasa	16-18
IF3210_K01	7608	Rabu	10-12
IF3210_K02	lab_pemrograman	Selasa	13-14
IF3210_K02	7604	Selasa	12-13
IF3220_K01	7610	Jumat	14-16
IF3220_K01	lab_pemrograman	Senin	9-11
IF3220_K02	lab_pemrograman	Jumat	11-15
IF3230_K01	7610	Selasa	10-11
IF3230_K01	multimedia	Kamis	7-8
IF3230_K02	lab_jaringan	Rabu	14-15
IF3240_K01	7610	Senin	11-12
IF3240_K01	multimedia	Jumat	12-13
IF3240_K01	7609	Senin	14-16
IF3240_K02	7608	Rabu	8-11
IF3250_K01	lab_pemrograman	Jumat	14-15
IF3250_K01	7603	Rabu	12-13
IF3250_K02	7602	Jumat	7-9

Initial Objective Function: 569.75

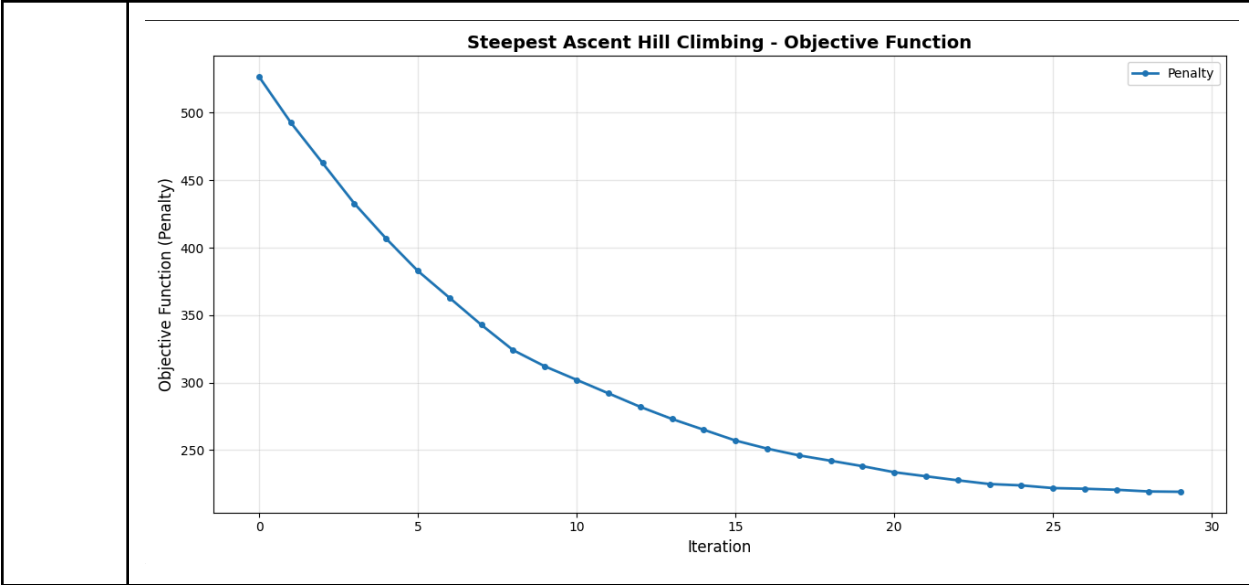
Schedule has conflicts - displaying in list:

SCHEDULE			
Kode	Ruangan	Hari	Jam
IF3071_K01	7605	Senin	10-12
IF3071_K01	7608	Senin	16-17
IF3071_K02	7609	Kamis	12-15
IF3130_K01	lab_pemrograman	Rabu	9-10
IF3130_K01	lab_jaringan	Jumat	8-9
IF3130_K02	lab_pemrograman	Jumat	9-11
IF3110_K01	7606	Selasa	11-14
IF3110_K02	7610	Senin	12-14
IF3140_K01	7609	Senin	15-16
IF3140_K01	7606	Kamis	14-15
IF3140_K02	7603	Rabu	12-13
IF3140_K02	lab_pemrograman	Senin	8-9
IF3150_K01	multimedia	Jumat	12-13
IF3150_K01	lab_jaringan	Senin	13-16
IF3150_K02	7604	Selasa	11-12
IF3150_K02	multimedia	Senin	8-9
IF3150_K02	7602	Jumat	7-9
IF3160_K01	7606	Rabu	8-9
IF3160_K01	7607	Rabu	15-17
IF3160_K02	lab_pemrograman	Selasa	13-14
IF3160_K02	7606	Jumat	17-18
IF3160_K02	7603	Kamis	14-15
IF3170_K01	7602	Selasa	8-9
IF3170_K01	7602	Senin	11-12
IF3170_K02	multimedia	Selasa	16-18
IF3180_K01	lab_pemrograman	Jumat	13-16
IF3180_K02	7606	Rabu	16-18
IF3180_K02	7606	Jumat	16-17
IF3190_K01	7607	Jumat	17-18
IF3190_K01	lab_jaringan	Selasa	12-13
IF3190_K02	lab_pemrograman	Senin	7-8
IF3190_K02	7609	Kamis	11-12
IF3200_K01	7607	Kamis	11-13
IF3200_K01	7609	Kamis	7-8
IF3200_K02	7610	Senin	11-12
IF3200_K02	7609	Senin	14-16
IF3210_K01	7608	Rabu	10-12
IF3210_K02	lab_jaringan	Rabu	14-15
IF3210_K02	7604	Selasa	12-13
IF3220_K01	lab_pemrograman	Senin	13-15
IF3220_K01	lab_pemrograman	Senin	9-11
IF3220_K02	lab_pemrograman	Jumat	11-15
IF3230_K01	7602	Kamis	15-16
IF3230_K01	multimedia	Kamis	7-8
IF3230_K02	7602	Senin	9-10
IF3230_K02	7610	Senin	11-12
IF3240_K01	7608	Jumat	17-18
IF3240_K01	7610	Jumat	14-16
IF3240_K02	7608	Rabu	8-11
IF3250_K01	lab_pemrograman	Jumat	14-15
IF3250_K01	7602	Kamis	15-16
IF3250_K02	7604	Rabu	15-17

RESULTS

Final Objective Function: 219.0
Number of Iterations: 30
Duration: 8.7985 seconds

Visualisasi



3.1.1.2. Sideways Move (maximum neighbors to sample = 500, maximum sideways moves = 100)

Iterasi	State Awal dan Akhir
---------	----------------------

INITIAL STATE			
Kode	Ruangan	Hari	Jam
IF3071_K01	lab_pemrograman	Jumat	7-9
IF3071_K01	7602	Jumat	9-10
IF3071_K02	7603	Senin	7-10
IF3130_K01	7609	Kamis	11-13
IF3130_K02	7607	Senin	15-16
IF3130_K02	multimedia	Rabu	12-13
IF3110_K01	7607	Jumat	12-13
IF3110_K01	7607	Rabu	8-10
IF3110_K02	7606	Rabu	8-11
IF3140_K01	lab_pemrograman	Kamis	16-18
IF3140_K02	7604	Senin	10-11
IF3140_K02	7605	Senin	17-18
IF3150_K01	7604	Kamis	13-17
IF3150_K02	7605	Rabu	14-16
IF3150_K02	lab_pemrograman	Selasa	14-16
IF3160_K01	lab_jaringan	Kamis	12-15
IF3160_K02	7608	Kamis	7-9
IF3160_K02	7610	Selasa	12-13
IF3170_K01	7604	Rabu	16-18
IF3170_K02	lab_jaringan	Kamis	8-10
IF3180_K01	7608	Senin	9-12
IF3180_K02	multimedia	Jumat	15-18
IF3190_K01	7604	Senin	12-14
IF3190_K02	lab_pemrograman	Rabu	8-9
IF3190_K02	7610	Selasa	11-12
IF3200_K01	7603	Senin	14-17
IF3200_K02	7608	Kamis	15-17
IF3200_K02	7607	Selasa	9-10
IF3210_K01	7608	Senin	15-16
IF3210_K01	multimedia	Kamis	11-12
IF3210_K02	7605	Jumat	7-8
IF3210_K02	7605	Senin	8-9
IF3220_K01	7610	Senin	14-15
IF3220_K01	lab_jaringan	Selasa	12-15
IF3220_K02	7602	Selasa	12-13
IF3220_K02	lab_pemrograman	Jumat	9-11
IF3220_K02	7610	Selasa	10-11
IF3230_K01	multimedia	Senin	8-9
IF3230_K01	7608	Kamis	17-18
IF3230_K02	7608	Selasa	14-15
IF3230_K02	7604	Selasa	10-11
IF3240_K01	7604	Rabu	14-15
IF3240_K01	7604	Rabu	11-12
IF3240_K01	lab_jaringan	Jumat	7-8
IF3240_K02	7605	Rabu	9-11
IF3240_K02	7603	Rabu	8-9
IF3250_K01	7606	Jumat	13-14
IF3250_K01	7610	Rabu	7-8
IF3250_K02	7607	Selasa	7-8
IF3250_K02	7606	Rabu	8-9

Initial Objective Function: 364.0

FINAL STATE			
Schedule has conflicts - displaying in list:			
SCHEDULE			
Kode	Ruangan	Hari	Jam
IF3071_K01	7609	Kamis	11-13
IF3071_K01	7606	Jumat	13-14
IF3071_K02	lab_jaringan	Kamis	12-15
IF3130_K01	lab_pemrograman	Jumat	7-9
IF3130_K02	7607	Senin	15-16
IF3130_K02	7607	Selasa	9-10
IF3110_K01	7610	Selasa	10-11
IF3110_K01	7608	Kamis	7-9
IF3110_K02	7608	Senin	9-12
IF3140_K01	7607	Rabu	8-10
IF3140_K02	7608	Senin	15-16
IF3140_K02	7605	Senin	17-18
IF3150_K01	7604	Kamis	13-17
IF3150_K02	7605	Rabu	14-16
IF3150_K02	lab_pemrograman	Selasa	14-16
IF3160_K01	7603	Senin	7-10
IF3160_K02	lab_pemrograman	Jumat	9-11
IF3160_K02	7610	Selasa	12-13
IF3170_K01	7604	Rabu	16-18
IF3170_K02	lab_jaringan	Kamis	8-10
IF3180_K01	multimedia	Jumat	15-18
IF3180_K02	7606	Rabu	8-11
IF3190_K01	7604	Senin	12-14
IF3190_K02	lab_pemrograman	Rabu	8-9
IF3190_K02	7610	Selasa	11-12
IF3200_K01	7603	Senin	14-17
IF3200_K02	7608	Kamis	15-17
IF3200_K02	7605	Jumat	7-8
IF3210_K01	7604	Senin	10-11
IF3210_K01	7604	Rabu	11-12
IF3210_K02	multimedia	Rabu	12-13
IF3210_K02	7605	Senin	8-9
IF3220_K01	7610	Senin	14-15
IF3220_K01	lab_jaringan	Selasa	12-15
IF3220_K02	7607	Selasa	7-8
IF3220_K02	lab_pemrograman	Kamis	16-18
IF3220_K02	7607	Jumat	12-13
IF3230_K01	multimedia	Senin	8-9
IF3230_K01	7603	Rabu	8-9
IF3230_K02	multimedia	Kamis	11-12
IF3230_K02	7602	Selasa	12-13
IF3240_K01	7610	Rabu	7-8
IF3240_K01	7608	Selasa	14-15
IF3240_K01	7608	Kamis	17-18
IF3240_K02	7605	Rabu	9-11
IF3240_K02	7606	Rabu	8-9
IF3250_K01	7602	Jumat	9-10
IF3250_K01	7604	Rabu	14-15
IF3250_K02	7604	Selasa	10-11
IF3250_K02	lab_jaringan	Jumat	7-8

RESULTS

Final Objective Function: 154.25
Number of Iterations: 120
Maximum Sideways Moves: 100
Duration: 27.0151 seconds

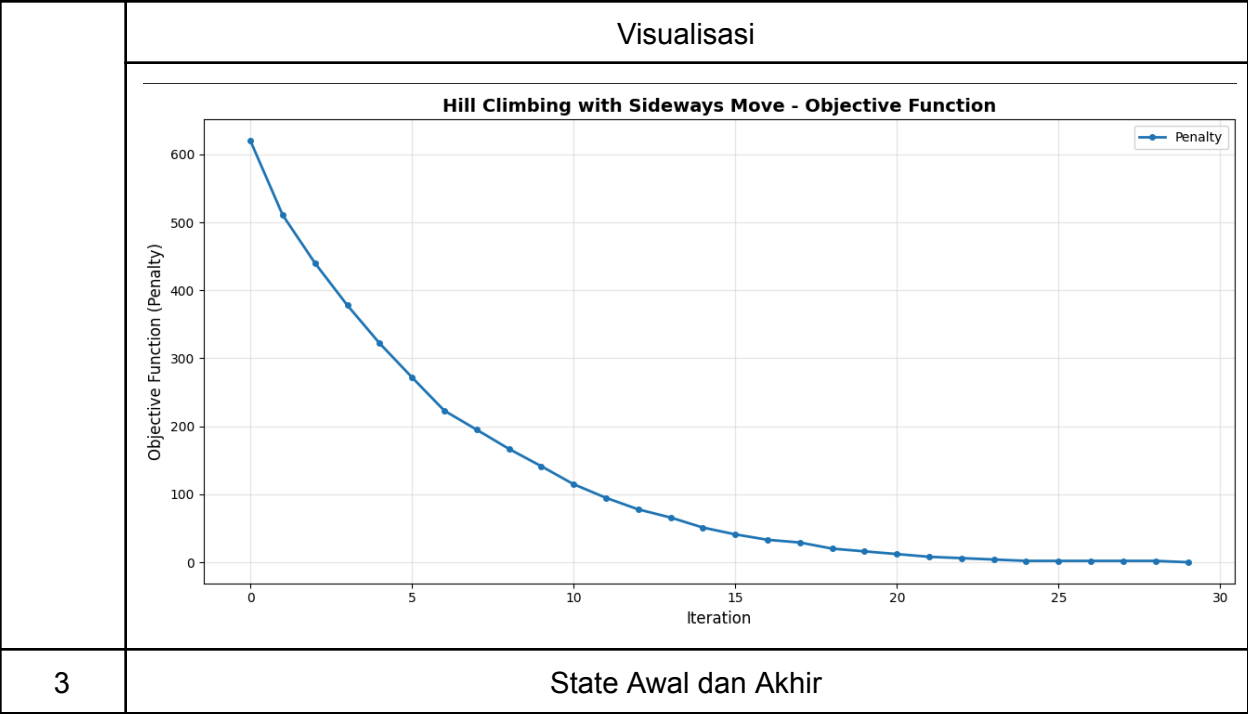
Visualisasi

JADWAL RUANGAN: 7605					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8	IF3071_K02				
9	IF3071_K02		IF3160_K01		
10	IF3071_K02				
11			IF3190_K02		
12					
13		IF3190_K02			
14					
15					IF3240_K02
16					
17			IF3130_K02		
18					
=====					
JADWAL RUANGAN: 7606					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					IF3160_K01
8					IF3160_K01
9					
10	IF3110_K01		IF3140_K01		
11	IF3110_K01		IF3140_K01		
12					
13					
14					
15					
16		IF3180_K01		IF3240_K02	
17		IF3180_K01	IF3240_K01	IF3240_K02	
18					
=====					
JADWAL RUANGAN: 7607					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8					
9					
10					
11					
12					
13					
14					
15				IF3200_K01	
16				IF3200_K01	
17					
18					
=====					

JADWAL RUANGAN: 7608					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7				IF3180_K02	IF3180_K02
8					IF3180_K02
9					
10					IF3110_K02
11					IF3110_K02
12					IF3110_K02
13					
14	IF3240_K01				
15				IF3220_K02	
16				IF3220_K02	
17					IF3220_K01
18					
=====					
JADWAL RUANGAN: 7609					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7		IF3071_K01			
8	IF3160_K02	IF3071_K01			IF3220_K01
9	IF3160_K02	IF3071_K01			IF3220_K01
10	IF3160_K02				IF3220_K01
11					
12					
13		IF3200_K01			
14					
15			IF3200_K02		
16			IF3200_K02		
17			IF3200_K02		
18					
=====					
JADWAL RUANGAN: 7610					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8			IF3240_K01		
9					
10					
11					
12					IF3110_K01
13					
14					
15					
16	IF3180_K01				
17					
18					
=====					

JADWAL RUANGAN: multimedia					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7		IF3230_K02			
8	IF3150_K01				
9					
10					
11	IF3230_K02				
12					
13					
14					
15		IF3230_K01			
16	IF3150_K02				
17					IF3210_K02
18					
=====					
JADWAL RUANGAN: lab_pemrograman					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					IF3170_K02
8					IF3170_K02
9					
10				IF3220_K02	
11				IF3220_K02	
12					
13					
14					
15					
16					
17					
18					
=====					
JADWAL RUANGAN: lab_jaringan					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8					
9		IF3190_K01			
10		IF3190_K01			
11					
12					
13					IF3130_K01
14		IF3150_K02			IF3130_K01
15		IF3150_K02			
16		IF3150_K02			
17					
18					
=====					

RESULTS					
=====					
Final Objective Function: 0					
Number of Iterations: 29					
Maximum Sideways Moves: 100					
Duration: 6.5100 seconds					
=====					



=====

INITIAL STATE

=====

Kode	Ruangan	Hari	Jam
IF3071_K01	lab_pemrograman	Selasa	14-15
IF3071_K01	7604	Jumat	8-10
IF3071_K02	7606	Rabu	14-17
IF3130_K01	multimedia	Rabu	14-15
IF3130_K01	7602	Senin	15-16
IF3130_K02	7607	Rabu	13-14
IF3130_K02	7608	Senin	12-13
IF3110_K01	multimedia	Selasa	14-15
IF3110_K01	multimedia	Senin	11-13
IF3110_K02	7602	Senin	13-16
IF3140_K01	7606	Selasa	16-18
IF3140_K02	7608	Rabu	14-15
IF3140_K02	7610	Selasa	12-13
IF3150_K01	lab_jaringan	Jumat	13-15
IF3150_K01	multimedia	Senin	17-18
IF3150_K01	lab_jaringan	Selasa	13-14
IF3150_K02	7608	Selasa	17-18
IF3150_K02	7602	Kamis	14-16
IF3150_K02	7608	Kamis	12-13
IF3160_K01	lab_pemrograman	Senin	12-15
IF3160_K02	7603	Rabu	7-9
IF3160_K02	7605	Senin	12-13
IF3170_K01	7605	Jumat	13-15
IF3170_K02	7606	Selasa	15-17
IF3180_K01	7602	Rabu	12-15
IF3180_K02	7609	Kamis	10-12
IF3180_K02	7602	Selasa	16-17
IF3190_K01	7602	Kamis	9-10
IF3190_K01	lab_jaringan	Rabu	12-13
IF3190_K02	multimedia	Jumat	9-11
IF3200_K01	lab_jaringan	Rabu	13-16
IF3200_K02	multimedia	Kamis	14-15
IF3200_K02	multimedia	Selasa	11-13
IF3210_K01	7610	Jumat	12-13
IF3210_K01	7602	Rabu	14-15
IF3210_K02	7608	Selasa	12-14
IF3220_K01	7605	Rabu	9-12
IF3220_K01	7604	Senin	10-11
IF3220_K02	7606	Selasa	10-12
IF3220_K02	7607	Kamis	11-13
IF3230_K01	7602	Jumat	15-16
IF3230_K01	lab_jaringan	Rabu	16-17
IF3230_K02	7608	Senin	15-16
IF3230_K02	lab_pemrograman	Senin	11-12
IF3240_K01	7609	Kamis	7-8
IF3240_K01	lab_jaringan	Senin	17-18
IF3240_K01	lab_pemrograman	Jumat	10-11
IF3240_K02	7602	Jumat	7-8
IF3240_K02	lab_jaringan	Selasa	14-15
IF3240_K02	7604	Kamis	14-15
IF3250_K01	7606	Senin	8-9
IF3250_K01	7608	Selasa	16-17
IF3250_K02	7608	Rabu	10-12

=====

Initial Objective Function: 616.75

=====

FINAL STATE

=====

Schedule has conflicts - displaying in list:

=====

SCHEDULE

=====

Kode	Ruangan	Hari	Jam
IF3071_K01	7605	Senin	12-13
IF3071_K01	7609	Kamis	10-12
IF3071_K02	7605	Rabu	9-12
IF3130_K01	lab_jaringan	Rabu	12-13
IF3130_K01	7604	Kamis	14-15
IF3130_K02	7607	Rabu	13-14
IF3130_K02	lab_pemrograman	Selasa	14-15
IF3110_K01	7606	Senin	8-9
IF3110_K01	7606	Selasa	10-12
IF3110_K02	7606	Rabu	14-17
IF3140_K01	7608	Rabu	10-12
IF3140_K02	lab_jaringan	Selasa	14-15
IF3140_K02	lab_pemrograman	Jumat	10-11
IF3150_K01	7604	Jumat	8-10
IF3150_K01	multimedia	Senin	17-18
IF3150_K01	7602	Kamis	9-10
IF3150_K02	7602	Jumat	15-16
IF3150_K02	7602	Kamis	14-16
IF3150_K02	7602	Jumat	7-8
IF3160_K01	lab_pemrograman	Senin	12-15
IF3160_K02	7603	Rabu	7-9
IF3160_K02	7609	Kamis	7-8
IF3170_K01	multimedia	Jumat	9-11
IF3170_K02	multimedia	Senin	11-13
IF3180_K01	lab_jaringan	Rabu	13-16
IF3180_K02	7608	Selasa	12-14
IF3180_K02	7608	Senin	12-13
IF3190_K01	lab_jaringan	Selasa	13-14
IF3190_K01	7604	Senin	10-11
IF3190_K02	7606	Selasa	15-17
IF3200_K01	7602	Rabu	12-15
IF3200_K02	7610	Jumat	12-13
IF3200_K02	7605	Jumat	13-15
IF3210_K01	multimedia	Kamis	14-15
IF3210_K01	7602	Rabu	14-15
IF3210_K02	multimedia	Selasa	11-13
IF3220_K01	7602	Senin	13-16
IF3220_K01	lab_pemrograman	Senin	11-12
IF3220_K02	7606	Selasa	16-18
IF3220_K02	7607	Kamis	11-13
IF3230_K01	7602	Selasa	16-17
IF3230_K01	lab_jaringan	Rabu	16-17
IF3230_K02	lab_jaringan	Senin	17-18
IF3230_K02	multimedia	Rabu	14-15
IF3240_K01	7608	Senin	15-16
IF3240_K01	7608	Selasa	17-18
IF3240_K01	7610	Selasa	12-13
IF3240_K02	7608	Kamis	12-13
IF3240_K02	7608	Rabu	14-15
IF3240_K02	7608	Selasa	16-17
IF3250_K01	multimedia	Selasa	14-15
IF3250_K01	7602	Senin	15-16
IF3250_K02	lab_jaringan	Jumat	13-15

=====

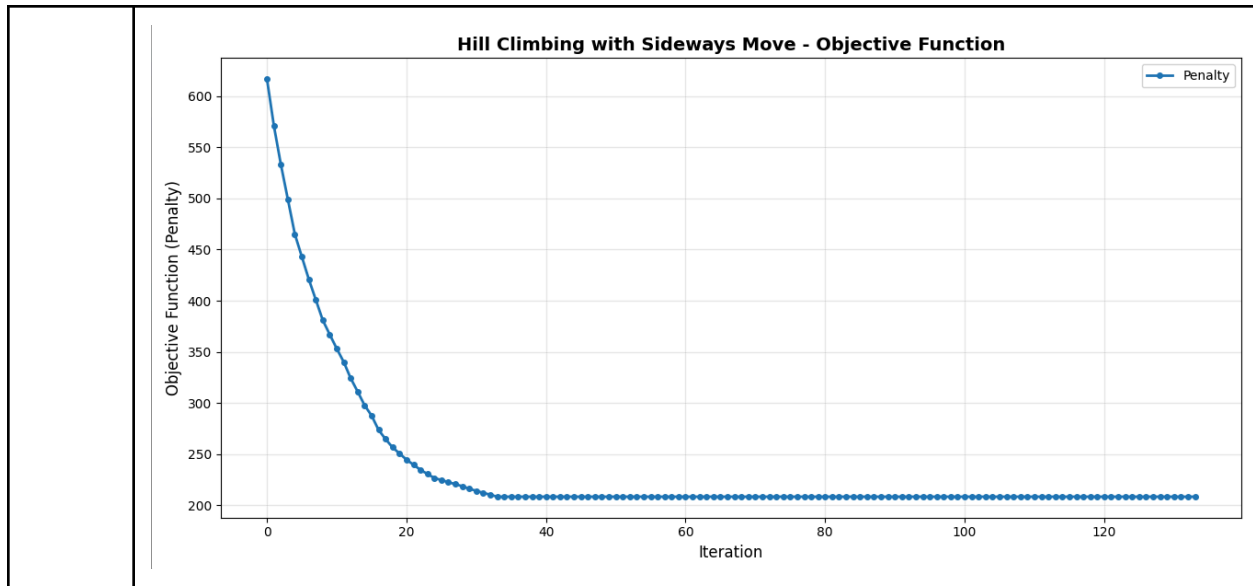
Final Objective Function: 208.5

Number of Iterations: 133

Maximum Sideways Moves: 100

Duration: 36.5575 seconds

Visualisasi



3.1.1.3. Random Restart (maximum neighbors to sample = 500, maximum restarts = 10)

Iterasi	State Awal dan Akhir	
1	<pre> --- RESTART 9/10 --- Initial State: ===== INITIAL STATE ===== Kode Ruangan Hari Jam ----- IF3071_K01 7608 Jumat 7-9 IF3071_K01 7605 Senin 9-10 IF3071_K02 lab_pemrograman Selasa 7-9 IF3071_K02 7608 Selasa 13-14 IF3130_K01 7608 Jumat 11-12 IF3130_K01 7607 Rabu 12-13 IF3130_K02 7602 Senin 13-15 IF3110_K01 7604 Rabu 15-18 IF3110_K02 7603 Kamis 9-10 IF3110_K02 7605 Senin 8-9 IF3110_K02 lab_jaringan Senin 12-13 IF3140_K01 lab_pemrograman Kamis 8-10 IF3140_K02 7610 Selasa 10-12 IF3150_K01 7610 Kamis 13-17 IF3150_K02 7603 Kamis 7-8 IF3150_K02 7607 Kamis 13-14 IF3150_K02 7610 Kamis 16-18 IF3160_K01 7603 Kamis 7-10 IF3160_K02 7609 Jumat 12-15 IF3170_K01 7603 Rabu 11-13 IF3170_K02 lab_jaringan Rabu 11-12 IF3170_K02 7604 Kamis 11-12 IF3180_K01 7605 Kamis 8-11 IF3180_K02 lab_jaringan Rabu 13-16 IF3190_K01 7602 Senin 16-17 IF3190_K01 multimedia Senin 9-10 IF3190_K02 7609 Kamis 9-11 IF3200_K01 lab_pemrograman Senin 15-17 IF3200_K01 7607 Kamis 9-10 IF3200_K02 7607 Selasa 7-9 IF3200_K02 lab_pemrograman Selasa 16-17 IF3210_K01 7607 Rabu 13-14 IF3210_K01 7607 Jumat 8-9 IF3210_K02 7603 Jumat 13-15 IF3220_K01 7609 Rabu 12-16 IF3220_K02 7610 Selasa 7-9 IF3220_K02 7607 Jumat 16-18 IF3230_K01 7603 Jumat 14-15 IF3230_K01 lab_pemrograman Senin 15-16 IF3230_K02 7607 Rabu 10-12 IF3240_K01 lab_pemrograman Kamis 9-10 IF3240_K01 7606 Kamis 10-12 IF3240_K02 7604 Selasa 14-17 IF3250_K01 lab_pemrograman Kamis 7-9 IF3250_K02 7603 Senin 7-9 ===== Initial Objective Function: 566.5 Penalty awal: 566.5 Penalty akhir: 0 (iterasi: 26) </pre>	<pre> ===== FINAL STATE ===== JADWAL RUANGAN: 7602 ===== Jam Senin Selasa Rabu Kamis Jumat ----- 7 8 9 10 11 12 13 IF3230_K02 14 IF3230_K02 15 16 IF3150_K02 17 18 ===== JADWAL RUANGAN: 7603 ===== Jam Senin Selasa Rabu Kamis Jumat ----- 7 IF3250_K02 8 IF3250_K02 9 10 11 12 13 14 IF3190_K01 15 16 17 18 ===== JADWAL RUANGAN: 7604 ===== Jam Senin Selasa Rabu Kamis Jumat ----- 7 8 IF3150_K02 9 IF3150_K02 10 11 IF3190_K01 12 13 14 15 16 17 18 ===== </pre>

JADWAL RUANGAN: 7605					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8	IF3200_K02			IF3180_K01	
9	IF3071_K01			IF3180_K01	
10				IF3180_K01	
11					
12					
13					
14					
15					
16					
17					
18					
=====					
JADWAL RUANGAN: 7606					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8					
9					
10				IF3240_K01	
11				IF3240_K01	
12					
13					
14				IF3240_K02	
15				IF3240_K02	
16				IF3240_K02	IF3110_K02
17					
18					
=====					
JADWAL RUANGAN: 7607					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7		IF3140_K02			
8		IF3140_K02			IF3210_K01
9				IF3210_K01	
10			IF3071_K02		
11			IF3071_K02		
12			IF3071_K02		
13			IF3200_K01	IF3150_K02	IF3160_K02
14					IF3160_K02
15					IF3160_K02
16					IF3250_K01
17					IF3250_K01
18					
=====					

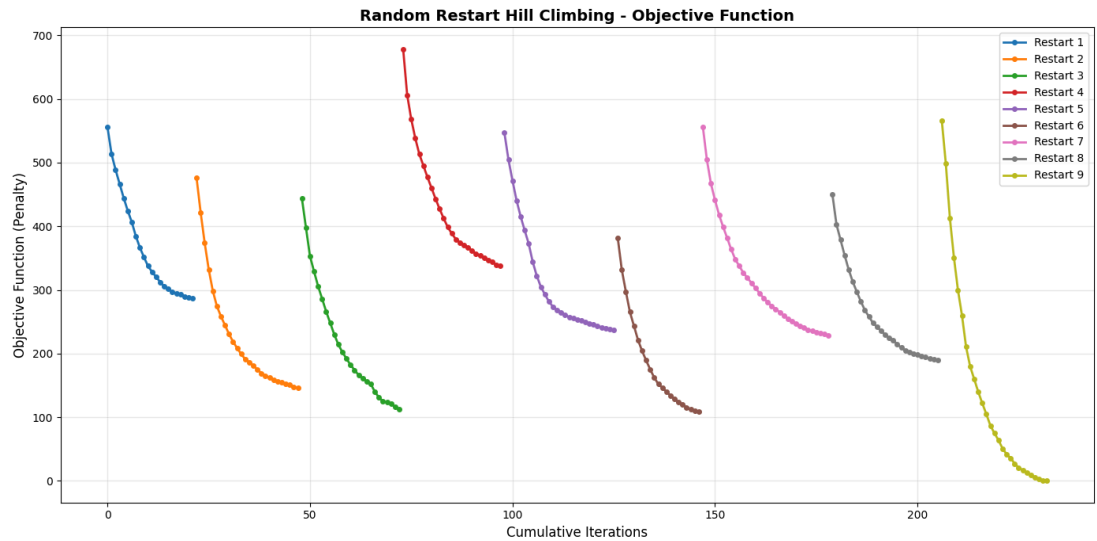
JADWAL RUANGAN: 7608					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					IF3071_K01
8					IF3071_K01
9					
10					
11	IF3180_K02				IF3110_K02
12	IF3180_K02				
13	IF3180_K02	IF3110_K02			
14					IF3220_K02
15					IF3220_K02
16					
17	IF3240_K01				
18					
=====					
JADWAL RUANGAN: 7609					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8					
9				IF3190_K02	
10				IF3190_K02	
11					
12			IF3220_K01		IF3160_K01
13			IF3220_K01		IF3160_K01
14	IF3140_K01		IF3220_K01		IF3160_K01
15	IF3140_K01		IF3220_K01		
16					
17					
18					
=====					
JADWAL RUANGAN: 7610					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7		IF3200_K01			
8		IF3200_K01			
9					
10		IF3200_K02			
11		IF3200_K02			
12					
13		IF3110_K01		IF3150_K01	
14		IF3110_K01		IF3150_K01	
15		IF3110_K01		IF3150_K01	
16				IF3150_K01	
17					
18					
=====					

JADWAL RUANGAN: multimedia					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8					
9	IF3170_K02				
10					
11					
12					
13					
14					
15					
16					
17					
18					
=====					
JADWAL RUANGAN: lab_pemrograman					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7		IF3130_K02		IF3220_K02	
8		IF3130_K02		IF3220_K02	
9					
10					
11					
12					
13					
14					
15	IF3230_K01				
16		IF3130_K01	IF3170_K01		
17			IF3170_K01		
18					
=====					
JADWAL RUANGAN: lab_jaringan					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8					IF3210_K02
9					IF3210_K02
10					
11					
12	IF3130_K01		IF3170_K02		
13					
14					
15					
16					
17					
18					
=====					

RESULTS			
=====			
Final Objective Function: 0			
Number of Restarts: 9			
Per-Restart Statistics:			
Restart	Initial	Final	Iterations

1	555.5	287.5	21
2	475.75	146.25	25
3	444.0	113.25	24
4	678.5	338.25	24
5	547.0	237.0	27
6	301.25	100.75	20
7	556.5	220.5	31
8	449.75	189.75	26
9	566.5	0	26
Total Duration: 75.4641 seconds			
=====			

Visualisasi



2

State Awal dan Akhir

--- RESTART 10/10 ---

Initial State:

INITIAL STATE			
Kode	Ruangan	Hari	Jam
IF3071_K01	lab_jaringan	Jumat	12-15
IF3071_K02	lab_jaringan	Jumat	9-12
IF3130_K01	7607	Selasa	10-12
IF3130_K02	7605	Jumat	13-14
IF3110_K01	7606	Rabu	16-17
IF3110_K01	7606	Rabu	8-10
IF3110_K01	7605	Rabu	9-10
IF3110_K02	7606	Jumat	12-15
IF3140_K01	7609	Kamis	14-16
IF3140_K02	multimedia	Rabu	13-14
IF3140_K02	7604	Selasa	14-15
IF3150_K01	lab_jaringan	Senin	7-8
IF3150_K01	7604	Senin	13-16
IF3150_K02	7605	Senin	7-8
IF3150_K02	7603	Jumat	14-17
IF3160_K01	7610	Kamis	8-11
IF3160_K02	7606	Senin	10-11
IF3160_K02	7602	Jumat	7-9
IF3170_K01	7607	Selasa	12-13
IF3170_K01	lab_jaringan	Selasa	9-10
IF3170_K02	7608	Kamis	7-9
IF3180_K01	7606	Rabu	12-15
IF3180_K02	7607	Rabu	15-17
IF3180_K02	multimedia	Jumat	12-13
IF3190_K01	7604	Kamis	13-14
IF3190_K01	7603	Kamis	17-18
IF3190_K02	7603	Rabu	7-9
IF3200_K01	7602	Selasa	11-12
IF3200_K01	multimedia	Kamis	11-12
IF3200_K01	multimedia	Jumat	10-11
IF3200_K02	lab_jaringan	Rabu	11-13
IF3200_K02	7607	Kamis	16-17
IF3210_K01	7602	Jumat	10-12
IF3210_K02	7603	Selasa	14-15
IF3210_K02	7604	Kamis	9-10
IF3220_K01	7605	Jumat	12-13
IF3220_K01	7606	Rabu	15-18
IF3220_K02	7607	Senin	14-15
IF3220_K02	7602	Kamis	10-12
IF3220_K02	7609	Jumat	10-11
IF3230_K01	multimedia	Rabu	8-9
IF3230_K01	7608	Senin	15-16
IF3230_K02	lab_jaringan	Selasa	7-8
IF3230_K02	lab_pemrograman	Rabu	9-10
IF3240_K01	7609	Senin	8-10
IF3240_K01	lab_pemrograman	Selasa	9-10
IF3240_K02	7606	Kamis	11-13
IF3240_K02	7610	Kamis	7-8
IF3250_K01	7603	Kamis	10-12
IF3250_K02	7602	Jumat	10-11
IF3250_K02	lab_pemrograman	Selasa	10-11

Initial Objective Function: 341.0

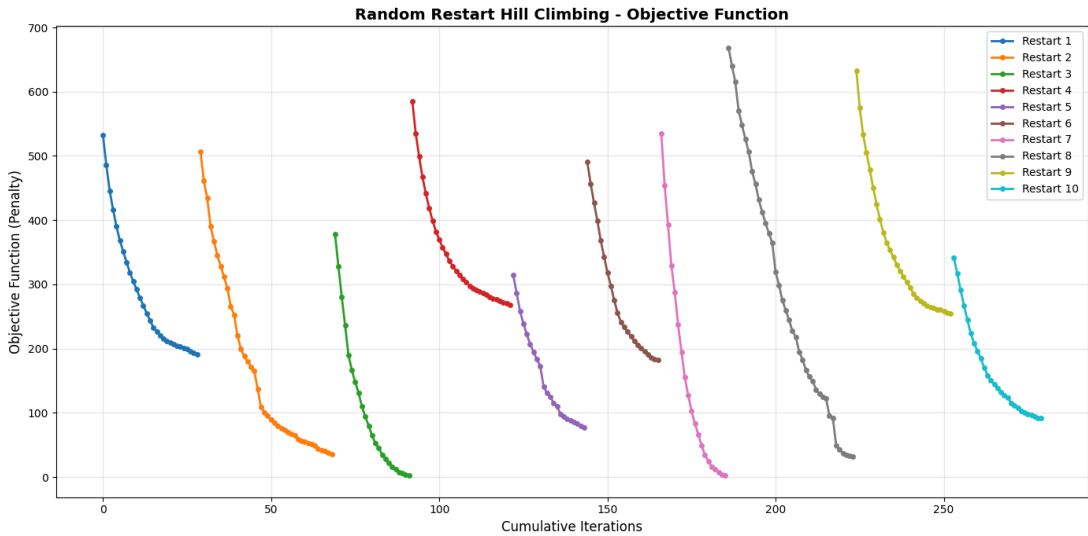
FINAL STATE

Schedule has conflicts - displaying in list:

SCHEDULE			
Kode	Ruangan	Hari	Jam
IF3071_K01	7605	Kamis	10-11
IF3071_K01	7608	Senin	15-16
IF3071_K01	7605	Selasa	12-13
IF3071_K02	7609	Selasa	13-16
IF3130_K01	7607	Jumat	16-18
IF3130_K02	7609	Kamis	11-13
IF3110_K01	7606	Senin	12-14
IF3110_K01	7610	Rabu	10-11
IF3110_K02	7610	Selasa	17-18
IF3110_K02	7610	Rabu	7-8
IF3110_K02	7606	Selasa	8-9
IF3140_K01	7609	Rabu	12-14
IF3140_K02	7608	Kamis	12-14
IF3150_K01	lab_pemrograman	Jumat	11-15
IF3150_K02	7604	Kamis	13-17
IF3160_K01	7605	Jumat	13-16
IF3160_K02	lab_pemrograman	Rabu	9-11
IF3160_K02	7603	Selasa	11-12
IF3170_K01	multimedia	Selasa	12-13
IF3170_K01	7602	Selasa	11-12
IF3170_K02	7604	Selasa	8-10
IF3180_K01	7610	Selasa	8-9
IF3180_K01	7608	Jumat	9-10
IF3180_K01	7606	Selasa	14-15
IF3180_K02	7606	Senin	10-12
IF3180_K02	7610	Senin	15-16
IF3190_K01	lab_pemrograman	Jumat	9-10
IF3190_K01	lab_pemrograman	Senin	16-17
IF3190_K02	7607	Rabu	12-14
IF3200_K01	7609	Kamis	7-9
IF3200_K01	7605	Rabu	15-16
IF3200_K02	7605	Kamis	7-8
IF3200_K02	7610	Jumat	9-11
IF3210_K01	7607	Kamis	12-13
IF3210_K01	7603	Kamis	14-15
IF3210_K02	7602	Selasa	15-17
IF3220_K01	7603	Kamis	8-9
IF3220_K01	7610	Senin	11-14
IF3220_K02	7609	Jumat	12-15
IF3220_K02	lab_pemrograman	Selasa	11-12
IF3230_K01	7609	Selasa	16-18
IF3230_K02	lab_jaringan	Jumat	7-9
IF3240_K01	7608	Selasa	15-16
IF3240_K01	7608	Senin	12-14
IF3240_K02	7610	Rabu	12-14
IF3240_K02	7605	Senin	11-12
IF3250_K01	7602	Senin	7-9
IF3250_K02	lab_jaringan	Senin	12-14

RESULTS			
Final Objective Function: 2			
Number of Restarts: 10			
Per-Restart Statistics:			
Restart	Initial	Final	Iterations
1	532.75	191.0	28
2	507.0	36	39
3	378.5	2	22
4	584.5	268.5	29
5	314.25	77.25	21
6	490.75	182.0	21
7	534.25	2	19
8	668.75	32	37
9	633.25	254.25	28
10	341.0	91.5	26
Total Duration: 71.2547 seconds			

Visualisasi



--- RESTART 9/10 ---

Initial State:

INITIAL STATE

Kode	Ruangan	Hari	Jam
IF3071_K01	lab_pemrograman	Rabu	8-9
IF3071_K01	7606	Jumat	9-10
IF3071_K01	7610	Jumat	7-8
IF3071_K02	7606	Selasa	7-8
IF3071_K02	7604	Jumat	8-10
IF3130_K01	7605	Selasa	9-11
IF3130_K02	lab_jaringan	Jumat	9-11
IF3110_K01	7606	Jumat	11-12
IF3110_K01	7605	Selasa	10-11
IF3110_K01	lab_jaringan	Kamis	8-9
IF3110_K02	7610	Kamis	9-12
IF3140_K01	7604	Senin	8-10
IF3140_K02	7610	Kamis	13-15
IF3150_K01	7608	Selasa	13-15
IF3150_K01	7608	Kamis	12-14
IF3150_K02	7606	Rabu	16-18
IF3150_K02	7605	Rabu	11-13
IF3160_K01	7602	Selasa	16-17
IF3160_K01	7604	Kamis	11-13
IF3160_K02	7608	Kamis	16-18
IF3160_K02	7606	Senin	16-17
IF3170_K01	7605	Kamis	13-14
IF3170_K01	7602	Kamis	14-15
IF3170_K02	7608	Selasa	12-14
IF3180_K01	7608	Kamis	15-18
IF3180_K02	7606	Senin	11-14
IF3190_K01	7608	Senin	11-12
IF3190_K01	7610	Selasa	7-8
IF3190_K02	7608	Rabu	10-11
IF3190_K02	7607	Jumat	13-14
IF3200_K01	7607	Rabu	13-16
IF3200_K02	lab_pemrograman	Senin	15-16
IF3200_K02	7609	Selasa	7-9
IF3210_K01	lab_jaringan	Jumat	16-18
IF3210_K02	7606	Jumat	12-13
IF3210_K02	7607	Rabu	9-10
IF3220_K01	7602	Selasa	7-8
IF3220_K01	7604	Selasa	14-17
IF3220_K02	7608	Rabu	14-18
IF3230_K01	lab_pemrograman	Jumat	16-17
IF3230_K01	lab_jaringan	Jumat	11-12
IF3230_K02	7604	Jumat	10-12
IF3240_K01	7609	Selasa	10-13
IF3240_K02	7605	Rabu	13-14
IF3240_K02	lab_jaringan	Kamis	16-18
IF3250_K01	7608	Senin	11-12
IF3250_K01	7608	Selasa	17-18
IF3250_K02	lab_jaringan	Kamis	8-9
IF3250_K02	7609	Senin	11-12

Initial Objective Function: 344.25
Penalty awal: 344.25
Penalty akhir: 0 (iterasi: 20)

FINAL STATE

JADWAL RUANGAN: 7602

Jam	Senin	Selasa	Rabu	Kamis	Jumat
7		IF3210_K02			
8					IF3250_K01
9					
10					
11					
12					
13					
14				IF3170_K01	
15					
16		IF3210_K02			
17					
18					

JADWAL RUANGAN: 7603

Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8					
9					
10					
11					
12					
13					
14					
15					
16					
17					
18					

JADWAL RUANGAN: 7604

Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8	IF3150_K02				IF3150_K02
9	IF3150_K02				IF3150_K02
10					IF3230_K02
11				IF3150_K01	
12				IF3150_K01	
13					
14					
15					
16					
17					
18					

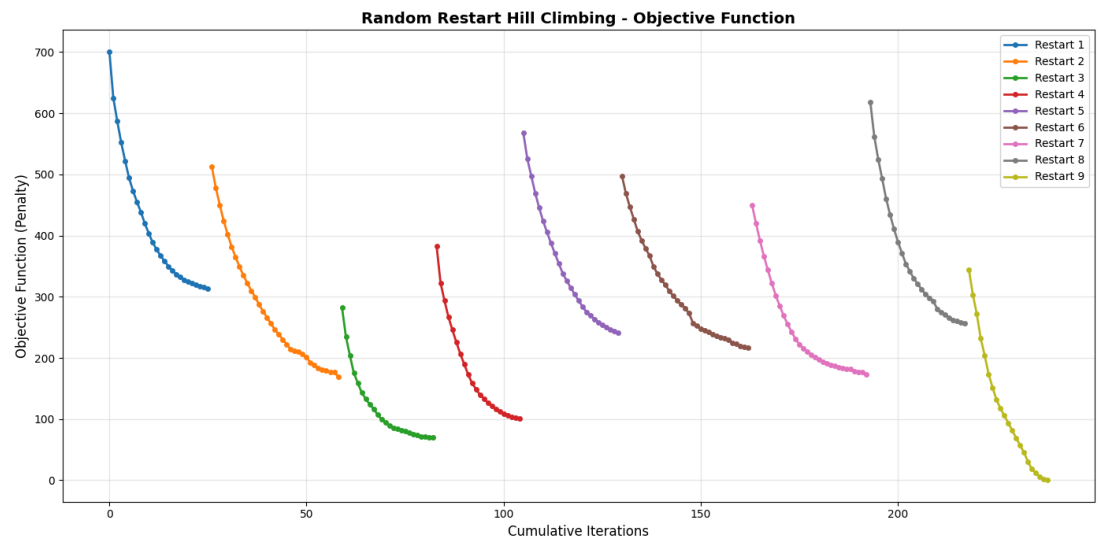
JADWAL RUANGAN: 7605					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8					
9		IF3240_K02			
10		IF3240_K02			
11			IF3071_K02		
12			IF3071_K02		
13			IF3240_K02	IF3170_K01	
14					
15					
16					
17					
18					
JADWAL RUANGAN: 7606					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7	IF3071_K02				
8					
9					IF3071_K01
10					
11	IF3180_K02				IF3110_K01
12	IF3180_K02				IF3160_K01
13	IF3180_K02				
14					
15					
16	IF3200_K02		IF3140_K01		
17			IF3140_K01		
18					
JADWAL RUANGAN: 7607					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8					
9			IF3220_K01		
10					
11					
12					
13			IF3200_K01		
14			IF3200_K01		IF3190_K02
15			IF3200_K01		
16					
17					
18					

JADWAL RUANGAN: 7608					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8					
9					
10			IF3190_K02		
11	IF3190_K01				
12		IF3130_K02		IF3160_K01	
13		IF3130_K02		IF3160_K01	
14			IF3220_K02		
15			IF3220_K02		
16			IF3220_K02	IF3160_K02	
17		IF3250_K01	IF3220_K02	IF3160_K02	
18					
JADWAL RUANGAN: 7609					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7		IF3200_K02			
8		IF3200_K02			
9					
10		IF3220_K01			
11	IF3071_K01	IF3220_K01			
12		IF3220_K01			
13					
14		IF3180_K01			
15		IF3180_K01			
16		IF3180_K01			
17					
18					
JADWAL RUANGAN: 7610					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7		IF3110_K01			IF3071_K01
8					
9				IF3110_K02	
10		IF3240_K01		IF3110_K02	
11		IF3240_K01		IF3110_K02	
12		IF3240_K01			
13	IF3190_K01			IF3140_K02	
14	IF3110_K01			IF3140_K02	
15					
16					
17					
18					

JADWAL RUANGAN: multimedia					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8					
9					
10					
11					
12					
13					
14					
15					
16					
17					
18					
JADWAL RUANGAN: lab_pemrograman					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8			IF3250_K02		
9					
10					
11					
12					
13					
14					
15	IF3160_K02				
16					IF3230_K01
17					
18					
JADWAL RUANGAN: lab_jaringan					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8				IF3250_K02	
9					IF3210_K01
10					IF3210_K01
11					IF3230_K01
12					
13					
14					
15	IF3150_K01				
16	IF3150_K01			IF3130_K01	IF3170_K02
17				IF3130_K01	IF3170_K02
18					

RESULTS			
Final Objective Function: 0			
Number of Restarts: 9			
Per-Restart Statistics:			
Restart	Initial	Final	Iterations
1	701.0	313.5	25
2	512.0	169.5	32
3	282.5	69.75	23
4	382.5	100.75	21
5	567.5	241.5	24
6	497.75	216.25	32
7	450.0	173.0	29
8	617.75	256.25	24
9	344.25	0	20
Total Duration: 65.3014 seconds			

Visualisasi



3.1.1.4. Stochastic (maximum iterations = 5000)

Iterasi	State Awal dan Akhir
1	<div> <pre> ===== INITIAL STATE ===== Kode Ruangn Hari Jam ----- IF3071_K01 7606 Jumat 10-11 IF3071_K01 multimedia Jumat 11-12 IF3071_K01 7605 Kamis 17-18 IF3071_K02 7606 Kamis 10-12 IF3071_K02 7605 Kamis 17-18 IF3130_K01 7610 Senin 11-13 IF3130_K02 7603 Kamis 10-12 IF3110_K01 7606 Jumat 12-14 IF3110_K01 7604 Kamis 10-11 IF3110_K02 7610 Senin 15-16 IF3110_K02 7609 Kamis 10-12 IF3140_K01 7607 Selasa 14-15 IF3140_K01 multimedia Rabu 12-13 IF3140_K02 7608 Rabu 11-13 IF3150_K01 7606 Senin 8-9 IF3150_K01 7610 Senin 13-16 IF3150_K02 7609 Senin 9-13 IF3160_K01 7604 Rabu 11-13 IF3160_K01 7609 Jumat 15-16 IF3160_K02 7602 Rabu 12-14 IF3160_K02 lab_jaringan Jumat 10-11 IF3170_K01 7608 Kamis 15-17 IF3170_K02 7608 Senin 11-13 IF3180_K01 7610 Senin 11-13 IF3180_K01 multimedia Kamis 13-14 IF3180_K02 7603 Senin 10-11 IF3180_K02 7607 Kamis 13-15 IF3190_K01 7605 Selasa 7-9 IF3190_K02 7604 Senin 14-15 IF3190_K02 lab_jaringan Selasa 10-11 IF3200_K01 lab_jaringan Kamis 16-17 IF3200_K01 7603 Rabu 7-8 IF3200_K01 lab_jaringan Senin 8-9 IF3200_K02 multimedia Senin 9-11 IF3200_K02 7605 Kamis 13-14 IF3210_K01 7602 Senin 14-15 IF3210_K01 7606 Jumat 7-8 IF3210_K02 7602 Kamis 11-12 IF3210_K02 7606 Kamis 11-12 IF3220_K01 7605 Senin 7-9 IF3220_K01 7603 Senin 10-12 IF3220_K02 7608 Senin 8-9 IF3220_K02 7608 Senin 7-10 IF3230_K01 7605 Rabu 12-14 IF3230_K02 7602 Kamis 8-10 IF3240_K01 7603 Kamis 7-10 IF3240_K02 7603 Selasa 10-12 IF3240_K02 7609 Senin 8-9 IF3250_K01 lab_jaringan Kamis 11-12 IF3250_K01 7610 Kamis 7-8 IF3250_K02 7609 Senin 15-17 ===== Initial Objective Function: 498.5 </pre> </div> <div> <pre> ===== FINAL STATE ===== JADWAL RUANGAN: 7602 ===== Jam Senin Selasa Rabu Kamis Jumat ----- 7 8 9 10 11 12 13 14 15 16 17 18 ===== JADWAL RUANGAN: 7603 ===== Jam Senin Selasa Rabu Kamis Jumat ----- 7 8 9 10 IF3170_K01 IF3160_K02 IF3220_K01 11 IF3170_K01 IF3160_K02 IF3220_K01 12 13 14 IF3230_K01 15 IF3230_K01 16 17 IF3160_K02 18 ===== JADWAL RUANGAN: 7604 ===== Jam Senin Selasa Rabu Kamis Jumat ----- 7 8 9 10 11 12 13 14 IF3210_K02 15 16 17 IF3210_K01 18 ===== </pre> </div>

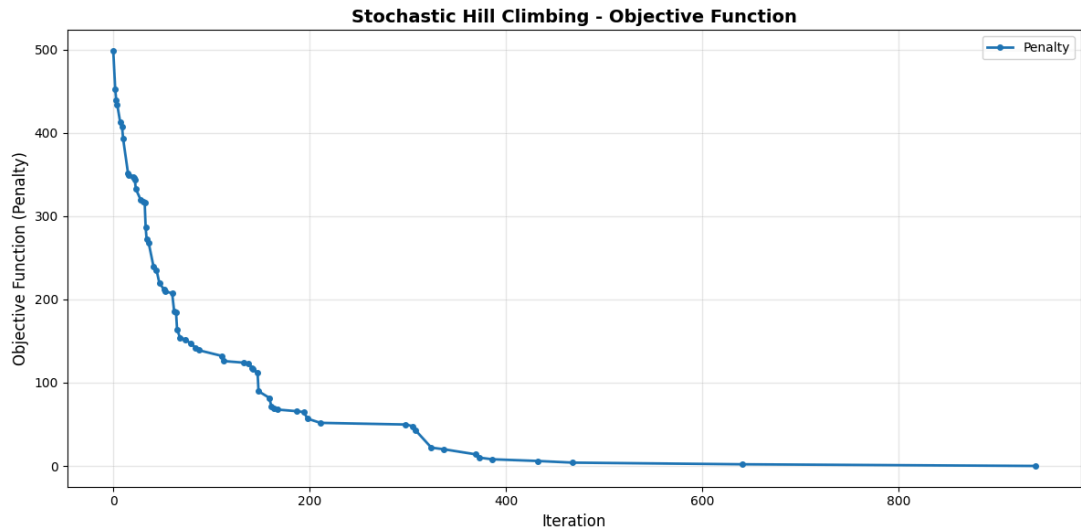
JADWAL RUANGAN: 7695					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7	IF3220_K01	IF3190_K01			IF3200_K01
8	IF3220_K01	IF3190_K01			
9					
10					
11					
12					
13				IF3240_K02	
14	IF3200_K02			IF3071_K01	
15	IF3200_K02				
16					IF3071_K01
17					
18					
=====					
JADWAL RUANGAN: 7696					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					IF3110_K02
8	IF3140_K01				
9				IF3200_K01	
10				IF3071_K02	IF3071_K01
11			IF3110_K02	IF3071_K02	
12			IF3110_K02		IF3110_K01
13					IF3110_K01
14					
15					
16					
17					
18					
=====					
JADWAL RUANGAN: 7697					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8					
9					
10					
11					
12			IF3190_K02		
13				IF3160_K01	
14		IF3140_K01		IF3160_K01	
15				IF3220_K02	
16		IF3210_K01		IF3220_K02	
17				IF3220_K02	
18					
=====					

JADWAL RUANGAN: 7698					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7	IF3190_K02				
8					
9					
10					
11	IF3180_K02		IF3130_K01		
12	IF3180_K02		IF3130_K01		
13					
14					
15				IF3140_K02	
16				IF3140_K02	
17					
18					
=====					
JADWAL RUANGAN: 7699					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7		IF3250_K02			
8	IF3200_K02	IF3250_K02			
9					
10				IF3170_K02	
11				IF3170_K02	
12					
13					
14					
15					IF3200_K01
16					
17					
18					
=====					
JADWAL RUANGAN: 7610					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7				IF3250_K01	IF3110_K01
8					
9		IF3180_K01			
10					
11	IF3180_K01				
12	IF3180_K01			IF3180_K02	IF3240_K02
13	IF3240_K01				IF3240_K02
14	IF3240_K01				
15	IF3240_K01	IF3160_K01			
16					
17				IF3071_K02	
18					
=====					

JADWAL RUANGAN: multimedia					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8					
9					
10					
11					
12		IF3150_K01	IF3250_K01		
13		IF3150_K01	IF3210_K02		
14		IF3150_K01			
15					
16					
17					
18					
=====					
JADWAL RUANGAN: lab_pemrograman					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8	IF3130_K02				
9	IF3130_K02		IF3220_K02		
10					
11					
12					
13					
14					
15					
16					
17					
18					
=====					
JADWAL RUANGAN: lab_jaringan					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8					
9					
10					
11				IF3150_K01	
12					
13					
14					
15					
16					
17					
18					
=====					

RESULTS					
=====					
Final Objective Function: 0					
Number of Iterations: 939					
Duration: 0.4327 seconds					
=====					

Visualisasi



2

State Awal dan Akhir

INITIAL STATE			
Kode	Ruangan	Hari	Jam
IF3071_K01	7606	Senin	11-12
IF3071_K01	multimedia	Rabu	17-18
IF3071_K01	7603	Selasa	11-12
IF3071_K02	7607	Kamis	11-12
IF3071_K02	7607	Kamis	12-13
IF3071_K02	7610	Senin	12-13
IF3130_K01	7610	Kamis	15-16
IF3130_K01	multimedia	Selasa	16-17
IF3130_K02	7610	Selasa	9-10
IF3130_K02	7610	Jumat	16-17
IF3110_K01	7606	Jumat	13-16
IF3110_K02	7602	Rabu	7-10
IF3140_K01	lab_pemrograman	Jumat	12-13
IF3140_K01	lab_pemrograman	Kamis	17-18
IF3140_K02	7604	Jumat	13-15
IF3150_K01	7602	Rabu	12-13
IF3150_K01	lab_pemrograman	Rabu	12-13
IF3150_K01	7602	Jumat	13-15
IF3150_K02	lab_pemrograman	Senin	9-13
IF3160_K01	7609	Senin	16-17
IF3160_K01	7607	Senin	9-11
IF3160_K02	7603	Selasa	14-17
IF3170_K01	multimedia	Kamis	11-12
IF3170_K01	7608	Selasa	12-13
IF3170_K02	7606	Senin	14-16
IF3180_K01	lab_jaringan	Rabu	8-11
IF3180_K02	7602	Jumat	16-17
IF3180_K02	lab_pemrograman	Selasa	17-18
IF3180_K02	7609	Selasa	11-12
IF3190_K01	lab_jaringan	Rabu	15-16
IF3190_K01	7610	Jumat	10-11
IF3190_K02	7609	Rabu	13-15
IF3200_K01	7604	Selasa	8-10
IF3200_K01	multimedia	Kamis	11-12
IF3200_K02	7604	Selasa	9-10
IF3200_K02	7610	Rabu	9-10
IF3200_K02	multimedia	Selasa	8-9
IF3210_K01	7606	Kamis	15-16
IF3210_K01	7603	Senin	7-8
IF3210_K02	7605	Selasa	16-18
IF3220_K01	lab_pemrograman	Kamis	15-16
IF3220_K01	7605	Selasa	14-17
IF3220_K02	lab_jaringan	Kamis	15-17
IF3220_K02	7602	Jumat	16-17
IF3220_K02	7605	Jumat	17-18
IF3230_K01	7606	Selasa	9-10
IF3230_K01	7608	Kamis	16-17
IF3230_K02	7603	Jumat	11-13
IF3240_K01	7603	Kamis	16-18
IF3240_K01	7607	Kamis	7-8
IF3240_K02	7606	Jumat	11-12
IF3240_K02	7608	Kamis	12-13
IF3240_K02	7606	Kamis	10-11
IF3250_K01	multimedia	Senin	12-13
IF3250_K01	lab_pemrograman	Selasa	14-15
IF3250_K02	7605	Rabu	7-9

Initial Objective Function: 449.0

FINAL STATE					
JADWAL RUANGAN: 7602					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8					
9					
10					
11					
12			IF3150_K01		
13					IF3170_K02
14					IF3170_K02
15					
16					IF3230_K01
17					
18					
JADWAL RUANGAN: 7603					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7	IF3210_K01				
8	IF3220_K02				
9	IF3220_K02				
10					
11					
12					
13					
14					
15					
16				IF3150_K01	IF3220_K02
17				IF3150_K01	
18					
JADWAL RUANGAN: 7604					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8		IF3250_K02			
9		IF3250_K02			
10					
11					
12					
13					
14		IF3210_K02			
15		IF3210_K02			
16					
17					
18					

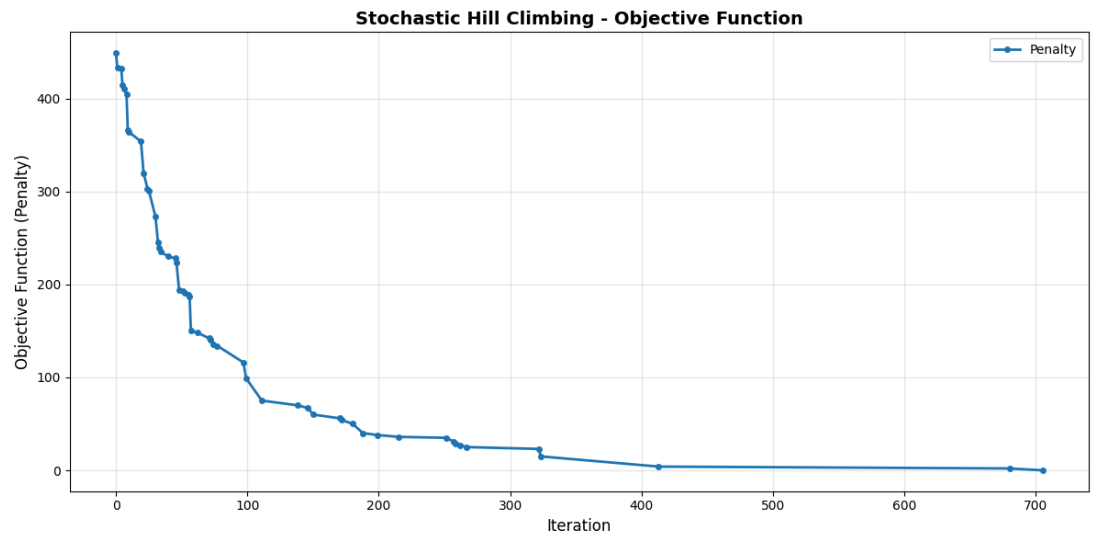
JADWAL RUANGAN: 7605					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7			IF3200_K01		
8			IF3200_K01		
9			IF3160_K02	IF3180_K01	
10			IF3160_K02	IF3180_K01	
11			IF3160_K02	IF3180_K01	
12					
13					
14					
15					
16	IF3150_K01	IF3190_K02			
17		IF3190_K02			IF3130_K02
18					
JADWAL RUANGAN: 7606					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8					IF3180_K02
9		IF3240_K02		IF3220_K02	
10					
11	IF3071_K01				IF3071_K01
12					
13	IF3130_K02				IF3220_K01
14	IF3240_K01				IF3220_K01
15	IF3240_K01			IF3240_K02	IF3220_K01
16			IF3110_K01		
17			IF3110_K01		
18					
JADWAL RUANGAN: 7607					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					IF3230_K01
8					
9	IF3160_K01				
10	IF3160_K01				
11				IF3071_K02	
12				IF3130_K01	
13					
14					
15					
16					
17					
18					

JADWAL RUANGAN: 7608					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8				IF3071_K01	
9					
10			IF3140_K01		
11					
12		IF3071_K02		IF3180_K02	
13					IF3110_K02
14					IF3110_K02
15					IF3110_K02
16				IF3180_K02	
17					
18					
JADWAL RUANGAN: 7609					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8					IF3230_K02
9					IF3230_K02
10					
11		IF3071_K02			
12					
13			IF3140_K02		
14			IF3140_K02		
15					
16	IF3160_K01				
17					
18					
JADWAL RUANGAN: 7610					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8					
9			IF3140_K01		IF3200_K02
10				IF3200_K02	IF3190_K01
11					
12	IF3240_K02				
13					
14					
15					IF3200_K01
16	IF3200_K02				IF3240_K01
17					
18					

JADWAL RUANGAN: multimedia					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8					
9					
10					
11				IF3170_K01	
12	IF3250_K01				
13					
14					
15					
16		IF3170_K01			
17					
18					
JADWAL RUANGAN: lab_pemrograman					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8					
9	IF3150_K02				
10	IF3150_K02				
11	IF3150_K02				
12	IF3150_K02				IF3210_K01
13					
14		IF3250_K01			
15				IF3220_K01	
16					
17					
18					
JADWAL RUANGAN: lab_jaringan					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8					
9					
10					
11					
12					
13					
14					
15			IF3190_K01		
16					
17		IF3130_K01			
18					

RESULTS					
Final Objective Function: 0					
Number of Iterations: 705					
Duration: 0.3619 seconds					

Visualisasi



3

State Awal dan Akhir

INITIAL STATE			
Kode	Ruangan	Hari	Jam
IF3071_K01	7603	Selasa	14-16
IF3071_K01	multimedia	Kamis	13-14
IF3071_K02	7610	Rabu	12-13
IF3071_K02	7602	Kamis	10-11
IF3071_K02	7605	Kamis	16-17
IF3130_K01	7608	Senin	16-17
IF3130_K01	7607	Jumat	10-11
IF3130_K02	7610	Selasa	8-9
IF3130_K02	7606	Jumat	9-10
IF3110_K01	lab_jaringan	Jumat	12-13
IF3110_K01	7607	Senin	14-16
IF3110_K02	7607	Kamis	16-18
IF3110_K02	7606	Senin	15-16
IF3140_K01	7604	Kamis	8-9
IF3140_K01	7603	Kamis	12-13
IF3140_K02	multimedia	Jumat	14-15
IF3140_K02	lab_jaringan	Jumat	14-15
IF3150_K01	7607	Kamis	9-13
IF3150_K02	7603	Jumat	17-18
IF3150_K02	7604	Senin	15-17
IF3150_K02	7603	Selasa	9-10
IF3160_K01	lab_pemrograman	Rabu	12-14
IF3160_K01	7606	Selasa	8-9
IF3160_K02	lab_pemrograman	Selasa	9-12
IF3170_K01	7603	Kamis	7-9
IF3170_K02	7607	Jumat	11-13
IF3180_K01	7603	Selasa	14-15
IF3180_K01	7602	Kamis	15-17
IF3180_K02	7608	Selasa	8-11
IF3190_K01	lab_jaringan	Jumat	16-18
IF3190_K02	7603	Jumat	7-8
IF3190_K02	multimedia	Jumat	17-18
IF3200_K01	multimedia	Selasa	13-14
IF3200_K01	7610	Rabu	11-12
IF3200_K01	7609	Kamis	16-17
IF3200_K02	7602	Jumat	10-12
IF3200_K02	7610	Rabu	13-14
IF3210_K01	7602	Senin	16-18
IF3210_K02	7609	Selasa	8-9
IF3220_K02	7604	Jumat	14-15
IF3220_K01	7610	Kamis	11-14
IF3220_K01	7609	Selasa	14-15
IF3220_K02	lab_jaringan	Selasa	9-10
IF3220_K02	7608	Kamis	8-11
IF3230_K01	lab_pemrograman	Kamis	9-11
IF3230_K02	7605	Kamis	11-13
IF3240_K01	lab_jaringan	Rabu	11-13
IF3240_K01	7608	Senin	8-9
IF3240_K02	7602	Jumat	14-16
IF3240_K02	7609	Selasa	12-13
IF3250_K01	7605	Selasa	11-13
IF3250_K02	7610	Senin	10-12

Initial Objective Function: 438.0

FINAL STATE					
JADWAL RUANGAN: 7602					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8			IF3230_K01		
9			IF3230_K01		
10					IF3170_K01
11					IF3170_K01
12					
13					
14					
15					
16					
17					
18					
JADWAL RUANGAN: 7603					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					IF3190_K02
8					
9		IF3150_K02	IF3150_K02		
10			IF3150_K02		
11					
12				IF3190_K02	
13					
14		IF3250_K02			
15		IF3250_K02	IF3140_K02		
16					
17					
18					
JADWAL RUANGAN: 7604					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8				IF3150_K02	
9					
10					
11					
12					
13					
14					IF3210_K02
15					
16					
17					
18					

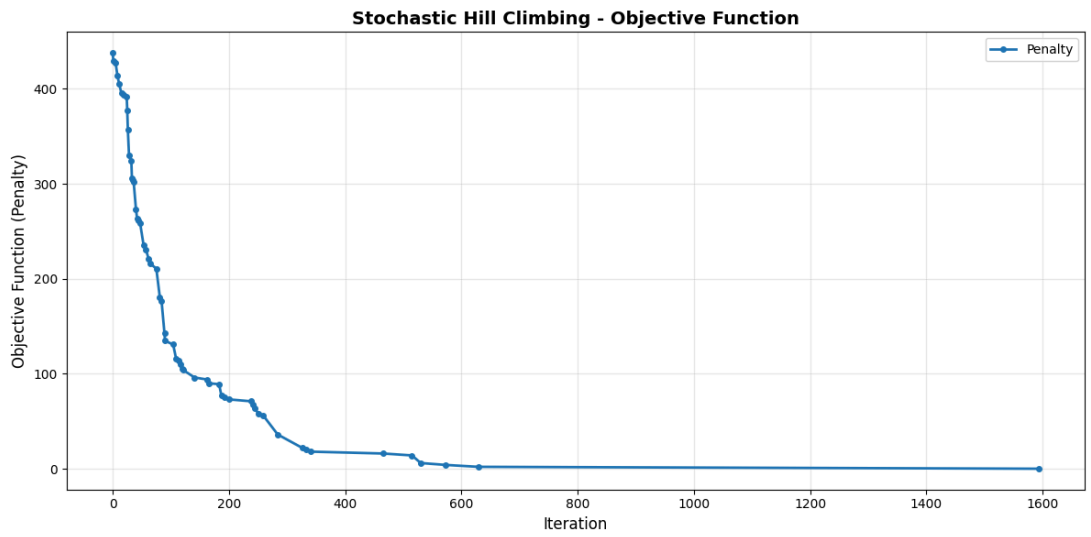
```
=====JADWAL RUANGAN: multimedia=====
Jam      Senin      Selasa      Rabu      Kamis      Jumat
=====
7
8          IF3170_K02
9          IF3170_K02
10
11
12
13
14
15
16
17
18
=====

JADWAL RUANGAN: lab_pemrograman
=====
Jam      Senin      Selasa      Rabu      Kamis      Jumat
=====
7      IF3130_K02
8
9          IF3220_K02      IF3140_K02
10         IF3220_K02
11         IF3220_K02
12
13          IF3250_K01      IF3250_K01
14
15          IF3210_K02
16
17          IF3130_K02
18
=====

JADWAL RUANGAN: lab_jaringan
=====
Jam      Senin      Selasa      Rabu      Kamis      Jumat
=====
7
8
9
10
11
12
13          IF3210_K01      IF3150_K01
14          IF3210_K01      IF3150_K01
15
16
17          IF3190_K01
18          IF3190_K01
=====
```

```
=====
                        RESULTS
=====
Final Objective Function: 0
Number of Iterations: 1593
Duration: 0.7730 seconds
```

Visualisasi



3.1.2. Simulated Annealing

(Initial temperature = 1000, cooling rate = 0.999, final temperature = 0.01)

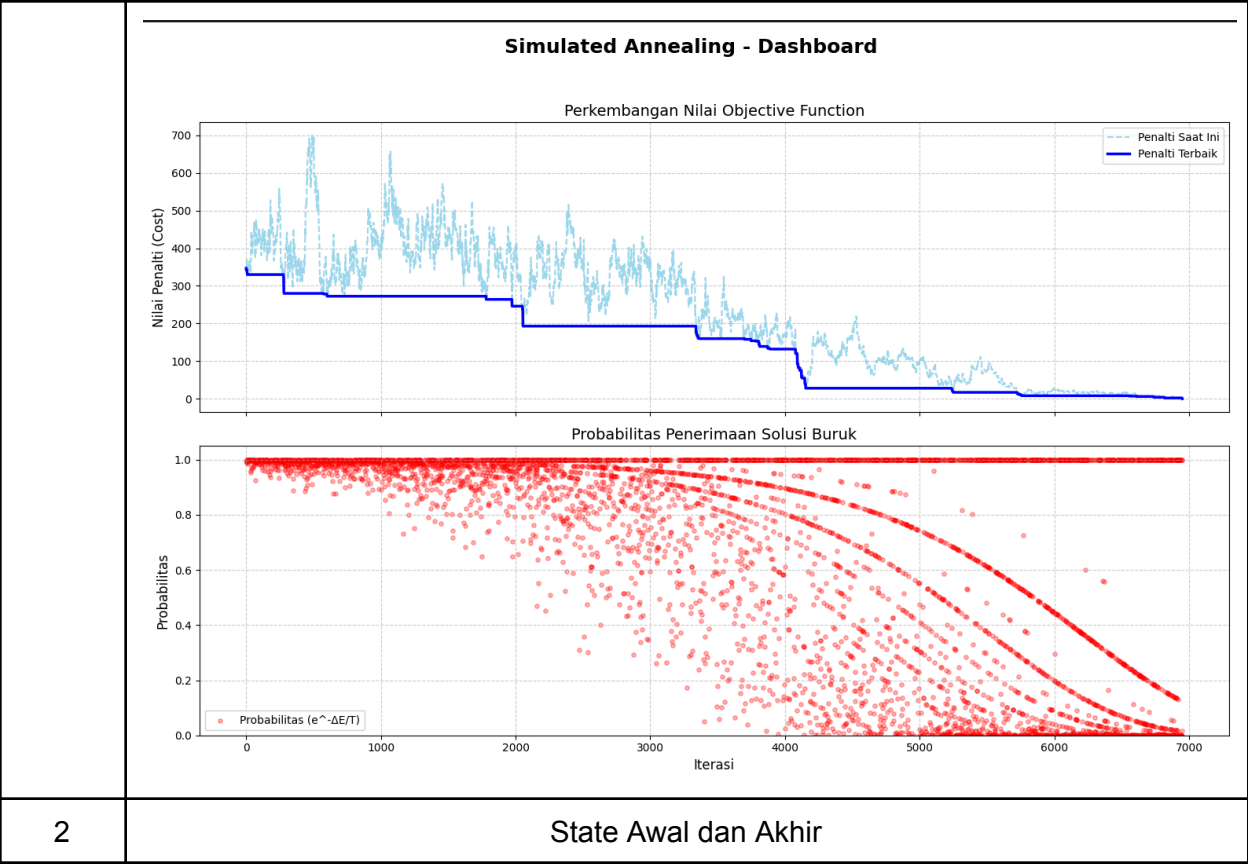
Iterasi	State Awal dan Akhir	
1	<pre> ===== INITIAL STATE ===== Kode Ruangan Hari Jam ----- IF3071_K01 lab_pemrograman Jumat 7-10 IF3071_K02 lab_jaringan Rabu 11-14 IF3130_K01 7604 Rabu 11-12 IF3130_K01 7603 Senin 15-16 IF3130_K02 7609 Senin 10-11 IF3130_K02 7609 Selasa 11-12 IF3110_K01 7606 Kamis 15-16 IF3110_K01 7602 Senin 15-17 IF3110_K02 lab_jaringan Jumat 7-10 IF3140_K01 7609 Senin 11-13 IF3140_K02 7606 Kamis 13-14 IF3140_K02 7607 Senin 12-13 IF3150_K01 7602 Kamis 10-12 IF3150_K01 7603 Senin 10-12 IF3150_K02 7604 Kamis 14-17 IF3150_K02 7607 Senin 10-11 IF3160_K01 7607 Rabu 8-10 IF3160_K01 7609 Rabu 15-16 IF3160_K02 lab_pemrograman Jumat 11-14 IF3170_K01 7608 Selasa 10-12 IF3170_K02 7607 Selasa 9-11 IF3180_K01 7603 Jumat 11-12 IF3180_K01 7608 Jumat 14-15 IF3180_K01 7608 Kamis 7-8 IF3180_K02 7607 Jumat 8-11 IF3190_K01 lab_jaringan Rabu 9-11 IF3190_K02 7602 Rabu 13-15 IF3200_K01 7609 Jumat 12-15 IF3200_K02 7610 Jumat 12-13 IF3200_K02 7602 Selasa 12-13 IF3200_K02 7606 Jumat 16-17 IF3210_K01 7604 Selasa 8-9 IF3210_K01 7608 Selasa 12-13 IF3210_K02 7606 Senin 14-16 IF3220_K01 7609 Selasa 17-18 IF3220_K01 7602 Selasa 8-11 IF3220_K02 7606 Rabu 12-14 IF3220_K02 7609 Kamis 9-11 IF3230_K01 7605 Selasa 14-16 IF3230_K02 multimedia Senin 7-8 IF3230_K02 multimedia Selasa 7-8 IF3240_K01 lab_jaringan Selasa 16-17 IF3240_K01 lab_pemrograman Rabu 10-11 IF3240_K01 7606 Selasa 15-16 IF3240_K02 7609 Selasa 8-11 IF3250_K01 7607 Selasa 10-11 IF3250_K01 lab_jaringan Selasa 12-13 IF3250_K02 7603 Rabu 14-16 ===== Initial Objective Function: 346.5 </pre>	<pre> ===== FINAL STATE ===== JADWAL RUANGAN: 7602 ===== Jam Senin Selasa Rabu Kamis Jumat ----- 7 8 9 10 11 12 13 14 15 16 17 18 ===== JADWAL RUANGAN: 7603 ===== Jam Senin Selasa Rabu Kamis Jumat ----- 7 8 9 10 11 12 13 14 15 16 17 18 ===== JADWAL RUANGAN: 7604 ===== Jam Senin Selasa Rabu Kamis Jumat ----- 7 8 9 10 11 IF3230_K02 12 13 14 15 16 17 18 ===== </pre>

JADWAL RUANGAN: 7605					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8	IF3071_K02	IF3160_K01		IF3160_K01	
9	IF3071_K02			IF3160_K01	
10	IF3071_K02	IF3210_K02			
11		IF3210_K02			
12	IF3200_K02				
13				IF3240_K02	
14				IF3240_K02	
15			IF3220_K01	IF3240_K02	
16			IF3220_K01		
17			IF3220_K01		
18					
JADWAL RUANGAN: 7606					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7	IF3210_K01		IF3220_K02		
8		IF3110_K02	IF3220_K02		
9		IF3110_K02	IF3110_K01		
10		IF3110_K02			
11	IF3110_K01				
12	IF3110_K01				
13		IF3180_K01			
14					
15					
16			IF3150_K01		
17	IF3240_K01	IF3230_K02	IF3150_K01	IF3240_K01	
18					
JADWAL RUANGAN: 7607					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8					
9					
10	IF3130_K01				
11					
12					
13					
14					
15					
16					
17					
18					

JADWAL RUANGAN: 7608					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7			IF3160_K02		
8			IF3160_K02		
9			IF3160_K02		
10					
11					
12					
13				IF3071_K01	
14				IF3071_K01	IF3140_K02
15	IF3180_K01			IF3071_K01	
16					
17					
18					
JADWAL RUANGAN: 7609					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8					
9		IF3140_K01			
10		IF3140_K01			
11				IF3150_K02	IF3170_K02
12				IF3150_K02	IF3170_K02
13			IF3230_K01	IF3150_K02	
14	IF3150_K02				
15					
16	IF3200_K02				
17					
18					
JADWAL RUANGAN: 7610					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8			IF3180_K01		IF3240_K01
9		IF3220_K01			
10					
11			IF3180_K02		
12	IF3210_K01		IF3180_K02		
13			IF3180_K02		
14					IF3200_K01
15		IF3190_K02			IF3200_K01
16		IF3190_K02		IF3200_K02	IF3200_K01
17					
18					

JADWAL RUANGAN: multimedia					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8					
9					
10					
11					
12					
13	IF3170_K01				
14	IF3170_K01				
15					
16					
17					
18					
JADWAL RUANGAN: lab_pemrograman					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8					
9				IF3220_K02	IF3250_K01
10				IF3220_K02	
11					
12			IF3150_K01		
13			IF3150_K01		
14		IF3130_K01			
15					
16					IF3140_K02
17				IF3130_K02	
18					
JADWAL RUANGAN: lab_jaringan					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8					
9					
10					
11			IF3250_K01		
12	IF3250_K02				
13	IF3250_K02				
14					
15					
16	IF3190_K01				
17	IF3190_K01				
18					

RESULTS					
Final objective Function: 0					
Total Iterations: 6951					
Duration: 3.1623 seconds					
Frekuensi 'Stuck' di Local Optima:					
- Jumlah periode stagnasi: 40 kali					
- Durasi stagnasi terpanjang: 1284 iterasi					
- Rata-rata durasi stagnasi: 143.60 iterasi					



	INITIAL STATE			
	Kode	Ruangan	Hari	Jam
=====				
IF3071_K01	7606	Kamis	11-12	
IF3071_K01	multimedia	Senin	10-12	
IF3071_K02	7604	Kamis	14-15	
IF3071_K02	multimedia	Jumat	12-13	
IF3071_K02	7606	Senin	7-8	
IF3130_K01	7605	Kamis	14-16	
IF3130_K02	7607	Jumat	10-12	
IF3110_K01	7605	Rabu	15-17	
IF3110_K01	7605	Jumat	12-13	
IF3110_K02	7607	Kamis	13-14	
IF3110_K02	7604	Selasa	16-18	
IF3140_K01	7607	Senin	11-13	
IF3140_K02	7602	Kamis	11-13	
IF3150_K01	7605	Jumat	9-13	
IF3150_K02	7607	Senin	14-16	
IF3150_K02	7609	Rabu	15-16	
IF3150_K02	multimedia	Senin	17-18	
IF3160_K01	lab_pemrograman	Rabu	11-12	
IF3160_K01	7603	Selasa	9-11	
IF3160_K02	lab_pemrograman	Senin	14-17	
IF3170_K01	7602	Jumat	12-13	
IF3170_K01	7609	Kamis	9-10	
IF3170_K02	7605	Jumat	12-13	
IF3170_K02	7607	Kamis	17-18	
IF3180_K01	multimedia	Selasa	9-12	
IF3180_K02	lab_jaringan	Selasa	14-15	
IF3180_K02	multimedia	Kamis	11-13	
IF3190_K01	multimedia	Kamis	12-14	
IF3190_K02	multimedia	Kamis	10-11	
IF3190_K02	7608	Selasa	10-11	
IF3200_K01	7610	Jumat	15-18	
IF3200_K02	7605	Rabu	13-15	
IF3200_K02	7602	Rabu	14-15	
IF3210_K01	7604	Senin	9-11	
IF3210_K02	7606	Rabu	14-15	
IF3210_K02	7607	Kamis	12-13	
IF3220_K01	lab_jaringan	Senin	9-10	
IF3220_K01	7605	Rabu	17-18	
IF3220_K01	7605	Jumat	11-13	
IF3220_K02	7609	Selasa	15-17	
IF3220_K02	7610	Rabu	10-12	
IF3230_K01	7602	Senin	7-9	
IF3230_K02	7609	Kamis	12-13	
IF3230_K02	7605	Senin	15-16	
IF3240_K01	7609	Kamis	13-14	
IF3240_K01	7606	Kamis	16-18	
IF3240_K02	7604	Jumat	13-16	
IF3250_K01	7607	Selasa	16-18	
IF3250_K02	7602	Senin	10-11	
IF3250_K02	7610	Kamis	16-17	
=====				
Initial Objective Function: 635.25				

FINAL STATE					
=====					
JADWAL RUANGAN: 7602					
=====					
Jam	Senin	Selasa	Rabu	Kamis	Jumat

7					IF3150_K02
8					IF3150_K02
9					
10					
11					
12					
13					
14					
15					
16					
17	IF3170_K02				
18					
=====					
JADWAL RUANGAN: 7603					
=====					
Jam	Senin	Selasa	Rabu	Kamis	Jumat

7			IF3230_K02		
8					
9					
10					
11					
12			IF3150_K02		
13			IF3170_K01		
14				IF3220_K01	
15					
16					
17					
18					
=====					
JADWAL RUANGAN: 7604					
=====					
Jam	Senin	Selasa	Rabu	Kamis	Jumat

7					
8					
9					
10					
11					
12					
13					
14					IF3250_K02
15					
16			IF3150_K02		
17					
18					
=====					

JADWAL RUANGAN: 7605					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7	IF3240_K02			IF3250_K02	
8	IF3240_K02				
9	IF3240_K02				IF3200_K02
10					IF3200_K02
11					IF3071_K02
12					
13	IF3230_K01				
14	IF3230_K01				
15		IF3160_K01			
16		IF3160_K01		IF3071_K01	
17				IF3071_K01	
18					
JADWAL RUANGAN: 7606					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7				IF3071_K02	
8	IF3110_K01				
9	IF3110_K01				
10	IF3240_K01				
11	IF3240_K01				
12	IF3130_K02				
13	IF3130_K02				
14					IF3140_K01
15					IF3140_K01
16					
17					IF3180_K02
18					
JADWAL RUANGAN: 7607					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7	IF3220_K01				
8					
9				IF3220_K01	
10				IF3220_K01	
11					
12					IF3210_K02
13					IF3170_K02
14					
15			IF3210_K01		
16			IF3210_K01		
17					
18					

JADWAL RUANGAN: 7608					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7		IF3110_K02	IF3160_K01		
8					IF3220_K02
9					IF3220_K02
10		IF3180_K02			
11		IF3180_K02			
12			IF3240_K01		
13				IF3110_K02	
14			IF3230_K02	IF3110_K02	
15					
16					
17					
18					
JADWAL RUANGAN: 7609					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8	IF3190_K02	IF3220_K02			
9		IF3220_K02			
10					
11					
12		IF3200_K01			
13		IF3200_K01			
14		IF3200_K01	IF3160_K02		
15		IF3190_K02	IF3160_K02		
16			IF3160_K02		
17					
18					
JADWAL RUANGAN: 7610					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8					
9					
10	IF3210_K02			IF3071_K02	IF3180_K01
11			IF3200_K02		IF3180_K01
12					IF3180_K01
13	IF3110_K01		IF3071_K01		
14					
15					
16					
17					
18					

JADWAL RUANGAN: multimedia					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7			IF3250_K01		
8			IF3250_K01		
9		IF3170_K01			
10					
11					
12					
13					
14					
15					
16					
17					
18					
JADWAL RUANGAN: lab_pemrograman					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7				IF3190_K01	
8				IF3190_K01	
9					
10					
11	IF3140_K02				
12	IF3140_K02				
13					
14					
15					
16					
17					
18					
JADWAL RUANGAN: lab_jaringan					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8					
9					
10					
11					
12					
13		IF3130_K01			IF3150_K01
14		IF3130_K01			IF3150_K01
15					IF3150_K01
16					IF3150_K01
17					
18					

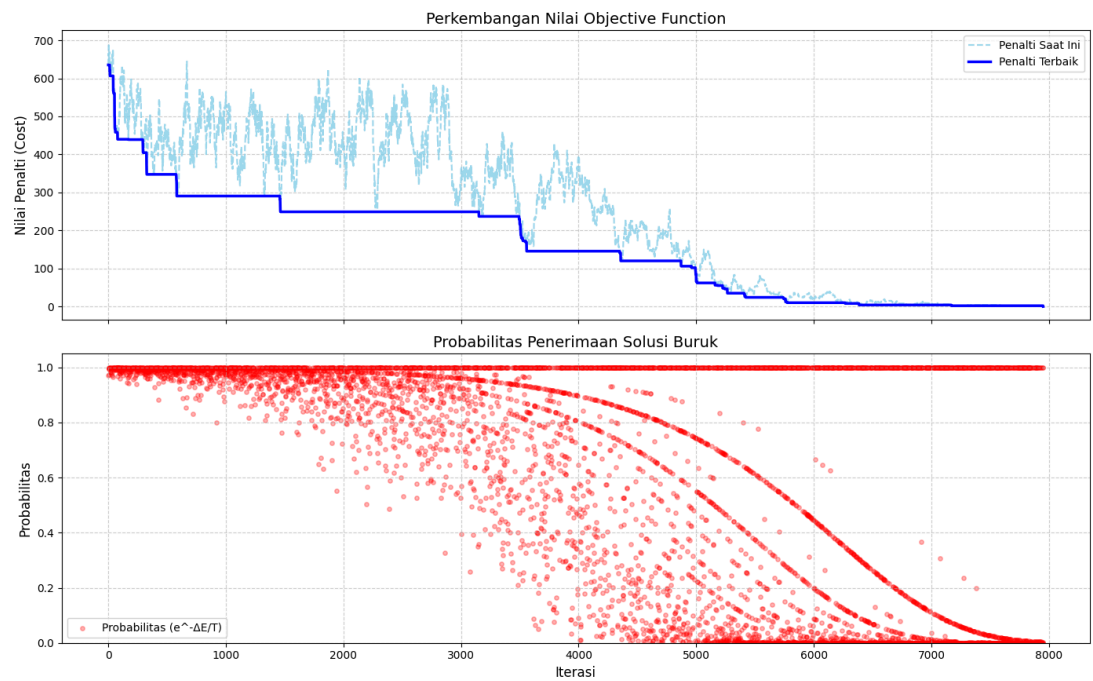
```

=====
RESULTS
=====
Final Objective Function: 0
Total Iterations: 7952
Duration: 3.7767 seconds
Frekuensi 'Stuck' di Local Optima:
- Jumlah periode stagnasi: 51 kali
- Durasi stagnasi terpanjang: 1687 iterasi
- Rata-rata durasi stagnasi: 154.45 iterasi
=====

```

Visualisasi

Simulated Annealing - Dashboard



3

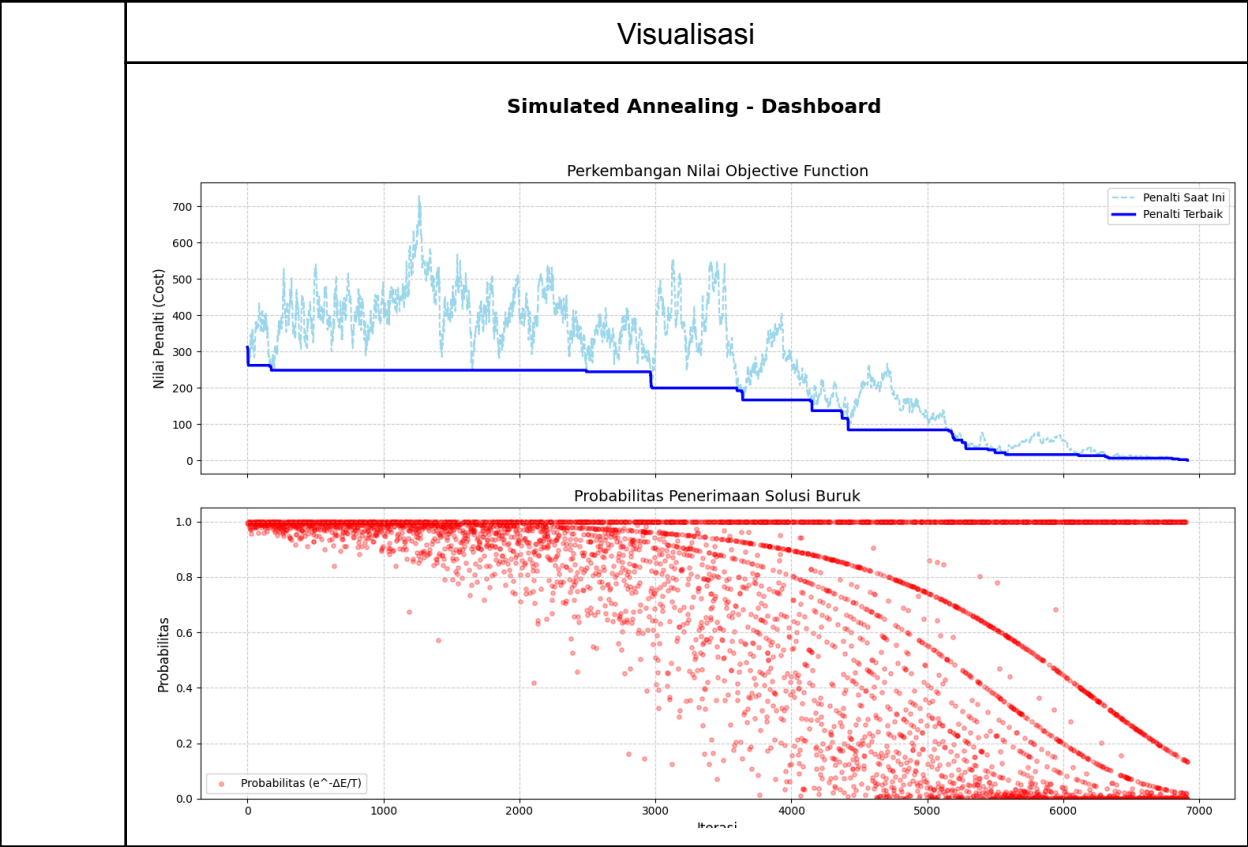
State Awal dan Akhir

INITIAL STATE			
Kode	Ruangan	Hari	Jam
IF3071_K01	7609	Rabu	8-10
IF3071_K01	7608	Senin	12-13
IF3071_K02	7605	Kamis	11-14
IF3130_K01	7603	Jumat	16-17
IF3130_K01	lab_jaringan	Senin	17-18
IF3130_K02	lab_pemrograman	Senin	13-15
IF3110_K01	7609	Kamis	12-14
IF3110_K01	multimedia	Kamis	11-12
IF3110_K02	7604	Rabu	11-12
IF3110_K02	7607	Senin	13-14
IF3110_K02	7604	Rabu	12-13
IF3140_K01	lab_pemrograman	Senin	9-10
IF3140_K01	7610	Jumat	10-11
IF3140_K02	7607	Jumat	11-13
IF3150_K01	7607	Kamis	8-12
IF3150_K02	7602	Selasa	8-10
IF3150_K02	7602	Kamis	12-14
IF3160_K01	7608	Jumat	13-16
IF3160_K02	7606	Rabu	8-9
IF3160_K02	7608	Jumat	10-12
IF3170_K01	multimedia	Senin	7-8
IF3170_K01	7605	Selasa	16-17
IF3170_K02	lab_jaringan	Jumat	12-13
IF3170_K02	7606	Jumat	7-8
IF3180_K01	7607	Rabu	11-13
IF3180_K01	7610	Senin	12-13
IF3180_K02	lab_pemrograman	Kamis	15-17
IF3180_K02	multimedia	Senin	14-15
IF3190_K01	multimedia	Senin	10-12
IF3190_K02	7607	Senin	10-12
IF3200_K01	7606	Kamis	15-16
IF3200_K01	lab_pemrograman	Jumat	13-15
IF3200_K02	7610	Selasa	9-10
IF3200_K02	7610	Kamis	14-16
IF3210_K01	7609	Rabu	16-17
IF3210_K01	lab_jaringan	Kamis	11-12
IF3210_K02	lab_pemrograman	Senin	17-18
IF3210_K02	7606	Rabu	7-8
IF3220_K01	lab_jaringan	Rabu	7-8
IF3220_K01	7603	Senin	7-9
IF3220_K01	7609	Senin	8-9
IF3220_K02	multimedia	Kamis	13-17
IF3230_K01	multimedia	Kamis	10-11
IF3230_K01	7608	Senin	7-8
IF3230_K02	multimedia	Selasa	15-16
IF3230_K02	7602	Kamis	17-18
IF3240_K01	7603	Senin	10-11
IF3240_K01	7607	Senin	9-10
IF3240_K01	7610	Selasa	14-15
IF3240_K02	7605	Jumat	11-14
IF3250_K01	lab_pemrograman	Kamis	15-16
IF3250_K01	7606	Jumat	9-10
IF3250_K02	7604	Selasa	14-15
IF3250_K02	multimedia	Selasa	12-13
Initial Objective Function: 312.0			

FINAL STATE					
JADMAL RUANGAN: 7602					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7					
8					
9		IF3250_K01			
10					
11					
12		IF3210_K02			
13					
14					
15					
16					
17				IF3230_K01	
18					
JADMAL RUANGAN: 7603					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7				IF3140_K02	
8				IF3140_K02	
9					
10	IF3150_K02				
11	IF3150_K02				
12					
13					
14					IF3130_K01
15	IF3150_K02				
16	IF3150_K02				
17					
18					
JADMAL RUANGAN: 7604					
Jam	Senin	Selasa	Rabu	Kamis	Jumat
7		IF3250_K02			
8					
9					
10					
11					
12					
13					
14					
15					
16					
17					
18					

```
JADWAL RUANGAN: multimedia
=====
Jam      Senin      Selasa      Rabu      Kamis      Jumat
-----
7
8
9
10
11
12
13
14
15
16
17
18
=====
JADWAL RUANGAN: lab_pemrograman
=====
Jam      Senin      Selasa      Rabu      Kamis      Jumat
-----
7
8
9
10
11
12
13
14
15
16
17
18
=====
JADWAL RUANGAN: lab_jaringan
=====
Jam      Senin      Selasa      Rabu      Kamis      Jumat
-----
7
8
9
10
11
12
13
14
15
16
17
18
=====
```

```
=====
                        RESULTS
=====
Final Objective Function: 0
Total Iterations: 6915
Duration: 3.5717 seconds
Frekuensi 'stuck' di Local Optima:
- Jumlah periode stagnasi: 41 kali
- Durasi stagnasi terpanjang: 2316 iterasi
- Rata-rata durasi stagnasi: 167.37 iterasi
=====
```



3.1.3. Genetic Algorithm

3.1.3.1. Populasi sebagai kontrol (50 populasi)

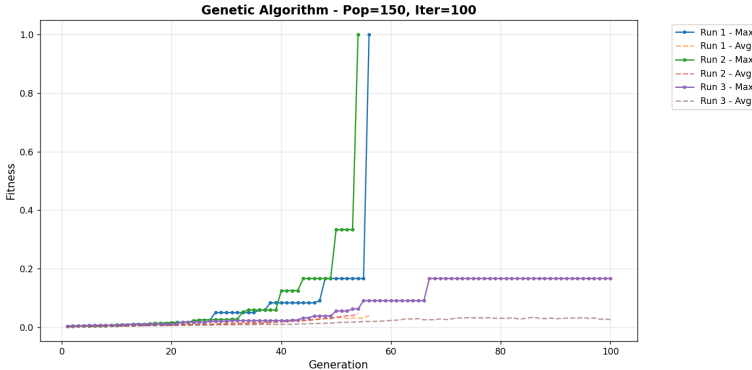
Iterasi	State Awal dan Akhir	Visualisasi dan Ringkasan
50 iterasi	<div><div>Run 1/3</div><div>----- Initial fitness: 0.001684 Initial penalty: 462.75 Final fitness: 0.076923 Final penalty: 24 Generations completed: 50 Total individuals evaluated: 2300 Duration: 32.1352 seconds</div><div>Run 2/3</div><div>----- Initial fitness: 0.001903 Initial penalty: 396.5 Final fitness: 0.027778 Final penalty: 45 Generations completed: 50 Total individuals evaluated: 2300 Duration: 17.0943 seconds</div></div>	<div><div>Genetic Algorithm - Pop=50, Iter=50</div></div>

	<div>Run 3/3</div> <div>-----</div> <div>Initial fitness: 0.001817 Initial penalty: 491.25 Final fitness: 1.000000 Final penalty: 0 Generations completed: 50 Total individuals evaluated: 2300 Duration: 30.1251 seconds</div>	<div>=====</div> <div>SUMMARY FOR Configuration: Pop=50, Iter=50</div> <div>=====</div> <div>Average Final Fitness: 0.368234 Average Final Penalty: 23.00 Average Total Individuals Evaluated: 2300.0 Average Duration: 26.4515 seconds</div> <div>=====</div>
100 iterasi	<div>Run 1/3</div> <div>-----</div> <div>Initial fitness: 0.001618 Initial penalty: 547.0 Final fitness: 0.166667 Final penalty: 5 Generations completed: 100 Total individuals evaluated: 4600 Duration: 29.6097 seconds</div> <div>Run 2/3</div> <div>-----</div> <div>Initial fitness: 0.001653 Initial penalty: 534.0 Final fitness: 1.000000 Final penalty: 0 Generations completed: 69 Total individuals evaluated: 3128 Duration: 25.3283 seconds</div> <div>Run 3/3</div> <div>-----</div> <div>Initial fitness: 0.001568 Initial penalty: 577.75 Final fitness: 1.000000 Final penalty: 0 Generations completed: 71 Total individuals evaluated: 3220 Duration: 36.9129 seconds</div>	<div>Genetic Algorithm - Pop=50, Iter=100</div> <div>=====</div> <div>SUMMARY FOR Configuration: Pop=50, Iter=100</div> <div>=====</div> <div>Average Final Fitness: 0.722222 Average Final Penalty: 1.67 Average Total Individuals Evaluated: 3649.3333333333335 Average Duration: 30.6170 seconds</div> <div>=====</div>
150 iterasi	<div>Run 1/3</div> <div>-----</div> <div>Initial fitness: 0.002363 Initial penalty: 402.25 Final fitness: 0.166667 Final penalty: 5 Generations completed: 150 Total individuals evaluated: 6900 Duration: 52.5985 seconds</div> <div>Run 2/3</div> <div>-----</div> <div>Initial fitness: 0.001349 Initial penalty: 748.5 Final fitness: 0.045455 Final penalty: 40 Generations completed: 150 Total individuals evaluated: 6900 Duration: 76.2161 seconds</div>	<div>Genetic Algorithm - Pop=50, Iter=150</div> <div>=====</div>

	<pre> Run 3/3 ----- Initial fitness: 0.001837 Initial penalty: 392.5 Final fitness: 1.000000 Final penalty: 0 Generations completed: 82 Total individuals evaluated: 3726 Duration: 25.0739 seconds </pre>	<pre> ===== SUMMARY FOR Configuration: Pop=50, Iter=150 ===== Average Final Fitness: 0.404040 Average Final Penalty: 15.00 Average Total Individuals Evaluated: 5842.0 Average Duration: 51.2962 seconds ===== </pre>
--	--	---

3.1.3.2. Iterasi sebagai kontrol (100 iterasi)

Populasi	State Awal dan Akhir	Visualisasi dan Ringkasan
50 individu	<pre> Run 1/3 ----- Initial fitness: 0.001747 Initial penalty: 577.25 Final fitness: 0.111111 Final penalty: 24 Generations completed: 100 Total individuals evaluated: 4600 Duration: 31.9072 seconds </pre> <pre> Run 2/3 ----- Initial fitness: 0.002191 Initial penalty: 368.5 Final fitness: 1.000000 Final penalty: 0 Generations completed: 65 Total individuals evaluated: 2944 Duration: 32.8192 seconds </pre> <pre> Run 3/3 ----- Initial fitness: 0.001939 Initial penalty: 522.75 Final fitness: 0.018182 Final penalty: 85 Generations completed: 100 Total individuals evaluated: 4600 Duration: 40.9512 seconds </pre>	<p>Genetic Algorithm - Pop=50, Iter=100</p> <pre> ===== SUMMARY FOR Configuration: Pop=50, Iter=100 ===== Average Final Fitness: 0.376431 Average Final Penalty: 36.33 Average Total Individuals Evaluated: 4048.0 Average Duration: 35.2259 seconds ===== </pre>
100 individu	<pre> Run 1/3 ----- Initial fitness: 0.002220 Initial penalty: 483.5 Final fitness: 1.000000 Final penalty: 0 Generations completed: 72 Total individuals evaluated: 6390 Duration: 95.3614 seconds </pre>	<p>Genetic Algorithm - Pop=100, Iter=100</p>

	<pre> Run 2/3 ----- Initial fitness: 0.002530 Initial penalty: 436.25 Final fitness: 0.333333 Final penalty: 2 Generations completed: 100 Total individuals evaluated: 9000 Duration: 133.5406 seconds </pre> <pre> Run 3/3 ----- Initial fitness: 0.001959 Initial penalty: 481.5 Final fitness: 1.000000 Final penalty: 0 Generations completed: 69 Total individuals evaluated: 6120 Duration: 93.6504 seconds </pre>	<pre> ===== SUMMARY FOR Configuration: Pop=100, Iter=100 ===== Average Final Fitness: 0.777778 Average Final Penalty: 0.67 Average Total Individuals Evaluated: 7170.0 Average Duration: 107.5175 seconds ===== </pre>
150 individu	<pre> Run 1/3 ----- Initial fitness: 0.002581 Initial penalty: 380.5 Final fitness: 1.000000 Final penalty: 0 Generations completed: 56 Total individuals evaluated: 7480 Duration: 156.9747 seconds </pre> <pre> Run 2/3 ----- Initial fitness: 0.002149 Initial penalty: 421.25 Final fitness: 1.000000 Final penalty: 0 Generations completed: 54 Total individuals evaluated: 7208 Duration: 266.2718 seconds </pre> <pre> Run 3/3 ----- Initial fitness: 0.002066 Initial penalty: 448.0 Final fitness: 0.166667 Final penalty: 5 Generations completed: 100 Total individuals evaluated: 13600 Duration: 393.0044 seconds </pre>	 <pre> ===== SUMMARY FOR Configuration: Pop=150, Iter=100 ===== Average Final Fitness: 0.722222 Average Final Penalty: 1.67 Average Total Individuals Evaluated: 9429.333333333334 Average Duration: 272.0836 seconds ===== </pre>

3.1.4. Heuristik Yang Digunakan

Persoalan penjadwalan kelas mingguan, yang termasuk dalam kategori NP-hard, ruang pencarian solusinya sangatlah besar. Varian *Hill Climbing* seperti *Steepest Ascent* idealnya mengevaluasi semua kemungkinan tetangga dari state saat ini untuk menentukan langkah terbaik berikutnya. Namun, jumlah tetangga potensial dari hasil dari *swap* dan *move* bisa mencapai ribuan hingga puluhan ribu untuk satu jadwal saja. Mengevaluasi *objective function* untuk setiap tetangga di setiap iterasi akan membuat algoritma sangat lambat dan tidak praktis untuk digunakan.

Maka dari itu, akan digunakan heuristik berupa sampling tetangga pada algoritma varian *Hill Climbing* dengan menambahkan parameter *max neighbor* yang kemudian akan digunakan sebagai batas maksimal pembangkitan tetangga hasil dari proses move. Heuristik ini secara langsung mengatasi masalah ledakan jumlah tetangga. Dengan membatasi evaluasi hanya pada sampel acak, misalnya 500 tetangga, akan dapat mengurangi beban komputasi di setiap iterasi secara drastis. Ini membuat algoritma *Hill Climbing* menjadi *feasible* dan dapat dijalankan dalam waktu yang masuk akal.

Memang akan terdapat *trade-off* apabila menggunakan heuristik sampling tetangga. Algoritma akan kehilangan jaminan bahwa langkah yang diambil adalah yang paling curam. Algoritma mungkin mengambil jalur yang sedikit berbeda atau bahkan terjebak di *local optimum* yang berbeda dibandingkan jika semua tetangga dievaluasi. Namun, keuntungan dalam hal kecepatan komputasi biasanya jauh lebih besar daripada potensi penurunan kualitas solusi, terutama ketika dikombinasikan dengan teknik lain seperti Random Restart atau Sideways Move.

3.1.5. Analisis

3.1.5.1. Seberapa dekat tiap-tiap algoritma bisa mendekati global optima dan mengapa hasilnya demikian?

- Steepest Ascent Hill Climbing: Algoritma ini tidak pernah mencapai global optima (penalti 0) dalam ketiga percobaan. Nilai penalty akhir berkisar antara 219.0 hingga 278.5. Hal ini wajar karena Steepest Ascent sangat rentan terjebak di *local optimum* pertama yang ditemui dan tidak memiliki mekanisme untuk keluar.
- Hill Climbing with Sideways Move: Algoritma ini berhasil mencapai global optima (penalti 0) pada satu dari tiga percobaan. Pada dua percobaan lainnya, algoritma ini terjebak di *local optimum* dengan penalty 154.25 dan 208.5. Hal tersebut karena kemampuan *sideways move* memberikan kesempatan untuk keluar dari *local optimum* dangkal, tetapi tidak menjamin pencapaian global optima jika terjebak di *local optimum* yang dalam atau batas *sideways move* tercapai.
- Random Restart Hill Climbing: Algoritma ini sangat efektif mendekati global optima. Algoritma ini berhasil mencapai penalty 0 pada dua dari tiga percobaan dan pada percobaan lainnya mencapai penalty sangat rendah yaitu 2.0. Mekanisme *restart* dari titik awal yang berbeda-beda secara signifikan meningkatkan peluang untuk

menemukan *global optimum* atau solusi yang sangat mendekati.

- Stochastic Hill Climbing: Algoritma ini secara konsisten berhasil mencapai global optima (penalti 0) dalam ketiga percobaan. Meskipun hanya memilih satu tetangga acak, sifat algoritma yang stokastik dan jumlah iterasi maksimum yang besar (5000) memberikan kesempatan luas untuk menjelajahi ruang solusi dan akhirnya menemukan solusi optimal.
- Simulated Annealing (SA): Algoritma ini juga secara konsisten berhasil mencapai global optima (penalti 0) dalam ketiga percobaan. Hal tersebut karena kemampuan algoritma SA untuk menerima solusi yang lebih buruk, terutama di awal saat suhu tinggi, sangat efektif untuk "melompat" keluar dari *local optimum* dan menjelajahi area solusi secara global. Pendinginan yang lambat (0.999) memberikan waktu yang cukup untuk konvergen ke solusi optimal.
- Genetic Algorithm (GA): Hasil GA bervariasi tergantung parameter. Dengan populasi 50:
 - Iterasi 50: Mencapai penalty 0 pada satu dari tiga run.
 - Iterasi 100: Mencapai penalty 0 pada dua dari tiga run.
 - Iterasi 150: Tidak mencapai penalty 0 (terendah 5). GA bisa mencapai global optima, tetapi keberhasilannya sangat dipengaruhi oleh parameter (ukuran populasi, jumlah generasi), operator genetik, dan faktor keacakan dalam evolusi.

3.1.5.2. Bagaimana perbandingan hasil pencarian tiap-tiap algoritma dengan algoritma local search yang lain?

- Stochastic Hill Climbing dan Simulated Annealing secara konsisten memberikan hasil yang terbaik yaitu penalty mencapai 0 atau global optima dalam eksperimen ini.
- Random Restart Hill Climbing juga memiliki hasil yang baik yaitu mencapai hasil dengan penalty 0 (global optima) atau penalti 2.0 yang cukup dekat dengan global optima.
- Genetic Algorithm menunjukkan hasil yang bervariasi. GA dapat mencapai penalty 0 atau nilai rendah (1.67 rata-rata pada iterasi 100), tetapi hasilnya tidak stabil antar run dan sangat bergantung pada parameter. Sideways Move Hill Climbing juga bervariasi, kadang optimal (penalti 0), kadang terjebak di nilai yang cukup tinggi (154-208).

- Steepest Ascent Hill Climbing secara konsisten menghasilkan penalti akhir yang paling tinggi yaitu antara 219 sampai 278 penalti. Ini menunjukkan bahwa algoritma Steepest Ascent paling mudah terjebak di *local optimum*.

3.1.5.3. Bagaimana perbandingan durasi proses pencarian tiap algoritma relatif terhadap algoritma lainnya?

- Stochastic Hill Climbing adalah yang tercepat secara signifikan. Algoritma ini dapat menyelesaikan pencarian dalam waktu di bawah 1 detik yaitu sekitar 0.36 sampai 0.77 detik. Hal tersebut karena algoritma ini hanya mengevaluasi satu tetangga per iterasi.
- Simulated Annealing juga relatif cepat dan konsisten dengan durasi sekitar 3 sampai 4 detik.
- Steepest Ascent Hill Climbing memiliki durasi yang cukup cepat atau moderat yaitu sekitar 4 sampai 9 detik.
- Sideways Move Hill Climbing memiliki durasi yang sangat bervariasi yaitu berkisar antara 6.5 hingga 36.6 detik. Jika cepat menemukan solusi 0, algoritma akan cepat. Tetapi, jika harus melakukan banyak sideways move atau terjebak dalam *local optima*, algoritma menjadi lambat.
- Genetic Algorithm memiliki durasi yang sangat bergantung pada ukuran populasi dan jumlah iterasi. Waktu yang dihasilkan bisa berkisar dari belasan hingga puluhan detik.
- Random Restart Hill Climbing adalah yang paling lambat secara konsisten yaitu antara 65 sampai 75 detik. Hal tersebut karena algoritma ini pada dasarnya menjalankan banyak kali proses Steepest Ascent.

3.1.5.4. Seberapa konsisten hasil akhir yang didapatkan dari tiap-tiap eksperimen yang dilakukan?

- Stochastic Hill Climbing dan Simulated Annealing keduanya sangat konsisten yaitu selalu menghasilkan penalti 0 dalam 3 run.
- Random Restart Hill Climbing juga cukup konsisten dalam menghasilkan solusi optimal atau sangat dekat dengan optimal (0, 2, 0).
- Steepest Ascent Hill Climbing tidak konsisten. Algoritma ini menghasilkan nilai penalti akhir yang berbeda-beda dan relatif tinggi di setiap run (278.5, 272.0, 219.0).
- Sideways Move Hill Climbing juga tidak konsisten (154.25, 0, 208.5).
- Genetic Algorithm terbukti tidak konsisten, terutama pada jumlah iterasi yang lebih rendah (hasil penalti 24, 45, 0 untuk 50 iterasi). Konsistensinya sedikit membaik dengan

iterasi lebih banyak (5, 0, 0 untuk 100 iterasi), namun satu run masih gagal mencapai 0.

3.1.5.5. Bagaimana pengaruh banyak iterasi dan jumlah populasi terhadap hasil akhir pencarian pada Genetic Algorithm?

Berdasarkan eksperimen yang telah dilakukan, jumlah iterasi dan populasi sama pentingnya untuk menemukan solusi yang optimal. Jumlah iterasi yang kecil dapat menyebabkan pencarian berhenti sebelum menemukan solusi yang matang. Sementara itu, iterasi yang banyak meningkatkan kemungkinan penemuan solusi optimal jika tidak terjebak dalam local optima. Di sisi lain, jumlah populasi yang terlalu kecil dapat menyebabkan algoritma terjebak pada solusi suboptimal. Sementara itu, jumlah populasi besar paling mungkin menemukan solusi optimal, tetapi membutuhkan waktu yang paling lama. Oleh karena itu, harus ada keseimbangan antara jumlah iterasi dan banyak populasi. Populasi yang lebih kecil akan memerlukan lebih banyak iterasi untuk menemukan solusi optimal. Sementara itu, populasi besar akan menemukan solusi bagus dalam jumlah iterasi lebih sedikit namun biaya per iterasinya akan lebih mahal.

BAB 3 KESIMPULAN DAN SARAN

3.1 Kesimpulan

Eksperimen ini menunjukkan bahwa untuk persoalan penjadwalan kelas yang NP-hard, pemilihan algoritma local search dan heuristik memiliki dampak yang sangat signifikan terhadap kualitas solusi, kecepatan, dan konsistensi.

1. Heuristik jumlah *neighbor* sangat berpengaruh pada proses pencarian untuk algoritma varian *Hill Climbing*. Penggunaan heuristik untuk membatasi max neighbor terbukti krusial. Tanpa heuristik ini, Hill Climbing menjadi tidak praktis karena waktu komputasi yang sangat lama. Ini adalah trade-off yang sangat menguntungkan karena sedikit penurunan akurasi pada setiap langkah ditukar dengan peningkatan kecepatan yang signifikan dan memungkinkan algoritma berjalan dalam waktu yang wajar.
2. Dalam menemukan solusi yang *global optima*, pentingnya suatu algoritma memiliki kemampuan untuk menghindari jebakan *local optima*. Algoritma yang sukses adalah yang memiliki mekanisme kuat untuk menghindari *local optimum*.
 - a. Simulated Annealing (SA) berhasil karena kemampuannya menerima solusi yang lebih buruk secara probabilistik sehingga bisa saja keluar dari jebakan local optimum.
 - b. Random Restart Hill Climbing berhasil dengan cara memulai pencarian dari banyak titik awal yang berbeda.
 - c. Stochastic Hill Climbing, meskipun sederhana, kecepatannya yang sangat tinggi memungkinkannya melakukan banyak sekali iterasi dalam waktu singkat, memberinya banyak kesempatan untuk "terlempar" ke jalur menuju solusi global optimum.
3. Algoritma yang sederhana cenderung mengalami kegagalan dalam mencapai global optima. Steepest Ascent Hill Climbing konsisten memberikan hasil terburuk. Sifatnya yang selalu harus memilih langkah terbaik membuatnya sangat mudah terjebak di local optimum pertama yang ditemuinya dan seringkali jauh dari solusi global.
4. Performa Algoritma Bervariasi:
 - a. Stochastic Hill Climbing dan Simulated Annealing memiliki performa yang jelas, karena keduanya secara konsisten mencapai solusi optimal (penalti 0) dengan cepat. Stochastic HC unggul dalam kecepatan, sementara SA unggul dalam keandalan teoritisnya.
 - b. Genetic Algorithm (GA) dan Sideways Move Hill Climbing menunjukkan hasil yang tidak stabil. Keduanya berpotensi menemukan solusi optimal, tetapi

keberhasilannya tidak dijamin di setiap percobaan dan sangat bergantung pada parameter (untuk GA) atau "medan" dari ruang solusi (untuk Sideways Move).

3.2 Saran

Berdasarkan kesimpulan di atas, berikut adalah rekomendasi untuk implementasi di masa depan atau untuk memilih algoritma terbaik untuk masalah serupa:

1. Pilihan utama:

Gunakan Stochastic Hill Climbing jika kecepatan adalah prioritas. Algoritma ini memberikan keseimbangan terbaik antara kualitas solusi dan waktu komputasi yang sangat singkat.

2. Pilihan alternatif:

Gunakan Simulated Annealing (SA) jika menginginkan algoritma yang andal dan konsisten dengan keyakinan kuat untuk keluar dari local optima. Kecepatannya sedikit di bawah Stochastic HC tetapi masih sangat cepat dan hasilnya terbukti sangat baik.

3. Gunakan dengan hati-hati:

- a. Genetic Algorithm (GA) sebaiknya digunakan jika memiliki waktu untuk melakukan eksperimen untuk menemukan parameter yang tepat (ukuran populasi, jumlah generasi, laju mutasi, dll.). Tanpa pengaturan yang baik, hasilnya bisa menjadi tidak optimal dan tidak konsisten.
- b. Random Restart Hill Climbing bisa menjadi pilihan jika waktu eksekusi bukanlah masalah. Namun, karena biayanya yang paling mahal, algoritma ini kurang praktis dibandingkan SA atau Stochastic HC.

4. Algoritma yang Sebaiknya Dihindari:

- a. Hindari Steepest Ascent Hill Climbing untuk masalah ini. Algoritma ini terlalu sederhana dan hampir selalu akan memberikan solusi yang buruk.
- b. Hill Climbing with Sideways Move juga kurang direkomendasikan karena kinerjanya yang tidak dapat diprediksi. Ada pilihan lain yang jauh lebih konsisten.

5. Menentukan Nilai Max Neighbor Yang Tepat

Jumlah max neighbor sangat berpengaruh pada hasil dari algoritma varian Hill Climbing. Dengan menambah atau mengurangi jumlah max neighbor, akan sangat menentukan neighbor yang dipilih oleh varian *Hill Climbing*. Jika jumlah neighbor terlalu sedikit, bisa memungkina varian Hill Climbing akan terjebak dalam *local optima*.

PEMBAGIAN TUGAS TIAP ANGGOTA KELOMPOK

No	Nama - NIM	Pembagian Tugas
1	Orvin Andika Ikhsan Abhista - 13523017	Laporan, Genetic Algorithm
2	Fajar Kurniawan - 13523027	Laporan, HC-SA, HC-SM, HC-RR
3	Reza Ahmad Syarif - 13523119	Laporan, HC-Stochastic, Simulated Annealing

REFERENSI

- [1] T. V. Mathew, "Genetic algorithm," Report, IIT Bombay, Mumbai, India, 2012
- [2] Materi Perkuliahan Artificial Intelligence ITB. Beyond Classical Search - Classical Vs Local Search
- [3] Materi Perkuliahan Artificial Intelligence ITB. Beyond Classical Search - Hill Climbing
- [4] Materi Perkuliahan Artificial Intelligence ITB. Beyond Classical Search - Simulated Annealing
- [5] Materi Perkuliahan Artificial Intelligence ITB. Beyond Classical Search - Genetic Algorithm