Algoritma Optimasi Terinspirasi Tumbuhan (Plant-Inspired Optimization Algorithms)

Muhammad Abiya Makruf (203012420034)

1. Nature Inspired Optimization Algorithm
   1.1. Apa Itu NIOA?

   Nature-Inspired Optimization Algorithms (NIOA) atau Algoritma Optimasi Terinspirasi Alam adalah sekumpulan teknik pemecahan masalah yang meniru strategi dan mekanisme yang ditemukan di alam untuk menyelesaikan masalah optimasi yang kompleks. Algoritma ini seringkali bersifat metaheuristik, yang berarti mereka tidak menjamin solusi optimal global tetapi bertujuan untuk menemukan solusi yang cukup baik dalam waktu yang wajar, terutama untuk masalah di mana metode eksak tidak praktis atau terlalu lambat. NIOA mengambil inspirasi dari berbagai fenomena alam, mulai dari perilaku kawanan hewan, proses evolusi biologis, hingga prinsip-prinsip fisika. Keunggulan utama NIOA adalah kemampuannya untuk menangani masalah optimasi berskala besar, non-linear, dan memiliki banyak variabel tanpa memerlukan pengetahuan mendalam tentang struktur matematis masalah tersebut. Algoritma ini biasanya melibatkan proses iteratif di mana solusi-solusi potensial dievaluasi dan ditingkatkan berdasarkan prinsip-prinsip yang ditiru dari alam.

   1.2. Pembagian Secara Umum

   NIOA dapat diklasifikasikan berdasarkan sumber inspirasinya. Pembagian ini membantu dalam memahami mekanisme dasar dan area aplikasi potensial dari masing-masing algoritma. Beberapa kategori utama meliputi:

   - Terinspirasi Biologi (Biology-Inspired)
   - Terinspirasi Fisika dan Kimia (Physics and Chemistry-Inspired)
   - Terinspirasi Perilaku Manusia (Human Behavior-Inspired)
   - Lain-lain: Ada juga kategori lain seperti yang terinspirasi dari musik, seni, atau algoritma hibrida yang menggabungkan beberapa inspirasi.

2. Plant (Non-Fungi) Algorithm
   2.1. Penjelasan

   Algoritma yang terinspirasi dari tumbuhan (Plant-Inspired Algorithms) merupakan cabang dari Nature-Inspired Optimization Algorithms (NIOA) yang relatif baru dibandingkan dengan algoritma yang terinspirasi dari hewan atau serangga. Ide dasarnya adalah meniru berbagai aspek dari "kecerdasan" dan strategi bertahan hidup tumbuhan dalam mencari sumber daya, berkembang biak, dan beradaptasi dengan lingkungannya. Tumbuhan, meskipun tidak dapat bergerak secara mandiri seperti hewan, menunjukkan kemampuan adaptasi yang luar biasa terhadap kondisi lingkungan yang beragam melalui mekanisme pertumbuhan, penyebaran biji, dan pengembangan sistem akar. Algoritma-algoritma ini mencoba memodelkan proses-proses tersebut secara matematis untuk diterapkan dalam penyelesaian masalah optimasi. Mereka seringkali berfokus pada bagaimana tumbuhan menjelajahi ruang

untuk menemukan lokasi yang paling menguntungkan bagi pertumbuhan dan kelangsungan hidupnya.

2.2. Algoritma

Berikut adalah beberapa contoh algoritma yang terinspirasi dari tumbuhan beserta inspirasi utamanya:

2.2.1. Plant Propagation Algorithm

- Inspirasi: Algoritma ini terinspirasi dari cara tumbuhan, khususnya stroberi, berkembang biak dan menyebar menggunakan tunas atau stolon (runners).
- Mekanisme: PPA memodelkan bagaimana tanaman induk mengirimkan tunas untuk mencari lokasi yang lebih baik untuk tumbuh. Tanaman yang berada di lokasi yang baik (memiliki fitness tinggi) akan menghasilkan banyak tunas pendek untuk mengeksploitasi area sekitarnya (intensifikasi). Sebaliknya, tanaman yang berada di lokasi kurang baik (fitness rendah) akan menghasilkan lebih sedikit tunas tetapi dengan jangkauan yang lebih panjang untuk menjelajahi area baru (eksplorasi). Kualitas lokasi direpresentasikan oleh nilai fungsi objektif dari masalah optimasi yang ingin diselesaikan.

2.2.2. Invansive Weed Optimization

- Inspirasi: Algoritma IWO terinspirasi dari perilaku kolonisasi gulma invasif di suatu lahan pertanian. Gulma dikenal karena kemampuannya menyebar dengan cepat, beradaptasi, dan mendominasi suatu area.
- Mekanisme: IWO memodelkan proses penyebaran benih oleh gulma. Setiap gulma (solusi) menghasilkan sejumlah benih (solusi baru) berdasarkan tingkat adaptasinya (fitness) terhadap lingkungan. Gulma yang lebih adaptif menghasilkan lebih banyak benih. Benih-benih ini kemudian disebarkan secara acak di sekitar gulma induk, dengan jangkauan penyebaran (standar deviasi) yang umumnya menurun seiring berjalannya iterasi (meniru kondisi di mana ruang menjadi lebih terbatas). Jika jumlah total gulma melebihi kapasitas maksimum lahan, maka gulma dengan adaptabilitas terendah akan tereliminasi (prinsip "survival of the fittest").

2.2.3. Root Growth Algorithm

- Inspirasi: RGO terinspirasi dari perilaku pertumbuhan adaptif sistem akar tumbuhan dalam mencari air dan nutrisi di dalam tanah. Akar tumbuhan menunjukkan plastisitas yang tinggi dalam merespons kondisi tanah, seperti menghindari rintangan dan menjelajahi area kaya nutrisi.
- Mekanisme: Dalam RGO, setiap ujung akar (root apex) merepresentasikan sebuah solusi potensial. Ujung akar ini dapat tumbuh memanjang, menghasilkan cabang akar baru (akar lateral atau anak), atau berhenti tumbuh. Akar dibagi menjadi beberapa kelompok berdasarkan fitnessnya, seperti akar utama (main roots), akar lateral (lateral roots), dan akar yang

menua (aging roots). Akar utama, yang memiliki fitness terbaik, akan tumbuh memanjang ke arah posisi lokal terbaik dan kemudian bercabang (monopodial branching). Akar lateral mungkin tumbuh dengan menggantikan ujung akar asli (sympodial branching). Terdapat juga mekanisme inhibisi yang terinspirasi hormon tumbuhan untuk mencegah pertumbuhan eksplosif di satu area dan terjebak di optimum lokal. Salah satu karakteristik penting yang dimodelkan adalah propagasi dengan kemiripan diri (self-similarity) dari sistem akar.

2.2.4. Tree Seed Algorithm

- Inspirasi: TSA didasarkan pada hubungan antara pohon dan biji mereka dalam proses reproduksi dan penyebaran. Pohon menyebar ke permukaan melalui biji-biji mereka, yang kemudian tumbuh menjadi pohon baru.
- Mekanisme: Dalam TSA, lokasi pohon dan biji dalam ruang pencarian n-dimensi merepresentasikan solusi yang mungkin untuk masalah optimasi. Setiap pohon (solusi saat ini) menghasilkan satu atau lebih biji (solusi kandidat baru). Lokasi biji baru ini ditentukan berdasarkan lokasi pohon induknya dan salah satu dari dua strategi: (1) bergerak menuju lokasi pohon terbaik yang ditemukan sejauh ini dalam populasi, atau (2) bergerak relatif terhadap lokasi pohon lain yang dipilih secara acak dari populasi. Pemilihan antara dua strategi ini dikendalikan oleh parameter yang disebut *Search Tendency* (ST). ST bertujuan untuk menyeimbangkan kemampuan eksplorasi (pencarian global) dan eksploitasi (pencarian lokal) dari algoritma. Jika lokasi biji yang dihasilkan lebih baik daripada pohon induknya, maka biji tersebut akan menggantikan pohon induknya.

2.3. Alur Umum

Meskipun setiap algoritma memiliki inspirasi dan mekanisme detail yang unik, terdapat beberapa kemiripan alur umum dalam cara kerja keempat algoritma yang terinspirasi tumbuhan ini:

- Inisialisasi Populasi
- Evaluasi Fitness
- Mekanisme Reproduksi/Propagasi (Generasi Solusi Baru)
- Seleksi dan/atau Penggantian
- Iterasi
- Keseimbangan Eksplorasi-Eksploitasi

3. Permasalahan

3.1. Vertex Cover

Permasalahan Vertex Cover adalah salah satu masalah optimasi klasik dalam teori graf dan ilmu komputer. Diberikan sebuah graf tidak berarah G=(V,E), di mana V adalah himpunan simpul (nodes) dan E adalah himpunan sisi (edges). Tujuan dari Minimum Vertex Cover Problem adalah untuk menemukan Vertex Cover dengan ukuran minimum, yaitu himpunan V yang memiliki jumlah simpul paling sedikit namun tetap menutupi semua sisi dalam graf. Permasalahan ini diketahui NP-hard untuk graf

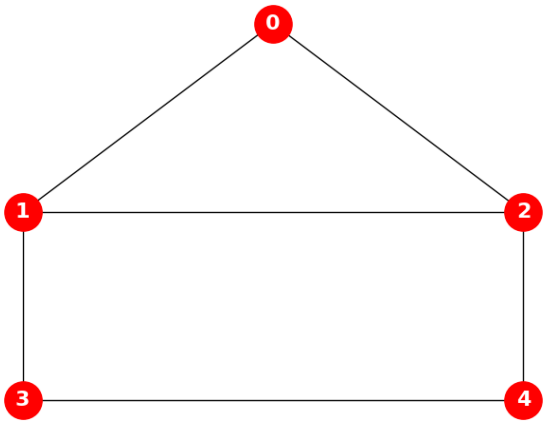
umum, yang berarti tidak ada algoritma waktu polinomial yang diketahui dapat menemukan solusi optimal secara pasti untuk semua kasus, kecuali P=NP. Oleh karena itu, algoritma heuristik dan aproksimasi seperti PPA dan IWO sering digunakan untuk mencari solusi yang baik (mendekati optimal) dalam waktu yang wajar.

3.2. Contoh Input

Berikut adalah contoh format isi file input.dat untuk graf dengan 5 node:

Isi Input.dat
5
0 1
0 2
1 2
1 3
2 4
3 4

- Baris pertama (5) adalah jumlah node dalam graf (node diindeks dari 0 hingga 4).
- Baris-baris berikutnya masing-masing mendefinisikan sebuah sisi, dengan dua angka yang merepresentasikan node yang dihubungkan oleh sisi tersebut.



Gambar 1. Visualisasi Graf.

3.3. Contoh Solusi

Untuk graf contoh di atas, mari kita analisis beberapa kemungkinan Vertex Cover dan ukurannya:

- Graf: Node={0,1,2,3,4}, Edge={(0,1), (0,2), (1,2), (1,3), (2,4), (3,4)}
- Solusi Kandidat 1: {0, 1, 2, 3, 4} (Semua node)
  - Ukuran: 5
  - Valid? Ya, semua edge pasti tertutup.
- Solusi Kandidat 2: {0, 1, 4}
  - Ukuran: 3
  - Cek edge:
    - (0,1): Tertutup (oleh 0 atau 1)
    - (0,2): Tertutup (oleh 0)

        - (1,2): Tertutup (oleh 1)
        - (1,3): Tertutup (oleh 1)
        - (2,4): Tertutup (oleh 4)
        - (3,4): Tertutup (oleh 4)
    - Valid? Ya.
- Solusi Kandidat 3: {1, 2, 3}
- Solusi Kandidat 4 (Optimal): {1, 2, 4}
- Dan seterusnya.

Dalam contoh ini, ukuran Vertex Cover minimum adalah 3. Ada beberapa himpunan node berukuran 3 yang merupakan Vertex Cover yang valid (misalnya, {1,2,4}, {0,2,3}). Algoritma PPA atau IWO akan mencoba menemukan salah satu dari solusi optimal ini. Representasi solusi dalam algoritma (misalnya, untuk {1,2,4}): [false, true, true, false, true]

4. Detail Algoritma PPA dan IWO Untuk Menyelesaikan Vertex Cover

Implementasi pseudocode menggunakan bahasa c++ dapat dilihat di lampiran.

4.1. Pseudocode PPA

Algorithm PPA_VertexCover(Graph G)

Input:

G = undirected graph with nodes and edges

Output:

best_solution = best plant found during evolution

Begin

Initialize population with random solutions

Set best_solution with very high fitness value

If graph is empty, return best_solution

For generation = 1 to MAX_GENERATIONS do

Evaluate fitness of all plants in population

Sort population by ascending fitness

Update best_solution if better plant is found

Normalize and map fitness for runner generation

Initialize combined_population with current population

For each plant in population do

Determine number of runners based on mapped fitness

For each runner do

Copy solution from parent plant

Modify runner using mapped distance and sigmoid

Evaluate fitness of runner

Add runner to combined_population

End for

End for

Sort combined_population by fitness

Select top individuals as new population

End for

If best_solution fitness not evaluated, evaluate it

Return best_solution

End

4.2.Pseudocode IWO

Algorithm IWO_VertexCover(Graph G)

Input:

G = undirected graph with nodes and edges

Output:

best_solution = best weed found

Begin

Initialize population with random solutions

Set best_solution with very high fitness

If graph is empty, return best_solution

For iteration = 1 to MAX_ITERATIONS do

Evaluate fitness of all weeds in population

If population is empty, continue

Find min and max fitness in current population

Initialize combined_population with current population

For each weed in population do

Determine number of seeds to produce based on fitness

Compute current sigma value (σ_iter)

For each seed to produce do

Copy solution from parent weed

Apply Gaussian perturbation and sigmoid to each bit

Evaluate fitness of new seed

Add seed to combined_population

End for

End for

Sort combined_population by fitness

Select best individuals up to max population size

Replace current population with selected individuals

Update best_solution if better individual found

End for

If best_solution not updated, assign from population or compute fitness manually

Return best_solution

End

5. Struktur Data

Berdasarkan kode C++ yang disajikan pada poin 4, berikut adalah detail struktur data yang digunakan untuk PPA dan IWO:

5.1. Struktur Data Umum (Digunakan oleh Kedua Algoritma)

- **pair<int, int>**: Digunakan dalam struct Graph dan struct Graph_IWO untuk merepresentasikan sebuah sisi (edge). first dan second adalah integer yang menunjukkan nomor node yang dihubungkan oleh sisi tersebut.
- **bool array_nama[UKURAN_MAKSIMUM]**: Array C-style dengan tipe data boolean digunakan untuk merepresentasikan solusi Vertex Cover dalam struct Plant dan struct Weed_IWO (yaitu, solution_nodes). UKURAN_MAKSIMUM adalah konstanta seperti MAX_NODES_CONST. Nilai true pada indeks i menandakan node i termasuk dalam cover, dan false jika tidak.
- **Array C-style dari struct (misal, Plant population[POPULATION_SIZE])**: Digunakan untuk menyimpan populasi tanaman (PPA) atau gulma (IWO). Ini adalah array dengan ukuran tetap yang elemen-elemennya adalah objek dari struct Plant atau Weed_IWO.
- **Variabel Primitif**: int untuk jumlah node, jumlah edge aktual, ukuran cover, jumlah edge tidak tertutup, iterator loop, dll. double untuk nilai fitness dan parameter internal algoritma.

5.2. Algoritma PPA

- Struct Graph
  - int num_nodes: Menyimpan jumlah node.
  - pair<int, int> edges[MAX_EDGES_CONST]: Array C-style untuk menyimpan sisi-sisi graf.
  - int actual_num_edges: Menyimpan jumlah sisi aktual.
- Struct Plant
  - bool solution_nodes[MAX_NODES_CONST]: Array C-style representasi solusi.
  - double raw_fitness, normalized_fitness_f, mapped_fitness_N: Menyimpan berbagai representasi fitness.
  - int cover_size, int uncovered_edges_count: Informasi terkait kualitas solusi.

- **Plant population[POPULATION_SIZE]**: Array C-style untuk populasi utama.
- **Plant combined_population[MAX_COMBINED_SIZE]**: Array C-style sementara untuk menampung tanaman induk dan *runner* baru sebelum proses seleksi. MAX_COMBINED_SIZE adalah POPULATION_SIZE + POPULATION_SIZE * MAX_RUNNERS_PER_PLANT.

5.3. Algoritma IWO

- Struktur Graph
  - int num_nodes
  - pair<int, int> edges[MAX_EDGES_CONST_IWO]
  - int actual_num_edges
- Struktur Weed
  - bool solution_nodes[MAX_NODES_CONST_IWO]: Array C-style representasi solusi.
  - double fitness: Menyimpan nilai fitness.
  - int cover_size, int uncovered_edges_count: Informasi terkait kualitas solusi.
  - Memiliki operator< yang di-*overload* untuk pengurutan berdasarkan fitness.
- **Weed_IWO population[MAX_POPULATION_SIZE_IWO]**: Array C-style untuk populasi gulma. Ukuran aktual yang digunakan dilacak oleh variabel current_population_size.
- **Weed_IWO combined_population[MAX_COMBINED_WEEDS]**: Array C-style sementara untuk menampung gulma induk dan benih baru sebelum eliminasi kompetitif. MAX_COMBINED_WEEDS adalah MAX_POPULATION_SIZE_IWO + MAX_POPULATION_SIZE_IWO * MAX_SEEDS_PER_WEED_IWO.

6. Kompleksitas

Analisis kompleksitas berikut didasarkan pada implementasi C++ menggunakan array C-style, untuk menyelesaikan masalah Vertex Cover.

Notasi yang Digunakan:

G_{max}: Jumlah generasi maksimum (untuk PPA) atau iterasi maksimum (untuk IWO).

N_{pop}: Ukuran populasi tetap untuk PPA.

N_{init}: Ukuran populasi awal untuk IWO.
$N_{\max\ \_pop}$: Ukuran populasi maksimum untuk IWO.
R_{max}: Jumlah *runner* atau benih maksimum per individu.
M: Jumlah node dalam graf.
E: Jumlah edge aktual dalam graf.

6.1. Kompleksitas Fungsi Fitness (Untuk PPA dan IWO)

- Menghitung cover_size : $O(M)$
- Menghitung uncovered_edges: $O(E)$
- Total: $O(M + E)$

6.2.PPA

Kompleksitas per Generasi

6.2.1. Evaluasi Fitness

- Ada N_{pop} tanaman
- Kompleksitas: $N_{pop} \times\ O(M + E)$

6.2.2. Pengurutan Populasi

- Mengurutkan N_{pop} tanaman
- Kompleksitas: $O(N_{pop} \log N_{pop})$

6.2.3. Normalisasi

- Iterasi sebanyak N_{pop} tanaman
- Kompleksitas: $O(N_{pop})$

6.2.4. Produksi Runner

- Outer loop untuk N_{pop} tanaman
- Inner loop untuk maksimal R_{max}runner per tanaman
- Setiap runner dihitung fitness nya: $O(M + E)$
- Total runner yang mungkin dihasilkan adalah $O(N_{pop} \times R_{max})$
- Kompleksitas produksi dan evaluasi: $N_{pop} \times R_{max} \times\ O(M + E)$

6.2.5. Pengurutan

- Ukuran combined_population maksimal adalah $N_{pop} + N_{pop}\ \times\ R_{max}$. Sebuh $S_{comb=}\ N_{pop}(1 + R_{max})$
- Kompleksitas: $O(S_{comb} \log S_{comb})$

6.2.6. Seleksi Generasi

- Menyalin N_{pop} elemen dari combined_population yang sudah terurut

- Kompleksitas: $O(N_{pop})$

Kompleksitas total PPA:

$G_{max} \times O(N_{pop}(M + E) + N_{pop} \log N_{pop})$

6.3. IWO

Kompleksitas per Iterasi

6.3.1. Evaluasi Fitness Populasi

- Ada N_{curr} gulma
- Kompleksitas: $N_{curr} \times O(M + E)$

6.3.2. Menemukan Fitness Min/Max

- Iterasi N_{curr} gulma
- Kompleksitas: $O(N_{curr})$

6.3.3. Reproduksi Benih dan Pengisian

- Outer loop untuk N_{curr} gulma
- Inner loop untuk maksimal R_{max} benih per gulma
- Setiap benih dihitung fitness nya: $O(M + E)$
- Total benih yang mungkin dihasilkan adalah $N_{curr} \times R_{max}$
- Kompleksitas produksi dan evaluasi fitness benih: $N_{curr} \times R_{max} \times (O(M + E)$
- Menyalin populasi awal ke combines_population: $O(N_{curr})$
- Menambahkan benih ke combined_population: $O(1)$

6.3.4. Pengurutan

- Ukuran combined_population maksimal adalah $N_{curr} + N_{curr} \times R_{max}$. Sebut $S_{comb\_iwo} = N_{curr}(1 + R_{max})$
- Kompleksitas: $O(S_{comb_{iwo}} \log S_{comb\_iwo})$

6.3.5. Eliminasi Kompetitif

- Menyalin maksimal $N_{\max\ _pop}$elemen dari combined_population yang sudah terurut.
- Kompleksitas: $O(N_{\max\ \_pop})$

Kompleksitas Total IWO:

$G_{max} \times O(N_{maxpop}(M + E) + N_{maxpop} \log N_{maxpop})$

7. Perbandingan PPA dan IWO

7.1. Representasi Solusi Individu

Perbandingan: Keduanya menggunakan pendekatan yang sama untuk representasi solusi inti, yaitu array boolean 1 dimensi. PPA menyimpan

lebih banyak jenis nilai fitness karena mekanisme internalnya yang berbeda dalam menentukan jumlah dan "jarak" runner.

7.2. Representasi Graf

Perbandingan: Keduanya menggunakan struktur yang sama (array 1 dimensi dari pair) untuk menyimpan informasi graf. Tidak ada perbedaan signifikan dalam hal ini.

7.3. Manajemen Populasi

Perbandingan: PPA cenderung bekerja dengan ukuran populasi yang lebih stabil (POPULATION_SIZE), sedangkan IWO memiliki mekanisme di mana ukuran populasi bisa membengkak sementara karena produksi benih sebelum dipangkas kembali ke MAX_POPULATION_SIZE_IWO. Keduanya menggunakan array sementara yang lebih besar untuk mengelola penggabungan dan seleksi, yang merupakan praktik umum saat bekerja dengan array berukuran tetap.

7.4. Pengurutan (Sorting)

Perbandingan: Keduanya mengandalkan std::sort dari C++ Standard Library untuk pengurutan, yang merupakan pilihan yang efisien dan standar. Cara mereka menyediakan kriteria pengurutan sedikit berbeda (fungsi pembanding eksternal untuk PPA vs. operator yang di-*overload* untuk IWO), tetapi ini lebih ke gaya implementasi daripada perbedaan fundamental dalam struktur data atau algoritma pengurutan yang digunakan.

8. Kelebihan dan Kekurangan PPA dan IWO

Berikut adalah analisis kelebihan dan kekurangan dari Plant Propagation Algorithm (PPA) dan Invasive Weed Optimization (IWO), khususnya dalam konteks penerapan pada masalah optimasi seperti Vertex Cover.

Kelebihan	
Plant Propagation Algorithm (PPA)	Invasive Weed Optimization (IWO)
Keseimbangan Eksplorasi-Eksploitasi yang Intuitif: Mekanisme dimana tanaman "baik" menghasilkan banyak *runner* pendek (eksploitasi) dan tanaman "kurang baik" menghasilkan sedikit *runner* panjang (eksplorasi) menyediakan cara yang cukup alami untuk menyeimbangkan pencarian.	**Transisi Eksplorasi ke Eksploitasi yang Mulus:** Pengurangan standar deviasi (σ) secara non-linear seiring iterasi memungkinkan algoritma secara alami beralih dari fase eksplorasi global (saat σ besar) ke fase eksploitasi lokal (saat σ kecil).

Konsep Sederhana: Inspirasi dari penyebaran stroberi relatif mudah dipahami, yang dapat membantu dalam implementasi dan pemahaman algoritma.	**Mekanisme Seleksi yang Kuat:** Eliminasi kompetitif, di mana hanya individu terbaik yang bertahan jika populasi melebihi batas maksimum, memberikan tekanan seleksi yang kuat menuju solusi yang lebih baik.
Adaptif terhadap Kualitas Solusi: Jumlah dan "jarak" runner secara langsung dipengaruhi oleh kualitas (fitness) dari solusi induk, memungkinkan adaptasi dinamis dari strategi pencarian.	**Reproduksi Diferensial:** Jumlah benih yang dihasilkan sebanding dengan fitness gulma, yang berarti solusi yang lebih baik memiliki kesempatan lebih besar untuk menyebarkan karakteristiknya.

Kekurangan	
Plant Propagation Algorithm (PPA)	Invasive Weed Optimization (IWO)
Adaptasi ke Domain Diskrit/Biner: PPA terinspirasi dari proses kontinu (pertumbuhan tunas dengan jarak). Adaptasinya ke masalah diskrit seperti Vertex Cover memerlukan mekanisme tambahan (misalnya, penggunaan fungsi sigmoid untuk mengonversi nilai kontinu ke biner) yang mungkin terasa kurang "alami" dibandingkan inspirasi aslinya dan dapat menambah kompleksitas.	**Konvergensi Prematur: Jika standar deviasi (σ) menurun terlalu cepat, atau jika satu solusi yang sangat baik mendominasi produksi benih terlalu dini, algoritma berisiko mengalami konvergensi prematur ke optimum lokal.**
Sensitivitas Parameter: Kinerja PPA bisa sensitif terhadap parameter seperti MAX_RUNNERS_PER_PLANT, cara "jarak" dihitung dan diterapkan, serta parameter normalisasi dan pemetaan fitness. Penyetelan parameter ini mungkin memerlukan eksperimen.	**Sensitivitas Parameter: Parameter seperti laju penurunan σ (NONLINEAR_MODULATION_INDEX_N, SIGMA_INITIAL, SIGMA_FINAL), serta batas minimum dan maksimum benih (MIN_SEEDS_PER_WEED, MAX_SEEDS_PER_WEED) dapat sangat mempengaruhi kinerja dan memerlukan penyetelan.**

Potensi Stagnasi: Jika semua tanaman dalam populasi memiliki fitness yang mirip atau jika mekanisme "jarak pendek" terlalu dominan tanpa mekanisme diversifikasi yang cukup kuat, algoritma bisa terjebak dalam optimum lokal.	**Adaptasi Dispersi Spasial ke Domain Biner: Mirip dengan PPA, mekanisme dispersi benih (yang menggunakan distribusi normal kontinu) perlu diadaptasi ke domain biner. Metode seperti penggunaan fungsi sigmoid adalah salah satu cara, tetapi mungkin bukan satu-satunya atau yang paling optimal untuk semua masalah biner.**
	Komputasi Fitness Berulang: Dalam beberapa implementasi, fitness untuk individu yang sama mungkin dihitung berulang kali jika tidak ada mekanisme caching atau jika individu tersebut bertahan tanpa perubahan melewati beberapa tahap.

9. Manfaat Algoritma PPA dan IWO dalam Menyelesaikan Permasalahan Optimasi

   Algoritma PPA dan IWO, sebagai bagian dari keluarga besar Nature-Inspired Optimization Algorithms (NIOA), menawarkan sejumlah manfaat dalam menyelesaikan berbagai permasalahan optimisasi, termasuk masalah kompleks seperti Vertex Cover:

   9.1. Kemampuan Mengatasi Masalah Kompleks (NP-hard):

   Baik PPA maupun IWO dirancang sebagai metaheuristik yang dapat memberikan solusi berkualitas baik (mendekati optimal) untuk masalah-masalah di mana pencarian solusi optimal secara eksak tidak praktis karena kompleksitas waktu yang sangat tinggi (misalnya, NP-hard seperti Vertex Cover). Mereka tidak terjebak oleh asumsi linearitas atau diferensiabilitas fungsi objektif.

   9.2. Pendekatan Berbasis Populasi:

   Kedua algoritma memelihara sekumpulan solusi (populasi) secara bersamaan. Ini memungkinkan eksplorasi yang lebih luas terhadap ruang pencarian dibandingkan dengan metode pencarian titik tunggal, sehingga mengurangi risiko terjebak dalam optimum lokal yang buruk.

   9.3. Keseimbangan Bawaan antara Eksplorasi dan Eksploitasi:

   - **PPA:** Secara eksplisit menyeimbangkan melalui mekanisme *runner* pendek (eksploitasi solusi yang baik) dan *runner* panjang (eksplorasi dari solusi yang kurang baik).

- **IWO:** Mencapai keseimbangan ini secara dinamis melalui penurunan standar deviasi penyebaran benih (dari eksplorasi ke eksploitasi) dan melalui reproduksi diferensial (solusi yang lebih baik lebih banyak bereproduksi).

9.4. Tidak Memerlukan Informasi Gradien:
PPA dan IWO adalah metode optimasi *black-box*, artinya mereka hanya memerlukan evaluasi fungsi objektif (fitness) untuk memandu pencarian. Mereka tidak memerlukan turunan atau informasi gradien dari fungsi objektif, sehingga cocok untuk masalah dengan fungsi objektif yang non-diferensiabel, diskontinu, atau berisik (*noisy*).

9.5. Adaptabilitas dan Fleksibilitas:
Konsep dasar PPA dan IWO relatif fleksibel dan dapat diadaptasi untuk berbagai jenis masalah optimasi, baik kontinu maupun diskrit/biner (dengan modifikasi yang sesuai pada representasi solusi dan operator pencarian). Contohnya adalah adaptasi untuk masalah Vertex Cover yang telah dibahas.

9.6. Robustness terhadap Kondisi Awal:
Karena sifat stokastik dan pencarian berbasis populasi, algoritma ini umumnya kurang sensitif terhadap pilihan solusi awal dibandingkan dengan metode optimasi lokal berbasis gradien. Mereka memiliki kemampuan untuk melepaskan diri dari solusi awal yang buruk.

9.7. Inspirasi dari Alam yang Memberikan Perilaku Pencarian Unik:
Mekanisme yang terinspirasi dari strategi bertahan hidup dan reproduksi tumbuhan dapat menghasilkan perilaku pencarian yang berbeda dari algoritma yang terinspirasi dari kawanan hewan atau fisika, yang mungkin lebih cocok untuk struktur lanskap pencarian tertentu.

9.8. Potensi Paralelisasi:
Banyak langkah dalam PPA dan IWO, seperti evaluasi fitness untuk setiap individu dalam populasi atau generasi *runner*/benih dari setiap induk, dapat dilakukan secara independen. Hal ini membuka peluang untuk implementasi paralel yang dapat secara signifikan mengurangi waktu komputasi pada sistem multi-core atau terdistribusi.

10. Kesimpulan
Laporan ini telah membahas secara komprehensif mengenai Algoritma Optimasi Terinspirasi Alam (Nature-Inspired Optimization Algorithms - NIOA), dengan fokus khusus pada sub-kategori yang relatif baru dan menarik, yaitu Algoritma Terinspirasi Tumbuhan (Plant (Non-Fungi)

Inspired Algorithms). Telah dijelaskan bahwa NIOA merupakan pendekatan metaheuristik yang kuat untuk menyelesaikan masalah optimasi kompleks dengan meniru berbagai fenomena dan strategi cerdas yang ada di alam.

Dari berbagai algoritma yang terinspirasi tumbuhan, laporan ini mendalami empat contoh utama: Plant Propagation Algorithm (PPA), Invasive Weed Optimization (IWO), Root Growth Algorithm (RGO), dan Tree-Seed Algorithm (TSA). Meskipun masing-masing memiliki inspirasi dan mekanisme unik—mulai dari penyebaran tunas stroberi (PPA), kolonisasi gulma invasif (IWO), pertumbuhan adaptif akar (RGO), hingga hubungan pohon dan biji (TSA) —mereka semua berbagi alur kerja umum yang melibatkan inisialisasi populasi, evaluasi fitness, generasi solusi baru melalui mekanisme reproduksi atau propagasi yang terinspirasi dari tumbuhan, dan proses seleksi atau pembaruan.

Dua algoritma, yaitu Plant Propagation Algorithm (PPA) dan Invasive Weed Optimization (IWO), dipilih untuk analisis dan implementasi lebih lanjut dalam menyelesaikan permasalahan Vertex Cover, sebuah masalah optimasi NP-hard klasik dalam teori graf. Implementasi menggunakan bahasa C++ dengan struktur data array C-style menunjukkan bahwa kedua algoritma ini, meskipun seringkali terinspirasi dari proses yang bersifat kontinu di alam, dapat diadaptasi secara efektif untuk menangani masalah diskrit atau biner. Proses adaptasi ini melibatkan perancangan representasi solusi yang sesuai (array boolean untuk Vertex Cover) dan mekanisme untuk menerjemahkan "pergerakan" atau "penyebaran" ke dalam perubahan pada solusi biner tersebut, misalnya melalui fungsi sigmoid setelah adanya gangguan atau modifikasi.

Melalui contoh ilustratif dengan graf 5 node, telah ditunjukkan langkah demi langkah bagaimana PPA dengan mekanisme runner-nya dan IWO dengan mekanisme penyebaran benih serta eliminasi kompetitifnya menjelajahi ruang solusi untuk mencari Vertex Cover dengan ukuran minimal. Kedua algoritma menunjukkan kemampuan untuk menyeimbangkan antara eksplorasi (pencarian di area baru) dan eksploitasi (pemfokusan pada area yang menjanjikan). PPA melakukannya melalui variasi panjang dan jumlah runner, sementara IWO mencapainya melalui dinamika standar deviasi penyebaran benih dan tekanan seleksi dari eliminasi kompetitif.

Analisis kompleksitas waktu per generasi/iterasi untuk PPA dan IWO pada masalah Vertex Cover menunjukkan bahwa kedua algoritma memiliki kompleksitas yang sebanding, umumnya didominasi oleh evaluasi fitness dari populasi dan individu baru yang dihasilkan, serta proses pengurutan. Dengan parameter populasi yang relatif kecil, kedua algoritma ini menawarkan pendekatan yang menjanjikan untuk menemukan solusi berkualitas baik dalam waktu yang wajar untuk masalah yang sulit seperti Vertex Cover.

Secara keseluruhan, algoritma PPA dan IWO, sebagai representasi dari Plant-Inspired Algorithms, menunjukkan potensi besar sebagai alat optimasi yang efektif dan adaptif. Keunikan inspirasi mereka dari dunia tumbuhan membuka jalan untuk pengembangan teknik-teknik baru yang mungkin memiliki keunggulan dalam mengatasi jenis-jenis masalah optimasi tertentu. Studi lebih lanjut dan aplikasi pada domain masalah yang lebih luas akan terus memperkaya pemahaman kita tentang kekuatan dan keterbatasan dari pendekatan optimasi yang menarik ini.

Daftar Pustaka

Lampiran

No	Nama	Tautan
	Implementasi C++ PPA	
	Implementasi C++ PPA	
	Contoh Input	
	Slide Presentasi	