OPTIMALISASI PEMILIHAN URUTAN SYUTING *SCENE* FILM MENGGUNAKAN ALGORITMA *PARTICLE SWARM OPTIMIZATION*
(Studi Kasus: Film ”Ketika Adzan Sudah Tidak Lagi Berkumandang”)

DRAF SKRIPSI

Langgeng Prassadewo Sukma Adi Winoto Basla
NIM. 1807065014

PROGRAM STUDI S1 MATEMATIKA
JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS MULAWARMAN
SAMARINDA
2025

OPTIMALISASI PEMILIHAN URUTAN SYUTING *SCENE* FILM MENGGUNAKAN ALGORITMA *PARTICLE SWARM OPTIMIZATION*
(Studi Kasus: Film ”Ketika Adzan Sudah Tidak Lagi Berkumandang”)

DRAF SKRIPSI

Diajukan kepada
Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Mulawarman untuk memenuhi sebagai persyaratan
memperoleh gelar Sarjana Matematika

Oleh:
Langgeng Prassadewo Sukma Adi Winoto Basla
NIM. 1807065014

PROGRAM STUDI S1 MATEMATIKA
JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS MULAWARMAN
SAMARINDA
2025

HALAMAN PENGESAHAN

PRAKATA

Bismillahirahmanirrahim

Assalamuálaikum Warahmatullahi Wabarakatuh

Segala puji dan syukur penulis panjatkan kepada Allah SWT., atas segala limpahan rahmat, karunia dan hidayah-Nya khususnya kepada penulis, sehingga penulis dapat menulis draf skripsi ini yang berjudul ”Optimalisasi Pemilihan Urutan Syuting *Scene* Film Menggunakan Algoritma *Particle Swarm Optimization* (Studi Kasus: Film ”Ketika Adzan Sudah Tidak Lagi Berkumandang”)”.

Pada kesempatan ini, penulis mengucapkan terima kasih atas bantuan dan bimbingan dalam penyusunan draf skripsi ini, terutama kepada yang terhormat:

1. Ibu Dr. Dra. Hj. Ratna Kusuma, M.Si., selalu Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Mulawarman, Samarinda.
2. Bapak Wasono, S.Si., M.Si., selaku Dosen Pembimbing I dan Bapak Fidia Deny Tisna Amijaya, S.Si., M.Si., selaku Dosen Pembimbing II.
3. Bapak Dr. Syaripuddin, S.Si., M.Si., selaku Dosen Penguji I dan Bapak Andri Azmul Fauzi, S.Si., M.Si., selaku Dosen Penguji II.
4. Ketiga orang tua tercinta dan tersayang, Bapak Harwanoto, Ibu Nor Handayani, dan Ibu Anik Suryani, serta adik saya, Liliana Devita Sari yang selalu memberikan doa, dukungan, semangat, dan kasih sayang selama proses pembuatan skripsi ini.
5. Bang David Richard selaku Sutradara sekaligus Produser dan Bang Fayed selaku Asisten Sutradara I yang telah mengizinkan penulis menggunakan karya Film ”Ketika Adzan Sudah Tidak Lagi Berkumandang” dalam tugas akhir skripsi.
6. Dana, Mahatir, dan Hafif serta seluruh kru produksi film ”Ketika Adzan Sudah Tidak Lagi Berkumandang”.
7. Tim dosen serta staf Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Mulawarman, terutama yang berada di Program Studi S1 Matematika.
8. Teman-teman Matematika angkatan 2018 yang telah memberikan dukungan, semangat, dan motivasi hingga selesainya proses penyusunan draf skripsi ini.

9. Semua pihak yang tidak dapat disebutkan satu per satu oleh penulis. Penulis mengucapkan banyak terima kasih kepada semua pihak yang terlibat.

Akhir kata, penulis bersama semua pihak yang turut membantu dalam penulisan draf skripsi ini berharap bahwa semoga draf skripsi ini dapat memberikan manfaat serta memberikan tambahan pengetahuan bagi pembacanya.

Wassalamuálaikum Warahmatullahi Wabarakatuh

Samarinda, 07 Maret 2025

Langgeng Prassadewo Sukma Adi Winoto Basla

DAFTAR ISI

DAFTAR GAMBAR

Halaman

DAFTAR TABEL

Halaman

DAFTAR SIMBOL

Simbol	Arti
G	Graf G
$V(G)$	Himpunan simpul-simpul dalam graf G
$E(G)$	Himpunan sisi-sisi dalam graf G
$d(v)$	Derajat simpul v
N	Ukuran *swarm*
j	Indeks dari partikel dalam *swarm*
n	Ukuran iterasi maksimum
i	Indeks dari iterasi sekarang
d	Ukuran dari dimensi partikel
$\boldsymbol{X}_j^{(i)}$	Vektor posisi partikel j pada iterasi ke-i
$x_{j,d}^{(i)}$	Posisi partikel j pada dimensi ke-d saat iterasi ke-i
$\boldsymbol{V}_j^{(i)}$	Vektor kecepatan partikel j pada iterasi ke-i
$v_{j,d}^{(i)}$	Kecepatan partikel j pada dimensi d saat iterasi ke-i
$r_1^{(i)}$	Bilangan acak pertama berdistribusi *uniform* saat iterasi ke-i
$e_2^{(i)}$	Bilangan acak kedua berdistribusi *unifrom* saat iterasi ke-i
c_1	Parameter kecerdasan kognitif partikel
c_2	Parameter kecerdasan sosial partikel
$\omega^{(i)}$	Bobot inersia saat iterasi ke-i
$\boldsymbol{P}_{best,j}^{(i)}$	Vektor posisi terbaik partikel $\boldsymbol{X}_j$ saat iterasi ke-i
$P_{j,d}^{(j)}$	Posisi terbaik partikel j pada dimensi ke-d saat iterasi ke-i
$G_{best}^{(i)}$	Posisi terbaik dari semua partikel saat iterasi ke-i
$F(\boldsymbol{X}_j^{(i)})$	Nilai fungsi *fitness* hasil rute pada partikel j saat iterasi ke-i
$f(\boldsymbol{X}_j^{(i)})$	Total biaya yang dikeluarkan dengan hasil rute pada partikel j saat iterasi ke-i

BAB I

PENDAHULUAN

1.1 Latar Belakang

Syuting atau kegiatan pengambilan gambar dalam proses produksi film, sering kali memiliki susunan urutan adegan-adegan (*scenes*) yang tidak diambil secara terurut seperti urutan saat film tersebut ditayangkan (Qin dkk., 2016). Salah satu faktor utama yang dipertimbangkan dalam pemilihan urutan *scenes* adalah lokasi dan biaya. Besar biaya transportasi yang dikeluarkan berbanding lurus dengan semakin banyak *scenes* dengan lokasi syuting yang berbeda. Jumlah biaya transportasi yang bertambah perlu diwaspadai agar tidak menyebabkan permasalahan pada biaya operasional produksi film.

Tahap produksi sebuah film merupakan proses eksekusi semua hal yang telah dipersiapkan pada tahap pra-produksi, termasuk kegiatan syuting untuk setiap *scene* (Haren, 2020). Pelaksanaan tahap produksi mengharuskan seluruh kru dan pemain (*talent*) produksi film berpindah-pindah lokasi sesuai lokasi syuting *scene*. Akibatnya terdapat biaya tambahan berupa biaya transportasi yang dikeluarkan saat melakukan perpindahan lokasi syuting. Oleh karena itu, pemilihan urutan *scene* syuting mempertimbangkan faktor lokasi dengan memilih total biaya pelaksanaan syuting dan biaya perpindahan antar lokasi *scene* yang minimum. Permasalahan pemilihan urutan *scene* dapat dipandang sebagai permasalahan rute terpendek dengan pengoptimalan biaya perpindahan lokasi *scene* sebagai objek bahasan. Matematika diskret dapat diterapkan terhadap optimasi biaya dalam permasalahan rute terpendek (Qin dkk., 2016).

Matematika diskret merupakan salah satu cabang matematika dengan ruang lingkup kajian berupa objek-objek permasalahan secara diskret (Nasir dkk., 2022). Suatu objek dapat dikatakan objek secara diskret jika objek tersebut terdiri dari sejumlah berhingga anggota dan tiap anggotanya berbeda atau tidak terhubung. Salah satu penerapan dari matematika diskret adalah sistem penyimpanan informasi pada komputer digital yang disimpan dalam bentuk diskret. Cakupan dalam matematika diskret mempelajari diantaranya himpunan, relasi dan fungsi, induksi matematika,

kombinatorial, dan teori graf (Yurinanda dan Rozi, 2023).

Teori graf termasuk ke dalam cakupan matematika diskret. Konsep teori graf pertama kali diperkenalkan oleh Leonhard Euler saat memodelkan solusi untuk melewati ketujuh jembatan K$\ddot{o}$nigsberg di Rusia dengan masing-masing jembatan hanya boleh dilewati tepat satu kali dan kembali ke tempat semula perjalanan dimulai. Sebuah graf G merupakan pasangan terurut himpunan berhingga yang tidak kosong dari simpul (*vertices*) dan himpunan pasangan dari simpul-simpul secara tidak terurut. Implementasi teori graf sering digunakan untuk memecahkan permasalahan optimasi, seperti optimasi pencarian rute terpendek pada konsep *traveling salesman problem* (TSP) (Setiawati dkk., 2023).

Optimasi atau optimalisasi merupakan suatu prinsip pencarian solusi layak yang bersifat tepat, efektif, dan efisien (optimum) (Atiqoh, 2020). Pengertian ptimasi juga dapat diartikan sebagai tindakan tersistematis yang memaksimalkan sumber daya yang tersedia dengan tujuan menemukan solusi optimal. Penerapan optimasi sering diterapkan dalam banyak permasalahan yang memiliki keterbatasan sumber daya, seperti permasalahan pencarian rute terpendek pada konsep *traveling salesman problem* (TSP).

Traveling Salesman Problem (TSP) merupakan konsep rute terpendek pada seorang penjual (*salesman*) yang harus mengunjungi semua kota dan harus kembali ke kota awal serta satu kota hanya boleh dikunjungi satu kali (Rahimi dkk., 2023). Permasalahan optimasi TSP termasuk ke dalam persoalan optimasi kombinatorial dengan memanfaatkan graf sebagai representasi dari model permasalahan. Representasi masalah TSP umumnya dimodelkan ke dalam graf dengan simpul (*vertices*) sebagai lokasi dan sisi (*edge*) sebagai bobot atau biaya yang akan dioptimalkan. Permasalan TSP dapat diselesaikan menggunakan penerapan algoritma *Particle Swarm Optimization* (PSO).

Particle Swarm Optimization (PSO) merupakan algoritma optimasi stokasi terinspirasi dari perilaku sosial burung atau ikan yang berkelompok dalam mencari makan (Azhari dkk., 2018). Algoritma *Particle Swarm Optimization* (PSO) ini termasuk ke dalam algoritma heuristik berbasis populasi. Pencarian solusi optimal menggunakan perilaku partikel dalam populasi itu sendiri. Setiap partikel berperilaku sebagai pencari lingkungan dengan kecerdasan untuk saling berkontribusi dan berkomunikasi

dalam mencari letak objek bahasan.

Penelitian sebelumnya tentang algoritma *Particle Swarm Optimization* (PSO) telah dilakukan oleh beberapa peneliti terdahulu, antara lain 2018 dalam penelitian yang membahas tentang penentuan rute penjemputan angkutan sekolah di MI Salafiyah Kasim. Dengan 2 kloter pengantaran, 150 partikel, 35 iterasi, dan 5 kali percobaan yang dihitung menghasilkan 3 percobaan dengan rute yang lebih optimal dari rute aktual. Natalia dkk. (2019) dalam penelitian yang membahas tentang penentuan jarak penjemputan penumpang optimal di CV. Eira Saudara. Dengan 8 titik lokasi penjemputan dan 90 partikel yang dihitung menghasilkan rute optimal dengan total jarak 59,2 Km. Dalyono dkk. (2017) dalam penelitian yang membahas tentang penentuan rute kunjungan tempat pariwisata di Kota Bandung. Dengan 2 set lokasi pariwasata dihitung dengan 2 kali pengujian, yaitu pengujian pertama dilakukan dengan mengubah 5 parameter pada set pertama dan pengujian kedua membandingkan waktu tempuh luaran antara algoritma *Particle Swarm Optimization* (PSO) dan algoritma *Artificial Immune System* (AIS).

Pada penelitian ini, permasalahan pemilihan urutan *scene* syuting dapat dipandang ke dalam permasalahan optimasi dengan konsep TSP. Sehingga penulis akan menggunakan algoritma *Particle Swarm Optimization* (PSO) untuk mencari solusi dalam pemilihan urutan *scene* syuting yang optimal pada film ”Ketika Adzan Sudah Tidak Lagi Berkumandang”. Film ”Ketika Adzan Sudah Tidak Lagi Berkumandang” merupakan film berkategori film layar lebar dengan genre horor. Film ini diproduksi oleh komunitas East Borneo Film (EBF). Proses produksi film ini menggunakan tiga lokasi utama yang berbeda, yaitu Waduk Tenggarong, Desa Loa Raya, dan Desa Kedang Ipil. Setiap lokasi utama terdapat beberapa titik lokasi yang menjadi lokasi syuting *scene* yang berbeda. Hal ini memerlukan strategi dalam pemilihan urutan *scene* syuting guna menjaga peningkatan biaya transportasi agar biaya produksi tetap minimum.

Berdasarkan uraian di atas, penulis akan membahas mengenai masalah pemilihan urutan *scene* syuting dengan menggunakan algoritma *Particle Swarm Optimization* (PSO) dengan judul ”Optimalisasi Pemilihan Urutan Syuting *Scene* Film Menggunakan Algoritma *Particle Swarm Optimization* (Studi Kasus: Film ”Ketika Adzan Sudah Tidak Lagi Berkumandang”)”.

1.2 Batasan Masalah

Batasan masalah yang digunakan pada penelitian ini untuk mencapai tujuan yang diharapkan adalah:

1. Data yang digunakan adalah data dari *Master Breakdown* film ”Ketika Adzan Sudah Tidak Lagi Berkumandang”.
2. Data lokasi yang digunakan adalah data lokasi syuting dari setiap *scene*.
3. Asumsi yang digunakan antara lain:
   (a) biaya perpindahan antar setiap *scene* merupakan total biaya transportasi ditambahkan biaya talent yang bermain pada *scene* tujuan,
   (b) biaya transportasi merupakan jumlah konsumsi bahan bakar dalam Rupiah,
   (c) kendaraan berjumlah tiga, yaitu satu kendaraan untuk seluruh pemain (*talent*) dan dua kendaraan untuk perlengkapan produksi,
   (d) harga bahan bakar yang digunakan adalah harga bahan bakar jenis Pertalite per bulan Juni 2024, yaitu seharga Rp. $10.000,-$.
4. Graf yang digunakan dalam penelitian ini adalah graf tak berarah.

1.3 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan, rumusan masalah pada penelitian ini adalah:

1. Bagaimana model graf *scene* film ”Ketika Adzan Sudah Tidak Lagi Berkumandang”?
2. Bagaimana penerapan algoritma *Particle Swarm Optimization* (PSO) untuk optimalisasi pemilihan urutan *scene* syuting film ”Ketika Adzan Sudah Tidak Lagi Berkumandang”?
3. Bagaimana hasil optimalisasi pemilihan urutan *scene* syuting film ”Ketika Adzan Sudah Tidak Lagi Berkumandang” menggunakan algoritma *particle swarm optimization* (PSO)?

1.4 Tujuan Penelitian

Berdasarkan rumusan masalah yang telah diuraikan, maka tujuan yang diharapkan dapat dicapai pada penelitian ini adalah:

1. Mengetahui model graf *scene* film ”Ketika Adzan Sudah Tidak Lagi Berkumandang”.

2. Mengetahui penerapan algoritma *Particle Swarm Optimization* (PSO) untuk optimalisasi pemilihan urutan *scene* syuting film ”Ketika Adzan Sudah Tidak Lagi Berkumandang”.
3. Mengetahui hasil optimalisasi pemilihan urutan *scene* syuting film ”Ketika Adzan Sudah Tidak Lagi Berkumandang” menggunakan algoritma *particle swarm optimization* (PSO).

1.5 Manfaat Penelitian

Manfaat yang diharapkan pada penelitian ini adalah:

1. Bagi penulis dan pembaca, dapat memberikan pemahaman lebih luas mengenai penerapan algoritma *Particle Swarm Optimization* (PSO).
2. Bagi universitas, dapat memberikan kontribusi pengembangan ilmu pengetahuan serta dapat menjadi bahan referensi bagi mahasiswa lain.
3. Bagi pihak terkait, dapat memberikan pemahaman mengenai penerapan algoritma *Particle Swarm Optimization* (PSO) dan dapat digunakan untuk membantu pemilihan urutan *scene* syuting film yang akan mendatang.

BAB II

TINJAUAN PUSTAKA

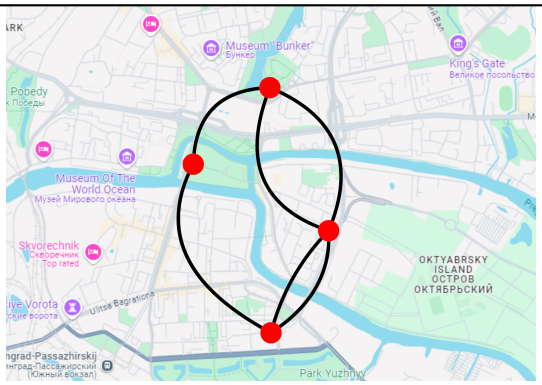
Pada bab ini akan disajikan mengenai pengertian dan penjelasan materi yang berkaitan dengan matematika diskret, teori graf, konsep *Traveling Salesman Problem* (TSP), metode heuristik, algoritma *Particle Swarm Optimization* (PSO), kriteria pemberhentian algoritma, dan Film ”Ketika Adzan Sudah Tidak Lagi Berkumandang”.

2.1 Matematika Diskret

Matematika diskret merupakan salah satu dari cabang matematika dengan ruang lingkup kajian mengenai segala objek-objek permasalahan yang bersifat diskret. Objek-objek yang bersifat diskret memiliki konsep bahwa setiap objek tersebut adalah objek yang terpisah dan berbeda dari yang lain (tidak kontinu). Objek dalam domain matematika seperti bilangan bulat, logika, dan graf dapat diamati sebagai entitas diskret. Beberapa topik utama yang termasuk ke dalam matematika diskret seperti teori himpunan, permutasi, teori kombinatorial, logika, dan teori graf (Sari dkk., 2024).

2.2 Teori Graf

Teori graf merupakan kajian matematika dalam lingkup matematika diskrit yang relatif baru . Kajian mengenai teori graf pertama kali dilakukan oleh matematikawan terkemuka bernama Leonhard Euler pada Tahun 1735. Permasalahan mengenai ”Tujuh Jembatan K*ö*nigsberg” menjadi inspirasi awal mula teori graf dikaji oleh Euler. Euler merepresentasikan jembatan sebagai sisi dan tempat yang dihubungkan oleh jembatan sebagai simpul, sehingga tercipta graf sederhana mengenai ”Tujuh Jembatan K*ö*nigsberg” (Levin, 2021).



Gambar 2.1 Graf Representasi ”Tujuh Jembatan Königsberg”

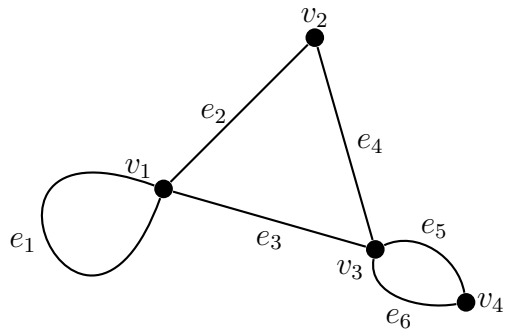
2.2.1 Definisi Graf

Terminologi dari sebuah graf dapat memiliki makna yang berbeda akibat dari luasnya aplikasi graf pada berbagai bidang. Sebuah simpul dari graf pada studi kasus yang berbeda mungkin menyatakan objek yang berbeda. Oleh karena itu secara kasar, graf dapat dipandang sebagai suatu diagram yang memuat sejumlah informasi tertentu dari sebuah studi kasus.

Suatu graf dinotasikan dengan $G = (V, E)$ dalam matematika memiliki definisi sebagai berikut.

Definisi 2.1 *Suatu graf* $G = (V, E)$ *terdiri 2 himpunan pasangan berurut yang berhingga, yaitu himpunan tidak kosong dari titik-titik disebut simpul atau ”vertices” (dinotasikan* $V(G)$*) dan himpunan garis-garis dua elemen subset dari* $V(G)$ *disebut sisi atau ”edges” (dinotasikan* $E(G)$*).*

Setiap sisi di dalam graf G berhubungan dengan satu atau dua simpul. Sisi yang terhubung hanya dengan satu simpul disebut *loop*. Simpul di dalam graf G dapat terhubung dengan lebih dari satu sisi yang berbeda. Dua sisi berbeda yang terhubung dengan satu simpul sama disebut sebagai garis atau sisi paralel. Sisi yang terhubung dengan dua simpul berbeda menyebabkan kedua simpul tersebut menjadi terhubung (*adjacent*) (Jek, 2016).



Gambar 2.2 Graf G

2.2.2 Jalan, Lintasan, dan Sirkuit

Simpul-simpul di dalam graf G dapat dihubungkan oleh lebih dari satu sisi. Jumlah sisi yang terhubung dengan suatu simpul pada graf disebut derajat simpul sesuai dalam definisi berikut.

Definisi 2.2 *Misalkan v merupakan simpul dalam suatu graf G. Derajat simpul v (dinotasikan $d(v)$) adalah jumlah sisi yang terhubung dengan simpul v dan sisi suatu loop dihitung dua kali. Derajat total G didapat dengan menjumlahkan derajat semua simpul dalam G.*

Derajat suatu simpul dalam graf berarah dapat dinyatakan sebagai jumlahan sisi berarah masuk ke simpul dan jumlahan sisi berarah keluar dari simpul (Nurdiyanto dan Susanti, 2019).

Barisan yang diawali dari simpul awal dan diakhiri pada simpul akhir serta terdapat sisi-sisi dan simpul-simpul secara selang-seling disebut jalan (*walk*). Jalan sederhana dengan panjang n dari v_0 sampai v_n dituliskan sebagai berikut : $v_0e_1v_1 \ldots v_{n-1}e_nv_n$. Jalan dapat dimulai dari simpul awal sampai dengan simpul akhir yang semua sisinya berbeda akan membuat sebuah lintasan (*path*). Lintasan dengan simpul awal dan simpul akhir yang sama dapat disebut sebagai sirkuit (*circuit*) (Rahayuningsih, 2018).

2.2.3 Graf Berbobot

Suatu graf berbobot (*Weighted Graph*) merupakan konsep dalam teori graf dengan memberikan bobot atau nilai bertipe numerik terhadap sisi yang terhubung dengan dua simpul pada suatu graf. Bobot ini merepresentasikan suatu atribut tertentu yang terkait dengan relasi antara simpul-simpul sesuai dengan informasi peman-

faatan graf tersebut. Salah satu pemanfaatan graf berbobot dapat digunakan untuk pencarian rute terpendek dengan bobot yang mungkin merepresentasikan jarak, biaya transportasi, atau waktu (Pratama dan Darmawan, 2022).

2.2.4 Representasi Graf Tak Berarah Dalam Matriks

Selaras dengan terminologi graf yang memiliki banyak makna bergantung pada fungsi suatu graf dalam aplikasinya, Suatu graf tak berarah dapat dipetakan menjadi sebuah matriks untuk merepresentasikan hubungan antar elemen-elemen dalam graf tersebut. Keuntungan dari merepresentasikan graf dalam matriks dapat mempermudah perhitungan yang diperlukan. Namun, keterbatatasan matriks untuk mencakup keseluruhan informasi pada suatu graf menjadi kesulitan utama. Terdapat beberapa matriks untuk menyatakan suatu graf tak berarah, diantaranya matriks ketetanggaan dan matriks ketetanggaan berbobot (Chen dkk., 2020).

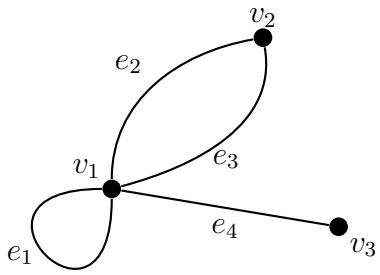
2.2.4.1 Matriks Ketetanggaan

Matriks ketetanggaan (*adjacency matrix*) merupakan matriks yang menyatakan hubungan jumlah sisi dan simpul dari suatu graf yang didefinisikan sebagai berikut.

Definisi 2.3 *Misalkan G adalah graf tak berarah dengan simpul-simpul berhingga (n). Matriks ketetanggaan yang sesuai dengan graf G adalah matriks $A = (a_{ij})$; $i, j = 1, 2, \ldots, n$ dengan a_{ij} merepresentasikan jumlah sisi yang menghubungkan antar simpul yang terhubung.*

Perhatikan pada graf tak berarah bahwa jumlah sisi yang menghubungkan simpul v_i ke simpul v_j selalu sama dengan jumlah sisi yang menghubungkan simpul v_j ke simpul v_i. Akibatnya, matriks ketetanggaan yang merepresentasikan graf tak berarah selalu merupakan matriks yang simetris (Jek, 2016).

Contoh 2.4 Diberikan suatu graf G tak berarah sebagai $G = (\{v_1, v_2, v_3\}, \{(v_1, v_1), (v_1, v_2), (v_1, v_2), (v_1, v_3)\})$. Nyatakan graf G ke dalam bentuk matriks!



Gambar 2.3 Graf Contoh 2.4

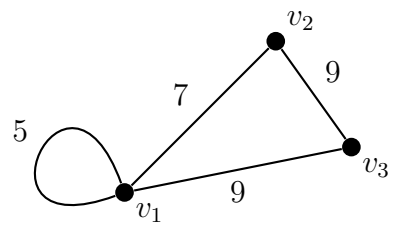
Penyelesaian:

Matriks ketetanggaan yang merepresentasikan graf G adalah sebagai berikut.

$$M_G = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \end{array} \begin{array}{c} \begin{array}{ccc} v_1 & v_2 & v_3 \end{array} \\ \begin{bmatrix} 1 & 2 & 1 \\ 2 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \end{array}$$

Misal suatu graf tak berarah G memiliki bobot pada setiap sisinya, maka matriks ketetanggaan menyatakan bobot dari simpul yang terhubung kemudian disebut sebagai matriks ketetanggaan berbobot (*weighted adjacency matrix*). Matriks ketetanggaan berbobot yang merepresentasikan graf tak berarah akan bernilai 0 jika simpul-simpul tersebut tidak terhubung (Akhirina dan Afrizal, 2020).

Contoh 2.5 Diberikan suatu graf G tak berarah berbobot direpresentasikan dalam gambar berikut.



Gambar 2.4 Graf Contoh 2.5

Nyatakan graf G ke dalam matriks ketetanggaan berbobot!

Penyelesaian:

Matriks ketetanggaan berbobot yang merepresentasikan graf G sebagai berikut.

$$A_G = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \end{array} \begin{array}{c} \begin{array}{ccc} v_1 & v_2 & v_3 \end{array} \\ \begin{bmatrix} 5 & 7 & 9 \\ 7 & 0 & 9 \\ 9 & 9 & 0 \end{bmatrix} \end{array}$$

Apabila suatu graf G tak berarah memiliki dua simpul (v_1 dan v_2) dihubungkan oleh dua sisi dengan bobot berbeda, maka representasi matriks ketetanggaan berbobot untuk dua simpul tersebut menyesuaikan pengaplikasiannya. Graf tak berarah yang merepresentasikan permasalahan maksimasi akan mengambil bobot dengan nilai terbesar dan berlaku sebaliknya.

2.2.4.2 Matriks Biner

Matriks biner dapat juga disebut matriks $(0-1)$ atau matriks insidensi (*incidence matrix*) adalah matriks yang menyatakan hubungan keterhubungan simpul dengan sisi dari suatu graf yang didefinisikan sebagai berikut.

Definisi 2.6 *Misalkan G adalah graf tanpa loop dengan n simpul $v_1, v_2, \ldots, v_n$ dan k sisi $e_1, e_2, \ldots, e_k$. Matriks biner yang sesuai dengan graf G adalah matriks A berukuran $n \times k$ yang elemennya terdiri dari*

$$a_{ij} = \begin{cases} 1 & \text{, jika simpul } v_i \text{ berhubungan dengan sisi } e_j \\ 0 & \text{, , jika simpul } v_i \text{ tidak berhubungan dengan sisi } e_j \end{cases} \tag{2.1}$$

Matriks biner sering digunakan ke dalam bidang teori kode, logika digital, dan operasi komputasi (Jek, 2016).

2.3 Optimasi

Optimasi atau optimalisasi merupakan suatu prinsip pencarian solusi layak yang bersifat tepat, efektif, dan efisien (optimum) (Atiqoh, 2020). Pengertian teknis dari optimasi dapat diartikan sebagai tindakan tersistematis yang memaksimalkan sumber daya yang tersedia dengan tujuan menemukan solusi optimal. Solusi dalam optimasi berada dalam daerah layak (*feasible region*) yang memiliki nilai minimum atau maksimum dari fungsi objektif. Proses pencarian solusi optimum dalam optimasi terkait dengan permasalahan komputasional yang bertujuan menemukan so-

lusi terbaik dan layak dari sejumlah alternatif solusi terbatas. Penerapan optimasi sering diterapkan dalam banyak permasalahan yang memiliki keterbatasan sumber daya, seperti permasalahan pencarian rute terpendek pada konsep *traveling salesman problem* (TSP) (Devita dan Wibawa, 2020).

2.4 Konsep *Traveling Salesman Problem* (TSP)

Traveling salesman problem (TSP) merupakan salah satu konsep klasik dalam teori graf dan kombinatorial mengenai pencarian rute terpendek atau lintasan tertutup yang mengunjungi setiap simpul (lokasi) tepat satu kali dan kembali ke simpul awal dengan mengoptimalkan total jarak atau biaya perjalanan agar minimum. Representasi suatu permasalahan TSP umumnya dimodelkan ke dalam graf dengan simpul sebagai lokasi dan sisi sebagai bobot yang akan dioptimalkan. Meskipun konsep TSP terlihat sederhana, namun tingkat kesulitan akan meningkat seiring eksponensial jumlah kemungkinan jalur saat jumlah simpul meningkat (Sinaga dan Marpaung, 2023).

Pembentukan pertama kajian matematika terhadap permasalahan TSP oleh seorang matematikawan Irlandia W.R. Hamilton pada abad ke-19. Penyelesaian permasalahan TSP diperoleh melalui pendekatan-pendekatan tertentu. Penyelesaian TSP melalui pendekatan matematis pertama oleh Merril Flood pada tahun 1930. Kemudian permasalahan tersebut dikenal dengan nama *Traveling Salesman Problem* yang diciptakan oleh Hassler Whitney dari Universitas Princeton. Permasalahan TSP termasuk ke dalam permasalahan *nondeterministic polynomial time-hard*(*NP-Hard*) yang berarti tidak ada algoritma yang efisien untuk setiap permasalahan TSP (Kumar dan Memoria, 2020).

Konsep TSP memiliki banyak penerapan praktis untuk memecahkan permasalahan optimasi yang melibatkan penentuan rute terpendek sehingga setiap simpul dapat dikunjungi dengan biaya atau waktu minimal. Pemecahan solusi optimal untuk permasalahan TSP dengan jumlah simpul yang besar akan menjadi cukup sulit. Pendekatan dengan berbagai metode banyak dilakukan agar dapat memperoleh solusi yang optimal, salah satunya adalah metode pendekatan secara heuristik (Saud dkk., 2018).

2.5 Metode Heuristik

Metode heruistik merupakan strategi pendekatan komputasional untuk menyelesaikan permasalahan pengambilan keputusan yang mungkin sulit bahkan tidak mungkin diselesaikan secara eksak dalam waktu yang wajar. Tujuan utama dari metode heuristik adalah untuk menghasilkan solusi yang cukup baik dengan biaya komputasi yang juga terjangkau. Heuristik mengacu pada aturan praktis atau strategi mental berbasis pengalaman, pengetahuan, atau asumsi yang digunakan dalam mengambil keputusan dengan penyederhanaan prosesnya. Meskipun metode heuristik dapat membantu terhadap permasalahan yang kompleks atau tidak pasti, namun solusi yang dihasilkan tidak menjamin optimal dan sering menyebabkan bias dalam pemikiran (Anwar dkk., 2024).

Metode heuristik secara umum memberikan kerangka kerja yang sederhana dan cepat dengan berbasis pengalaman dan pengetahuan. Adapun beberapa metode heuristik secara umum adalah sebagai berikut:

1. Heuristik Ketersediaan

   Heuristik ketersediaan mengambil keputusan didasarkan pada seberapa mudah mengingat kembali contoh-contoh atau informasi terkait. Contohnya orang yang baru mendengar kabar mengerikan mengenai kecelakaan kapal akan berpikir bahwa berlayar itu pilihan tidak aman.

2. Heuristik Representatif

   Heuristik representatif mengambil keputusan didasarkan pada asumsi suatu objek atau kondisi serupa dengan pola atau kategori tertentu seperti kesamaan fitur atau kesamaan karakteristik lain. Contohnya penampilan seseorang dinilai mewakili karakteristik kelompok tertentu.

3. Heuristik Ancoran dan Penyesuaian

   Heuristik estimasi awal (ancoran) dan penyesuaian mengambil keputusan dengan membuat estimasi awal berdasarkan informasi awal yang kemudian mengubah estimasi tersebut berdasarkan informasi yang diterima selanjutnya. Contohnya tawar-menawar harga mobil dengan estimasi awal mengikuti harga jual awal oleh penjual kemudian berubah berdasarkan informasi spesifikasi mobil tersebut.

4. Heuristik Kesederhanaan

   Heuristik kesederhanaan mengambil keputusan secara sederhana berdasarkan pada informasi yang paling mudah diakses untuk menggantikan informasi sebenarnya. Contohnya memutuskan membeli sepatu merk A menggunakan opini yang sedang *trending* di media sosial.

5. Heuristik Atribusi

   Heuristik atribusi mengambil keputusan didasarkan pengetahuan saat ini untuk memberikan penjelasan singkat terhadap kondisi atau perilaku tertentu. Contohnya memberikan penjelasan bahwa orang yang terlambat itu akibat dari rasa malas dari orang tersebut.

Metode heuristik banyak diaplikasikan ke dalam berbagai bidang yang dihadapkan pengambilan keputusan tertentu. Salah satu contoh penggunaan metode heuristik adalah algoritma *particle swarm optimization* yang termasuk ke dalam metode heuristik representatif dengan menggunakan perilaku sosial kawanan hewan dalam mencari mangsa sebagai representatif permasalahan optimasi (Muhammad. dkk., 2023).

2.6 Algoritma *Particle Swarm Optimization* (PSO)

Algoritma *particle swarm optimization* (PSO) merupakan algoritma optimasi heurisitik berbasis populasi yang diperkenalkan oleh James Kennedy dan Russel Eberhart pada tahun 1995. Basis populasi terinspirasi dari perilaku sosial hewan seperti burung dalam suatu kawanan (*swarm*) pada saat menelusuri area tertentu untuk mencari makanan. Perilaku sosial kawanan tersebut dalam mencari makanan meliputi kecerdasan, kecepatan, dan komunikasi yang terorganisir (Kusrahman dkk., 2020).

2.6.1 Definisi Algoritma *Particle Swarm Optimization*

Algoritma *particle swarm optimization* merupakan salah satu teknik optimasi komputasional. Perilaku sosial hewan dalam mencari makan pada suatu daerah menginspirasi konsep dasar algoritma *particle swarm optimization*. Konsep untuk mengeksploitasi individu dalam pencarian solusi, dilakukan dengan populasi yang disebut *swarm* dan individu disebut *particle*. Setiap *particle* akan bergerak semakin mendekati solusi dengan kecepatan (*velocity*) yang terus beradaptasi ter-

hadap daerah pencarian dan informasi posisi terbaik yang pernah dicapai selalu disimpan (Muhammad. dkk., 2023).

Konsep penelusuran algoritma *particle swarm optimization* memiliki kesamaan dengan algoritma genetika, yaitu populasi awal yang digunakan adalah populasi acak dalam bentuk matriks. Representasi populasi pada matriks menggunakan baris sebagai *particle* untuk algoritma *particle swarm optimization* atau kromosom untuk algoritma genetika. Perbedaan mendasar antara kedua algoritma ini adalah algoritma *particle swarm optimization* tidak memiliki operator yang dapat mengevolusi partikel seperti *crossover* dan mutasi pada algoritma genetika (Zahro dan Wahyuni, 2020).

Algoritma *particle swarm optimization* menggunakan beberapa istilah untuk mencakup informasi dari suatu persamalahan optimasi. Adapun istilah umum yang sering digunakan dalam algoritma *particle swarm optimization* dijelaskan sebagai berikut:

1. *Swarm* : populasi dari algoritma *particle swarm optimization*.
2. *Particle* (X_j) : individu di dalam suatu *swarm*. Dalam satu terdiri dari beberapa dimensi (d) sesuai dengan dimensi dari model permasalahan yang akan diselesaikan. Seluruh *particle* merepresentasikan solusi potensial pada saat proses optimisasi. Posisi setiap *particle* selalu diperbarui berdasarkan kecepatan (*velocity*) tertentu.
3. *Velocity* (v): vektor yang memperbarui posisi dan arah *particle* berpindah dari posisi semula.
4. *Inertia Weight* (ω) : parameter untuk mengendalikan dampak dari *velocity* sebelumnya terhadap *velocity* sekarang suatu *particle*.
5. *Personal Best* (P_{best}) : posisi terbaik yang pernah dicapai oleh *particle* yang dipersiapkan untuk mendapatkan solusi terbaik.
6. *Global Best* (G_{best}) : posisi terbaik *particle* pada *swarm*.

(Muhardeny dkk., 2023).

Partikel solusi dalam algoritma *particle swarm optimization* diperoleh dari partikel dengan posisi yang menghasilkan nilai fungsi *fitness* terbaik di semua iterasi. Partikel berisikan posisi dari sejumlah dimensi partikel, dimana dimensi partikel merupakan jumlah parameter yang akan dioptimalkan. Proses pencarian solusi pada

algoritma *particle swarm optimization* memandang bahwa untuk setiap partikel cenderung akan bergerak ke arah solusi berdasarkan kecepatan (*velocity*). Vektor posisi dan kecepatan partikel pada iterasi sekarang dapat dinotasikan dalam persamaan berikut:

$$\boldsymbol{X}_j^{(i)} = \begin{bmatrix} x_{j,1}^{(i)} \\ \vdots \\ x_{j,d}^{(i)} \end{bmatrix} \tag{2.2}$$

dan

$$\boldsymbol{V}_j^{(i)} = \begin{bmatrix} v_{j,1}^{(i)} \\ \vdots \\ v_{j,d}^{(i)} \end{bmatrix} \tag{2.3}$$

dengan j adalah indeks dari partikel dan d adalah jumlah dimensi partikel (Akpudo dan Jang-Wook, 2020).

Penggunaan algoritma *particle swarm optimization* memiliki keuntungan utama yaitu memiliki waktu komputasi yang rendah, parameter yang perlu disesuaikan relatif sedikit, dan mampu membangun model matematika yang akurat untuk suatu permasalahan kompleks. Keuntungan lain penggunaan algoritma *particle swarm optimization* adalah tidak terjadi tumpah tindih atau mutasi pada saat proses perhitungan. Keuntungan tersebut sudah cukup menjadikan algoritma *particle swarm optimization* sering untuk digunakan. Namun algoritma ini memiliki kelemahan, salah satunya adalah memerlukan penyimpanan data lebih untuk memperbarui *velocity* maupun posisi terbaik saat iterasi (Gad, 2022).

2.6.2 Parameter Algoritma *Particle Swarm Optimization*

Parameter dalam algoritma *particle swarm optimization* memberikan pengaruh besar terhadap kinerja algoritma. Pemilihan parameter yang digunakan dan penentuan nilainya dapat meningkatkan kinerja algoritma menjadi lebih efisien. Terdapat beberapa parameter dasar dari algoritma *particle swarm optimization* dijelaskan sebagai berikut:

1. Ukuran *Swarm*

   Ukuran *swarm* merupakan nilai banyaknya *swarm* yang menjadi populasi dari algoritma. Penentuan ukuran *swarm* yang besar dapat mengurangi jumlah iterasi untuk mencapai perhitungan yang konvergen. Akan tetapi, penggunaan ukuran *swarm* yang besar tentu akan meningkatkan perhitungan tiap iterasi menjadi lebih kompleks. Hal ini mengakibatkan waktu untuk menyelesaikan perhitungan dalam satu iterasi menjadi semakin lama. Besar dari sebuah populasi juga dapat menentukan ukuran *swarm*. Beberapa penelitian tentang implementasi algoritma *particle swarm optimization* klasik menggunakan interval $N \in [20, 50]$ untuk ukuran *swarm* (Piotrowski dkk., 2020).

2. Jumlah iterasi

   Jumlah iterasi merupakan jumlah pengulangan dari proses perhitungan yang akan dilakukan dalam algoritma. Penentuan jumlah iterasi pada algoritma *particle swarm optimization* mempengaruhi pencarian solusi optimal. Jika jumlah iterasi yang digunakan terlalu kecil akan berdampak pada proses perhitungan menjadi berhenti saat solusi belum mencapai optimal. Namun sebaliknya, jumlah iterasi yang terlalu besar akan menyebabkan proses perhitungan menjadi semakin kompleks, menambah waktu perhitungan, serta menambahkan perhitungan yang tidak diperlukan (Jain dkk., 2022).

3. *Learning Rates* (c_1 dan c_2)

   Learning rates atau *acceleration coefficients* terdiri dari dua parameter adaptif dalam algoritma *particle swarm optimization* yang mengatur keseimbangan dari kecerdasan posisi partikel terhadap pengalaman pribadi yaitu (P_{best}) dan pengalaman kolektif yaitu (G_{best}). Parameter koginitf (c_1) merupakan parameter yang mengatur pengaruh P_{best} terhadap posisi partikel. Sedangkan, parameter sosial (c_2) merupakan parameter yang mengatur pengaruh G_{best} terhadap posisi partikel. Terdapat beberapa penentuan nilai c_1 dan c_2, yaitu sebagai berikut:

   (a). Jika $c_1 = c_2 = 0$, maka setiap partikel akan bergerak dengan kecepatan partikel tersebut secara konstan sampai batas ruang pencarian.

   (b). Jika $c_1 > 0$ dan $c_2 = 0$, maka setiap partikel akan bergerak dengan kecepatan partikel terhadap P_{best} masing-masing (independen) karena komponen sosial tidak mempengaruhi. Sedangkan sebaliknya, maka setiap par-

tikel akan bergerak dengan kecepatan partikel terhadap G_{best} sehingga partikel hanya akan tertarik pada satu arah titik.

(c). Jika $c_1 = c_2$, maka setiap partikel akan bergerak dengan kecepatan partikel terhadap rata-rata dari P_{best} dan G_{best}.

(d). Jika $c_1 > c_2$, maka kecepatan partikel untuk setiap partikel akan lebih dipengaruhi oleh P_{best} daripada G_{best}. Sedangkan sebaliknya, maka kecepatan partikel akan lebih dipengaruhi G_{best} daripada P_{best}.

Penentuan nilai dari c_1 dan c_2 secara umum bersifat statis yang diperoleh dari beberapa penelitian terhadulu. Kesalahan dalam penentuan nilai c_1 dan c_2 menyebabkan proses pencarian solusi menjadi divergen. Dalam berbagai penelitian, telah diusulkan bahwa penentuan nilai c_1 dan c_2 dapat menggunakan nilai $c_1 = c_2 = 2$ (2022).

4. *Inertia Weight* (ω)

Inertia weight atau bobot inersia (ω) merupakan parameter yang mengontrol efek dari *velocity* sebelumnya terhadap *velocity* sekarang dari suatu partikel. Penentuan *inertia weight* yang terlalu besar akan menyebabkan peningkatan berlebihan pada *velocity* saat diperbarui, akibatnya partikel akan terlalu jauh bergerak dan terlalu cepat bahkan melewati solusi optimal. Terdapat beberapa penentuan nilai *inertia weight*, yaitu sebagai berikut:

(a). Jika $\omega \geq 1$, maka nilai *velocity* akan meningkat secara berlebih saat diperbarui dan tidak dapat bergerak ke arah yang optimal sehingga solusi akan menjadi divergen.

(b). Jika $\omega < 1$, maka akan terdapat momentum kecil pada *velocity* sebelumnya sehingga dapat digunakan untuk melakukan perubahan arah secara cepat menuju ke posisi yang lebih baik saat proses pencarian solusi.

(c). Jika $\omega = 0$, maka partikel akan bergerak tanpa mengetahui *velocity* sebelumnya sehingga partikel tidak dapat mengetahui perubahan posisinya secara benar bahkan dapat keluar dari arah solusi optimal.

Nilai *inertia weight* yang sesuai dengan ketentuan tersebut adalah pada interval $\omega \in [0.1, 0.9]$. Shi dan Eberhart memperkenalkan strategi *constant inertia weight* dan *random inertia weight*. Strategi *constant inertia weight* akan mengambil nilai *inertia weight* di setiap iterasi adalah konstan $(\omega^{(i)} = \omega)$. Sedangkan pada

strategi *random inertia weight*, nilai *inertia weight* diambil secara acak namun tetap berada di atas rata-rata dari interval ketentuan. Nilai *inertia weight* di setiap iterasi i dengan strategi *random inertia weight* ditentukan menggunakan persamaan berikut:

$$\omega^{(i)} = 0.5 + \frac{r}{2} \tag{2.4}$$

dengan nilai r adalah bilangan acak berdistribusi *uniform* $r \sim U(0, 1)$ (Zdiri dkk., 2021).

2.6.3 Kecepatan Partikel (*Velocity*)

Velocity atau kecepatan partikel (v) merupakan vektor yang memperbarui posisi dan arah *particle* berpindah dari posisi sebelumnya. Penentuan *velocity* yang terlalu besar dapat menyebabkan *particle* bergerak tidak menentu dan melewati solusi optimal. Dan sebaliknya, *velocity* yang terlalu kecil dapat menyebabkan *particle* bergerak terlalu lambat bahkan tidak bergerak sehingga posisinya terjebak hanya di lokal optimal. Eberhart dan Kennedy pertama kali memperkenalkan strategi batas penentuan kecepatan (*velocity clamping*). Strategi *velocity clamping* menentukan kecepatan maksimum (V_{max}) berdasarkan inisialisasi posisi dan konstanta acak. Kecepatan maksimum ditentukan menggunakan persamaan berikut:

$$V_{max} = \varepsilon(X_{max} - X_{min}) \tag{2.5}$$

dengan nilai ε adalah interval nilai acak $\varepsilon \in [0, 1]$ dan X_{max}, X_{min} adalah batas atas dan bawah dari inisialisasi posisi (X) (Wang dkk., 2022).

2.6.4 *Personal Best*

Personal best atau lokal optimum (P_{best}) dalam algoritma *particle swarm optimization* merupakan vektor yang menyimpan posisi terbaik untuk setiap partikel X_j di setiap iterasi i. Posisi terbaik didasarkan pada posisi dari X_j yang menghasilkan nilai fungsi *fitness* terbaik yang pernah dicapai di semua iterasi sampai iterasi terakhir $i = n$. Untuk setiap iterasi, hasil perhitungan nilai fungsi *fitness* partikel X_j sekarang akan dibandingkan dengan nilai fungsi *fitness* menggunakan posisi dari P_{best} sebelumnya. Dengan kata lain, *personal best* berisikan semua posisi terbaik yang pernah dikunjungi oleh partikel X_j hingga iterasi ke n. *Personal best* dapat

dinotasikan sebagai berikut:

$$\boldsymbol{P}_{best,j}^{(i)} = \begin{bmatrix} p_{j,1}^{(i)} \\ \vdots \\ p_{j,d}^{(i)} \end{bmatrix} \in X_j \tag{2.6}$$

dengan d adalah jumlah dimensi di dalam X_j (Ramadhan dkk., 2023).

2.6.5 *Global Best*

Global best atau global optimum (G_{best}) dalam algoritma *particle swarm optimization* merupakan vektor yang menyimpan posisi dari P_{best} terbaik saat iterasi sekarang. Untuk setiap iterasi i, G_{best} didasarkan pada P_{best} dari setiap partikel X_j pada iterasi i yang memiliki nilai fungsi *fitness* terbaik. Dengan kata lain, *global best* berisikan posisi dari P_{best} terbaik pada iterasi sekarang. *Global best* dapat dinotasikan sebagai berikut:

$$\boldsymbol{G}_{best}^{(i)} = \begin{bmatrix} g_1^{(i)} \\ \vdots \\ g_d^{(i)} \end{bmatrix} \tag{2.7}$$

dengan d adalah jumlah dimensi di dalam X (Darmawan dkk., 2022).

2.6.6 Pembaruan Posisi dan *Velocity*

Proses pencarian solusi pada algoritma *particle swarm optimization* memandang bahwa setiap partikel cenderung akan bergerak ke arah solusi berdasarkan kecepatan (*velocity*) tertentu. Posisi dan *velocity* dari partikel akan selalu diperbarui di setiap iterasi sampai iterasi terakhir $i = n$. Nilai dari *velocity* akan menentukan arah perpindahan dari setiap partikel di setiap iterasi i agar dapat menemukan ruang solusi terbaik. Dalam pembaruan nilai *velocity* akan terpengaruh oleh *velocity*, *personal best*, dan *global best* dari iterasi sebelumnya. Pembaruan posisi dan *velocity* ditentukan menggunakan persamaan berikut:

$$\boldsymbol{V}_j^{(i)} = \omega^{(i)} \boldsymbol{V}_j^{(i-1)} + c_1 r_1^{(i)} \left(\boldsymbol{P}_{best,j}^{(i-1)} - \boldsymbol{X}_j^{(i-1)} \right) + c_2 r_2^{(i)} \left(\boldsymbol{G}_{best} - \boldsymbol{X}_j^{(i-1)} \right) \tag{2.8}$$

dan

$$\boldsymbol{X}_j^{(i)} = \boldsymbol{X}_j^{(i-1)} + \boldsymbol{V}_j^{(i)} \tag{2.9}$$

dengan $r_1^{(i)}, r_2^{(i)}$ adalah bilangan acak berdistribusi *uniform* $r \sim U(0,1)$ di setiap iterasi i (Chafi dan Afrakhte., 2021).

2.6.7 Tahapan Algoritma *Particle Swarm Optimization*

Proses pencarian solusi optimum pada algoritma *particle swarm optimization* dilakukan hingga iterasi maksimum atau kriteria pemberhentian terpenuhi. Untuk setiap iterasi, perpindahan posisi selalu diperbarui sehingga mendekati solusi optimum. Tahapan dan proses perhitungan algoritma *particle swarm optimization* untuk konsep *traveling salesman problem* sebagai berikut:

1. Menginisialisasi nilai parameter algoritma *particle swarm optimization* seperti jumlah iterasi maksimum (n), ukuran *swarm* (N), (d), (c_1, c_2), batas posisi (X_{min}, X_{max}), posisi awal (X_i), batas *velocity* maksimum (V_{max}), dan *velocity* awal (V_j^0). Inisialisasi nilai awal dapat dilakukan melalui tahap berikut:
   (a). Menentukan jumlah iterasi maksimum (n).
   (b). Menentukan ukuran *swarm* (N) dan dimensi partikel (d) didasarkan pada beberapa penelitian tentang implementasi algoritma *particle swarm optimization* klasik yaitu menggunakan interval $N \in [20, 50]$ serta besar dimensi partikel menyesuaikan dari model permasalahan yang akan diselesaikan (Piotrowski dkk., 2020).
   (c). Menentukan *learning rates* (c_1) dan (c_2) didasarkan pada beberapa penelitian tentang *learning rates* dalam algortima *particle swarm optimization* yaitu menggunakan nilai $c_1 = c_2 = 2$ (Jain dkk., 2022).
   (d). Menentukan batas atas (X_{max}) dan batas bawah (X_{min}) untuk posisi partikel.
   (e). Menentukan batas *velocity* maksimum (V_{max}) diperoleh secara acak berdasarkan persamaan 2.5, sedangkan *velocity* minimum (V_{min}) diperoleh dari $V_{min} = -V_{max}$.
   (f). Membangkitkan *velocity* awal $(\boldsymbol{V}_j^0)$ sejumlah N partikel secara acak yang tetap berada di batas *velocity* menggunakan persamaan berikut:

$$v_{j,d}^{(0)} = U(V_{min}, V_{max}) \tag{2.10}$$

(g). Membangkitkan posisi awal $(\boldsymbol{X}_j^0)$ sejumlah N partikel secara acak yang tetap berada di batas posisi partikel menggunakan persamaan berikut:

$$x_{j,d}^{(0)} = U(X_{min}, X_{max}) \tag{2.11}$$

(h). Mengevaluasi hasil rute setiap partikel berdasarkan urutan dimensi dalam partikel dengan nilai posisi terkecil hingga terbesar.

2. Mengevaluasi nilai fungsi *fitness* hasil rute setiap partikel dengan menggunakan persamaan berikut:

$$F(X_j^{(i)}) = \frac{1}{f(X_j^{(i)})} \tag{2.12}$$

dengan $f(X_j)$ adalah total biaya yang dikeluarkan untuk menggunakan hasil rute dari partikel j.

3. Menentukan $P_{best}^{(0)}$ dan $G_{best}^{(0)}$ awal.
4. Membangkitkan nilai *inertia weight* berdasarkan strategi *random inertia weight* menggunakan persamaan 2.4.
5. Memperbarui *velocity* setiap partikel menggunakan persamaan 2.8 dengan nilai $r_1^(i)$ dan $r_2^(i)$ dibangkitkan secara acak menggunakan distribusi *uniform* $r \sim U(0, 1)$. Jika *velocity* terbaru tidak berada di dalam batas yang diizinkan maka dilakukan penyesuaian sebagai berikut:

$$v_{j,d}^{(i)} = \begin{cases} V_{min} & , v_{j,d}^{(i)} < V_{min} \\ V_{max} & , v_{j,d}^{(i)} > V_{max} \end{cases} \tag{2.13}$$

6. Memperbarui posisi setiap partikel menggunakan persamaan 2.9 dengan *velocity* yang telah diperbarui. Jika posisi terbaru tidak berada di dalam batas yang diizinkan maka dilakukan penyesuaian sebagai berikut:

$$x_{j,d}^{(i)} = \begin{cases} X_{min} & , x_{j,d}^{(i)} < X_{min} \\ X_{max} & , x_{j,d}^{(i)} > X_{max} \end{cases} \tag{2.14}$$

7. Mengevaluasi hasil rute setiap partikel.
8. Mengevaluasi nilai fungsi *fitness* hasil rute setiap partikel.

9. Menentukan $P_{best,j}^{(i)}$ setiap partikel dan $G_{best}^{(i)}$. Untuk setiap partikel, $P_{best,j}^{(i)}$ ditentukan dengan menggunakan persamaan berikut:

$$P_{best,j}^{(i)} = \begin{cases} X_j^{(i)} & , F(X_j^{(i)}) \geq F(P_{best,j}^{(i-1)}) \\ P_{best,j}^{(i-1)} & , F(X_j^{(i)}) < F(P_{best,j}^{(i-1)}) \end{cases} \tag{2.15}$$

dan untuk $G_{best}^{(i)}$ ditentukan menggunakan persamaan berikut:

$$G_{best}^{(i)} = \begin{cases} X_j^{(i)} & , F(X_j^{(i)}) \geq F(G_{best}^{(i-1)}) \\ G_{best}^{(i-1)} & , F(X_j^{(i)}) < F(G_{best}^{(i-1)}) \end{cases} \tag{2.16}$$

10. Mengevaluasi kriteria pemberhentian terhadap solusi terbaru yang diperoleh. Jika solusi terbaru telah memenuhi kriteria pemberhentian maka iterasi berhenti, jika tidak memenuhi maka iterasi diperbarui dan kembali ke langkah 4.

2.7 Kriteria Pemberhentian Algoritma *Particle Swarm Optimization*

Proses pencarian solusi pada algoritma *particle swarm optimization* selalu diperbarui di setiap iterasi. Jika solusi terbaru telah memenuhi kriteria pemberhentian, maka pencarian solusi dihentikan. Beberapa kriteria pemberhentian pada algoritma *particle swarm optimization* untuk konsep *traveling salesman problem* adalah sebagai berikut:

1. Iterasi maksimum, proses pencarian solusi akan dihentikan jika iterasi yang berjalan sudah mencapai batas iterasi maksimum.
2. Nilai fungsi *fitness* maksimum, proses pencarian solusi akan dihentikan jika nilai fungsi *fitness* telah mencapai batas yang diberikan, yaitu total biaya maksimum.
3. Populasi konvergen, proses pencarian solusi akan dihentikan jika standar deviasi dari posisi partikel dalam P_{best} kurang dari batas toleransi.
4. Nilai fungsi *fitness* konvergen, proses pencarian solusi akan dihentikan jika beda antara nilai fungsi *fitness* maksimum dan minimum dalam satu iterasi kurang dari batas toleransi.
5. Solusi (G_{best}) konvergen, proses pencarian solusi akan dihentikan jika beda antara nilai fungsi *fitness* dari G_{best} sekarang dan iterasi sebelumnya kurang dari batas toleransi.

Kriteria pemberhentian akan mencegah sumber daya tidak terpakai secara sia-sia (Eltamaly, 2021).

2.8 Film ”Ketika Adzan Sudah Tidak Lagi Berkumandang”

Film berjudul ”Ketika Adzan Sudah Tidak Lagi Berkumandang” merupakan film layar lebar dokumenter bergenre horor yang diproduksi oleh East Borneo Film. Dalam film ini menceritakan tentang dua wartawan yang bertugas di sebuah desa terpencil untuk membuat sebuah film dokumenter di sana. Pembuatan film ini diproduseri sekaligus disutradarai oleh David Richard. Keseluruhan proses pembuatan film ini dilakukan di Kalimantan Timur.

Naskah film ”Ketika Adzan Sudah Tidak Lagi Berkumandang” menghasilkan banyak pecahan *scene* yang harus diambil dalam proses syuting. Proses syuting pada film ini menggunakan tiga lokasi utama yang berbeda, yaitu Waduk Tenggarong, Desa Loa Raya, dan Desa Kedang Ipil. Setiap lokasi utama terbagi menjadi beberapa titik lokasi yang menjadi lokasi syuting *scene* yang berbeda. Perpindahan lokasi pada proses syuting antar *scene* akan memindahkan seluruh alat dan properti sehingga memerlukan biaya transportasi. Peningkatan biaya transportasi yang berlebih dapat membebani biaya produksi dari film ini.

BAB III

METODE PENELITIAN

3.1 Waktu dan Tempat Penelitian

Penelitian ini dilaksanakan pada bulan Maret 2025 sampai Mei 2025. Pengambilan data dilakukan pada film ”Ketika Adzan Sudah Tidak Lagi Berkumandang”. Pengolahan data dalam penelitian ini dilakukan di Laboratorium Matematika Dasar dan Laboratorium Matematika Komputasi yang terletak di Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Mulawarman, Samarinda, Kalimantan Timur.

3.2 Variabel Penelitian

Variabel yang digunakan dalam penelitian ini adalah lokasi syuting setiap *scene* dan biaya perpindahan antar setiap *scene* pada film ”Ketika Adzan Sudah Tidak Lagi Berkumandang”. Penelitian ini memfokuskan pada pemilihan urutan syuting *scene* dengan konsep *traveling salesman problem* menggunakan algoritma *particle swarm optimization* sehingga tahap syuting setiap *scene* berjalan lebih efektif dan efisien. Variabel yang digunakan dapat dilihat pada tabel berikut:

Tabel 3.1 Variabel Penelitian

Variabel	Keterangan	Satuan
v_i	Titik atau lokasi ke-i	-
$v_i v_j$	Biaya antara titik ke-i dan titik ke-j	Rupiah (Rp)

3.3 Teknik Pengumpulan Data

Teknik pengumpulan data yang digunakan dalam penelitian ini adalah menggunakan data sekunder. Data sekunder yang diambil dari *Master Breakdown* film ”Ketika Adzan Sudah Tidak Lagi Berkumandang” terdiri dari lokasi syuting setiap *scene*, pemain (*talent*) di setiap *scene*, dan biaya setiap pemain (*talent*). Sedangkan data sekunder yang diambil dari aplikasi Google Maps adalah jarak dari lokasi syuting setiap *scene*.

3.4 Populasi dan Sampel Penelitian

Populasi pada penelitian ini adalah rute dan biaya perpindahan antar setiap *scene* pada film ”Ketika Adzan Sudah Tidak Lagi Berkumandang”. Sampel yang digunakan pada penelitian ini sama dengan populasi penelitian, yaitu seluruh rute dan biaya perpindahan antar setiap *scene* pada film ”Ketika Adzan Sudah Tidak Lagi Berkumandang”.

3.5 Teknik Sampling

Teknik sampling yang digunakan dalam penelitian ini adalah teknik *purposive sampling*. Teknik *purposive sampling* merupakan teknik penentuan sampel dengan mempertimbangankan maksud dan tujuan tertentu karena sampel tersebut memiliki informasi yang diperlukan (Amin dkk., 2023). Pada penelitian ini sampel yang digunakan sama dengan populasi karena bertujuan untuk mengambil seluruh data pada film ”Ketika Adzan Sudah Tidak Lagi Berkumandang”.

3.6 Teknik Analisis Data

Teknik analisis data pada penelitian ini menggunakan konsep *traveling salesman problem* dengan metode algoritma *particle swarm optimization*. Adapun tahapan yang digunakan sebagai berikut:

1. Mengumpulkan dan mempersiapkan data, dengan tahapan sebagai berikut:
   (a) Mengumpulkan data lokasi syuting setiap *scene*, pemain (*talent*) di setiap *scene*, biaya pemain (*talent*) di setiap *scene*, dan jarak antar lokasi syuting setiap *scene*.
   (b) Menghitung biaya perpindahan antar setiap *scene*.
2. Studi Literatur
   Studi literatur dilakukan untuk mendapatkan informasi dari buku-buku dan catatan penelitian terdahulu mengenai konsep *traveling salesman problem* dengan metode algoritma *particle swarm optimization*. Informasi tersebut akan menjadi pendukung dari ide dalam penyelesaian masalah.
3. Merumuskan masalah dan variabel penelitian
4. Melakukan tahap inisialisasi, dengan tahapan sebagai berikut:
   (a) Menentukan jumlah iterasi maksimum (n).
   (b) Menentukan ukuran *swarm* (N) dan dimensi partikel (d).

(c) Menentukan *learning rates* (c_1 dan c_2).

(d) Menentukan batas atas (X_{max}) dan batas bawah (X_{min}) dari posisi partikel.

(e) Menentukan *velocity* maksimum (V_{max}).

(f) Membangkitkan *velocity* awal $(\boldsymbol{V}_j^{(0)})$ sejumlah N partikel.

(g) Membangkitkan posisi awal $(\boldsymbol{X}_j^{(0)})$ sejumlah N partikel.

(h) Mengevaluasi hasil rute setiap partikel.

(i) Mengevaluasi nilai fungsi *fitness* setiap partikel.

(j) Menentukan $P_{best,j}^{(0)}$ setiap partikel dan $G_{best}^{(0)}$.

5. Melakukan tahap perulangan, dengan tahapan sebagai berikut:

(a) Membangkitkan *inertia weight* $(\omega^{(i)})$.

(b) Membangkitkan $r_1^{(i)}$ dan $r_2^{(i)}$.

(c) Memperbarui *velocity* setiap partikel.

(d) Memperbarui posisi setiap partikel.

(e) Mengevaluasi hasil rute setiap partikel.

(f) Mengevaluasi nilai fungsi *fitness* setiap partikel.

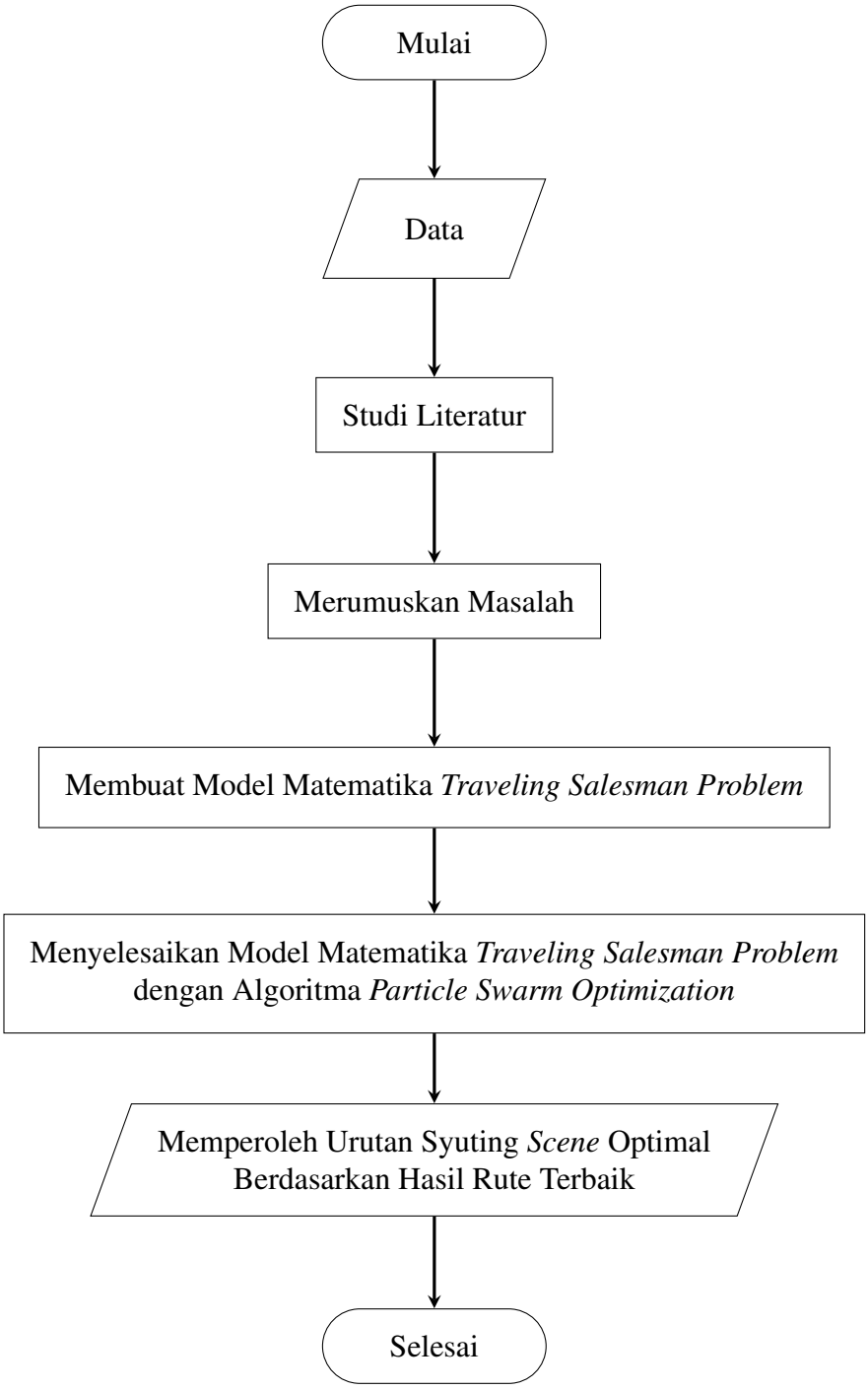
(g) Menentukan $P_{best}^{(i)}$ setiap partikel dan $G_{best}^{(i)}$.

(h) Mengevaluasi kriteria pemberhentian.

6. Menarik kesimpulan urutan syuting *scene* menggunakan hasil rute terbaik dari G_{best}.

3.7 Kerangka Penelitian

Kerangka dari penelitian ini disajikan dalam *flowchart* sebagai berikut:



Gambar 3.1 *Flowchart* Kerangka Penelitian

BAB 4

HASIL DAN PEMBAHASAN

Pada bab ini membahas tentang hasil penelitian yang telah dilakukan yaitu menentukan urutan syuting *scene* film menggunakan algoritma *particle swarm optimization* pada film ”Ketika Adzan Sudah Tidak Lagi Berkumandang”. Adapun yang akan dibahas yaitu deksripsi data, proses algortima *particle swarm optimization*,

4.1

DAFTAR PUSTAKA

Akhirina, T. Y., & Afrizal, T. (2020). Pendekatan Matriks Ketetanggaan Berbobot Untuk Solusi *Minimum Spanning Tree* (MST). *Satuan Tulisan Riset dan Inovasi Teknologi*, *4(3)* (2020), 280–287.

Akpudo, U. E., & Jang-Wook, H. (2020). A Multi-Domain Diagnotics Approach for Solenoid Pumps Based on Discriminative Features. *IEEE Access*, *8* (2020), 175020–175034.

Amin, N. F., Garancang., S., & Abunawas, K. (2023). Konsep umum populasi dan sampel dalam penelitian. *Jurnal Kajian Islam Kontemporer (Jurnal PILAR)*, *14(1)* (2023), 15–31.

Anwar, M., Pratamaningtyas., S., Rosita., Deni., A., Lusono., A., Hermayanti., Ulfa., A. K., Wibowo., M. E. S., Fatmawati., E. R., & Chasanah, A. N. (2024). *Pengambilan keputusan*. Yayasan Cendikia Mulia Mandiri.

Atiqoh, A. N. (2020). *Analisis inertia weight pada algoritma particle swarm optimization (pso) untuk optimalisasi dan pemodelan sistem terhadap persoalan vehicle routing problem with time window (vrptw) (skripsi)*. UIN Sunan Ampel Surabaya.

Azhari, M. K., Cholissodin, I., & Bachtiar., F. A. (2018). Optimasi *Travel Salesman Problem* pada angkutan sekolah dengan algoritma *Particle Swarm Optimization*. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, *2(11)* (2018), 5691–5699.

Chafi, Z. S., & Afrakhte., H. (2021). *Short-Term Load Forecasting Using Neutral Network and Particle Swarm Optimization* (pso) *Algorithm*. *Mathematical Problems in Engineering*, *2021(1)* (2021), 5598267.

Chen, F., Wang., Y.-C., Wang., B., & C.-C. Jay Kuo. (2020). *Graph Representation Learning: a Survey*. *APSIPA Transactions on Signal and Information Processing*, *9(e15)* (2020), 1–21.

Dalyono, Y. P., Herdiani, A., & Rohmawati., A. A. (2017). Penentuan rute pariwisata kota bandung menggunakan algoritma *Particle Swarm Optimization*. *eProcidings of Engineering*, *4(3)* (2017), 4811–4817.

Darmawan, R., Indra., & Surahmat, A. (2022). Optimalisasi *Support Vector Machine* (svm) berbasis *Particle Swarm Optimization* (pso) pada analisis sentimen terhadap *Official Account* ruang guru di twitter. *Jurnal Kajian Ilmiah*, *22(2)* (2022), 143–152s.

Devita, R. N., & Wibawa, A. P. (2020). Teknik-teknik optimasi *Knapsack Problem*. *Sains, Aplikasi, Komputasi dan Teknologi Informasi (SAKTI)*, *2(1)* (2020), 35–40.

Eltamaly, A. M. (2021). *A Novel Strategy for Optimal PSO Control Parameters Determination for PV Energy Systems*. *Sustainability*, *13(2)* (2021), 1008.

Gad, A. (2022). *Particle Swarm Optimization Algorithm and Its Applications: A Systematic Review*. *Achives of Computational Methods in Engineering*, *29* (2022), 2531–2561.

Haren, S. M. (2020). Model manajemen produksi film pendek cerita masa tua. *Jurnal Audiens*, *1(1)* (2020), 107–112.

Jain, M., Vibha, S., Narinder, S., & Satya, B. S. (2022). *An Overview of Variants and Advancements of PSO Algorithm*. *Applied Science*, *12(17)* (2022), 8392.

Jek, J. (2016). *Matematika diskrit dan aplikasinya pada ilmu komputer*.

Kumar, R., & Memoria, M. (2020). *A Review of Memetic Algorithm and its Application in Traveling Salesman Problem*. *International Journal on Emerging Technologies (IJET)*, *11(2)* (2020), 1110–1115.

Kusrahman, N. Y., Purnamasari., I., & Amijaya, F. D. T. (2020). Optimasi *Self-Organizing Map* menggunakan *Particle Swarm Optimization* untuk mengelompokkan desa/kelurahan tertinggal di kabupaten kutai kertanegara provinsi kalimantan timur (studi kasus: Data potensi desa tahun 2018). *Jurnal EKSPONENSIAL*, *11(2)* (2020), 139–144.

Levin, O. (2021). *Discrete mathematics: An open introduction*.

Muhammad., Febrianty., & Sentanu, I. G. E. P. S. (2023). *Manajemen pengambilan keputusan*. Perkumpulan Rumah Cemerlang Indonesia.

Muhardeny, M., Irfani., M. H., & Alie, J. (2023). Penjadwalan mata pelajaran menggunakan algoritma *Particle Swarm Optimization* (pso) pada smpit mufidatul ilmi. *Journal Software Engineering and Computational Intelligence (JSECI)*, *1(1)* (2023), 51–63.

Nasir, A. M., Faisal, & Setiawan, D. (2022). Optimalisasi penjadwalan mata kuliah menggunakan teori pewarnaan graf. *Jurnal Penelitian Matematika dan Pendidikan Matematika (PROXIMAL)*, *5(1)* (2022), 57–69.

Natalia, D., Yundari, & Yudhi. (2019). Optimasi jarak penjemputan penumpang cv. eira saudara menggunakan metode *Particle Swarm Optimization*. *Buletin Ilmiah Math, Stat, dan Terapannya (BIMASTER)*, *8(3)* (2019), 531–538.

Nurdiyanto, T., & Susanti, E. (2019). Efisiensi penggunaan matriks *In-Degree* untuk mengkontruksi *Spanning-Tree* pada graf berarah. *Jurnal Edukasi dan Sains Matematika (JES-MAT)*, *5(1)* (2019), 1–15.

Piotrowski, A. P., Napiorkowski., J. J., & Piotrowska, A. E. (2020). *Population Size in Particle Swarm Optimization*. *Swarm and Evolutionary Computation*, *58* (2020), 100718.

Pratama, V. L., & Darmawan, D. A. (2022). Sistem informasi geografis pencarian rute terdekat bengkel motor di kota surabaya menggunakan algoritma *Bellman-Ford*. *Journal of Informatics and Computer Science (JINACS)*, *3(4)* (2022), 580–599.

Qin, H., Zhang, Z., Lim, A., & Liang., X. (2016). *An Enhanced Branch-and-Bound Algorithm for The Talent Schedulling Problem*. *European Journal of Operational Research*, *250(2)* (2016), 412–426.

Rahayuningsih, S. (2018). *Teori graph dan penerapannya*. Universitas Wisnuwardhana Press Malang (Unidha Press).

Rahimi, A., Arthur, M., Nurfadillah, Amijaya, F. D. T., & Putri., D. F. (2023). Implementasi algoritma genetika dalam penentuan rute terbaik pendistribusian bbm pada spbu yang ada di samarinda. *Prosiding Seminar Nasional Matematika, Statistika, dan Aplikasinya (SNMSA)*, *3(1)* (2023), 196–207.

Ramadhan, M. R., Setianingsih., C., & Dirgantoro., B. (2023). Sistem penjadwalan perangkat listrik dengan metode algoritma *Particle Swarm Optimization* berbasis *Website*. *eProceedings of Engineering*, *10(1)* (2023), 865–872.

Sari, F. F., Pujiarti., T., Hidayat., & Anjosa. (2024). Pembelajaran matematika diskrit mengacu pada teori beban kognitif (*Cognitive Load Theory*). *Jurnal Kajian Pendidikan dan Sosial (DIKSI)*, *5(1)* (2024), 10–17.

Saud, S., Kodaz., H., & Babaoğlu, I. (2018). *Solving Traveling Salesman Problem Using Optimization Algorithms*. *KnE Social Sciences*, *3(1)* (2018), 17–32.

Setiawati, J., Febrilian, M., & Ekawati, D. (2023). Pelabelan total tak reguler sisi pada graf *Direction*, graf *Direction Right*, dan graf *Heart*. *Jurnal Penelitian Matematika dan Pendidikan Matematika (PROXIMAL)*, *6(2)* (2023), 64–72.

Sinaga, R. P., & Marpaung, F. (2023). Perbandingan algoritma *Cheapest Insertion Heuristic* dan *Nearest Neighbor* dalam menyelesaikan *Traveling Salesman Problem*. *Jurnal Riset Rumpun Matematika dan Ilmu Pengetahuan Alam (JURRIMIPA)*, *2(2)* (2023), 238–247.

Wang, Z., Ding., H., Wang., J., Hou., P., Li., A., Yang., Z., & Hu., X. (2022). *Adaptive Guided Salp Swarm Algorithm with Velocity Clamping Mechanism for Solving Optimization Problems*. *Journal of Computational Design and Engineering (CDE), Oxford*, *9(6)* (2022), 2196–2234.

Yurinanda, S., & Rozi, S. (2023). Penerapan pembelajaran berbasis proyek pada mata kuliah matematika diskrit untuk meningkatkan keterampilan mahasiswa dalam memanfaatkan struktur diskrit dalam menyelesaikan masalah. *Jurnal Pendidikan Matematika dan Matematika (ABSIS)*, *5(2)* (2023), 666–679.

Zahro, H. Z., & Wahyuni, F. S. (2020). Optimasi *Particle Swarm Optimization* (pso) untuk penentuan *Base Transceiver System* (bts). *Jurnal MNEMONIC*, *3(1)* (2020), 7–10.

Zdiri, S., Chrouta., J., & Zaafouri., A. (2021). *An Expanded Heterogeneous Particle Swarm Optimization Based on Adaptive Inertia Weight*. *Mathematical Problems in Engineering, Hindawi*, *2021(1)* (2021), 4194263.

LAMPIRAN

Lampiran 1 Contoh