

**OPTIMASI KOMPOSISI BAHAN MAKANAN BAGI PENDERITA
DIABETES MELITUS MENGGUNAKAN METODE
HYBRID ALGORITMA GENETIKA DAN
*SIMULATED ANNEALING***

SKRIPSI



**Oleh :
FEBRIAN KURNIA RAMADHAN
13650020**

**JURUSAN TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK IBRAHIM
MALANG
2020**

**OPTIMASI KOMPOSISI BAHAN MAKANAN BAGI
PENDERITA DIABETES MELITUS MENGGUNAKAN
METODE *HYBRID* ALGORITMA GENETIKA DAN
*SIMULATED ANNEALING***

SKRIPSI

Diajukan Kepada :
Universitas Islam Negeri (UIN) Maulana Malik Ibrahim Malang
Untuk Memenuhi Salah Satu Persyaratan Dalam
Memperoleh Gelar Sarjana Komputer (S.Kom)

Oleh :
FEBRIAN KURNIA RAMADHAN
13650020

**JURUSAN TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI MAULANA MALIK
IBRAHIM MALANG
2020**

LEMBAR PERSETUJUAN
OPTIMASI KOMPOSISI BAHAN MAKANAN BAGI
PENDERITA DIABETES MELITUS MENGGUNAKAN
METODE *HYBRID* ALGORITMA GENETIKA DAN
SIMULATED ANNEALING

SKRIPSI

Oleh :
FEBRIAN KURNIA RAMADHAN
13650020

Telah Diperiksa dan Disetujui untuk Diuji
Tanggal : 29 Desember 2020

Dosen Pembimbing I



Fatchurrochman, M.Kom
NIP. 19700731 200501 1 002

Dosen Pembimbing II



Roro Inda Melani, S.Kom., M.Sc
NIP. 19780925 200501 2 008

Mengetahui,
Ketua Jurusan Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang



Dr. Wahyo Crysdian
NIP. 19740424 200901 1 008

LEMBAR PENGESAHAN
OPTIMASI KOMPOSISI BAHAN MAKANAN BAGI
PENDERITA DIABETES MELITUS MENGGUNAKAN
METODE *HYBRID* ALGORITMA GENETIKA DAN
SIMULATED ANNEALING

SKRIPSI

Oleh :

FEBRIAN KURNIA RAMADHAN
13650020

Telah Dipertahankan di Depan Dewan Penguji
dan Dinyatakan Diterima Sebagai Salah Satu Persyaratan
untuk Memperoleh Gelar Sarjana Komputer (S.Kom)
Pada Tanggal 29 Desember 2020

Susunan Dewan Penguji

Tanda Tangan

Penguji Utama : (Suhartono, M.Kom)
NIP. 19680519 200312 1 001

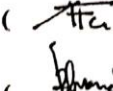
Ketua Penguji : (Ajlil Hanani, M.T)
NIDT. 19840731 20160801 1 076

Sekretaris Penguji : (Fatchurrochman, M.Kom)
NIP. 19700731 200501 1 002

Anggota Penguji : (Roro Inda Melani, S.Kom., M.Sc)
NIP. 19780925 200501 2 008

()

()

()

()

Mengetahui,
Ketua Jurusan Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Maulana Malik Ibrahim Malang



Dr. Sahyo Crysdian
NIP. 19740424 200901 1 008

SURAT PERNYATAAN

Yang Bertanda Tangan dibawah ini saya:

Nama : Febrian Kurnia Ramadhan
NIM : 13650020
Fakultas/Jurusan : Sains dan Teknologi/Teknik Informatika
Alamat : Perum Keboncandi Permai Blok E-13
Gondangwetan Pasuruan

Menyatakan bahwa “Skripsi” yang saya buat untuk memenuhi persyaratan kelulusan pada jurusan Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri (UIN) Maulana Malik Ibrahim Malang, dengan judul:

“OPTIMASI KOMPOSISI BAHAN MAKANAN BAGI PENDERITA DIABETES MELITUS MENGGUNAKAN METODE *HYBRID* ALGORITMA GENETIKA DAN *SIMULATED ANNEALING*”

Adalah hasil karya sendiri, bukan “Duplikat” dari karya orang lain.

Selanjutnya apabila dikemudian hari ada “Klaim” dari pihak lain, bukan menjadi tanggungjawab Dosen Pembimbing dan atau pihak Fakultas Teknik Informatika, tetapi menjadi tanggungjawab saya sendiri.

Demikian surat pernyataan ini saya buat dengan sebenar-benarnya dan tanpa paksaan dari siapapun.

Malang, 29 Desember 2020



Hormat saya,

Febrian Kurnia Ramadhan

NIM: 13650020

MOTTO

Jangan jadikan kegagalan sebagai musuhmu, karena kegagalanlah yang membuatmu menjadi lebih kuat.

Sesungguhnya sesudah kesulitan itu ada kemudahan (QS Asy-Syarh: 6)

“Gantungkan cita-citamu setinggi langit! Bermimpilah setinggi langit, jika engkau jatuh, maka engkau akan jatuh diantara bintang-bintang” – Ir. Soekarno

HALAMAN PERSEMBAHAN

Alhamdulillah, puji syukur kehadiran Allah yang Maha Agung dan Maha Penyayang, atas rahmat dan karunianya yang telah menjadikan aku manusia yang senantiasa berpikir, berilmu, beriman, serta bersabar dalam menjalani kehidupan ini.

Semoga keberhasilan ini menjadi langkah awal untuk meraih kesuksesan.

Aku persembahkan karya tulis ini untuk yang teristimewa Ibunda Yayuk Fauziah, Terima kasih atas segala pengorbanan dan kasih sayang yang tiada tara, serta telah memberikan kesempatan kepadaku untuk mengenyam pendidikan sampai perguruan tinggi.

Serta seluruh keluarga penulis yang telah memotivasi, menginspirasi, mendoakan dan selalu mendukungku sampai saat ini.

KATA PENGANTAR

Alhamdulillah, segala puji dan syukur kehadiran Allah SWT, karena atas rahmat, taufiq, dan hidayah-Nya penelitian ini dapat terselesaikan dengan judul “Optimasi Komposisi Bahan Makanan Bagi Penderita Diabetes Melitus Menggunakan Metode Hybrid Algoritma Genetika dan Simulated Annealing”.

Shalawat serta salam semoga tetap tercurahkan kepada junjungan kita Nabi besar Muhammad SAW yang telah membimbing dari kegelapan menuju jalan kebaikan, yakni Din al-Islam.

Penulis menyadari bahwa dalam penyusunan tugas akhir skripsi ini tidak akan berhasil dengan baik tanpa adanya bimbingan, dukungan dan sumbangan pemikiran dari berbagai pihak. Pada kesempatan ini penulis menyampaikan terima kasih yang tak terhingga kepada:

1. Bapak Prof. Dr. Abdul Haris, M.Ag, selaku Rektor Universitas Islam Negeri Maulana Malik Ibrahim Malang.
2. Ibu Dr. Sri Harini, M.Si, selaku Dekan Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang.
3. Bapak Dr. Cahyo Crysdian, selaku Ketua Jurusan Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang.
4. Bapak Fatchurrochman, M.Kom dan Ibu Roro Inda Melani, S.Kom., M.Sc selaku Dosen Pembimbing yang telah memberikan masukan, saran serta bimbingan dalam proses penulisan skripsi.
5. Bapak Suhartono, M.Kom dan Bapak Ajib Hanani, M.T selaku Dosen Penguji yang telah memberikan masukan dan saran dalam penulisan skripsi ini.
6. Bapak dan Ibu Dosen Teknik Informatika Universitas Islam Negeri Maulana Malik Ibrahim Malang yang telah memberikan ilmunya dengan tulus dan ikhlas.

7. Ibu drg. Novita Kusuma Wardhani selaku Kepala Klinik Al Fattah Pasuruan, yang telah memberikan izin untuk melaksanakan penelitian.
8. Ibu Yayuk Fauziah yang senantiasa memberikan doa dan dukungan yang tiada henti-hentinya untuk kelancaran dalam mengerjakan tugas akhir skripsi ini.
9. Teman-teman Fakultas Teknik Informatika angkatan 2013 yang telah membantu dan memberikan dukungan.
10. Dan seluruh pihak yang terlibat secara langsung maupun tidak langsung yang tidak bisa disebutkan satu-persatu.

Akhir kata, dengan segala kerendahan hati penulis menyadari bahwa penulisan skripsi ini masih jauh dari kata sempurna. Oleh karena itu penulis mengharapkan kritik dan saran yang konstruktif demi kesempurnaan penulisan ini. Penulis berharap semoga karya yang sederhana ini dapat bermanfaat dengan baik bagi semua pihak. Aamiin Yaa Rabbal 'Alamin.

Malang, 29 Desember 2020

Penulis

DAFTAR ISI

HALAMAN JUDUL	i
LEMBAR PERSETUJUAN	ii
LEMBAR PENGESAHAN	iii
PERNYATAAN KEASLIAN TULISAN	iv
MOTTO	v
HALAMAN PERSEMBAHAN	vi
KATA PENGANTAR.....	viii
DAFTAR ISI.....	ix
DAFTAR TABEL	xii
DAFTAR GAMBAR.....	xiii
ABSTRAK	xiv
ABSTRACT	xv
SMDKNK.....	xvi
BAB I PENDAHULUAN.....	1
1.1 Latar Belakang.....	1
1.2 Identifikasi Masalah	3
1.3 Tujuan Masalah	4
1.4 Batasan Masalah.....	4
1.5 Manfaat.....	4
BAB II STUDI PUSTAKA.....	5
2.1 Diabetes Melitus	5
2.2 Berat Badan Ideal	5
2.3 Daftar Bahan Makanan.....	6
2.4 Gizi	10
2.4.1 Energi	11
2.4.2 Protein	14
2.4.3 Lemak.....	14
2.4.4 Karbohidrat	15
2.5 Algoritma Genetika	15
2.5.1 Parameter Algoritma Genetika	16
2.5.2 Struktur Algoritma Genetika	17
2.6 <i>Simulated Annealing</i>	20

2.7 Hybrid Algoritma Genetika dan <i>Simulated Annealing</i>	22
2.8 Penelitian Terkait.....	23
BAB III METODOLOGI PENELITIAN	29
3.1 Studi Literatur.....	30
3.2 Analisis Kebutuhan.....	30
3.2.1 Data	30
3.2.2 Spesifikasi Sistem	31
3.2.3 Spesifikasi Pengguna	31
3.2.4 Lingkungan Operasi	31
3.3 Pengumpulan Data.....	31
3.4 Perancangan Sistem.....	32
3.4.1 Siklus Algoritma Genetika.....	34
3.4.2 Siklus Algoritma <i>Simulated Annealing</i>	37
3.4.3 Siklus <i>Hybrid</i> Algoritma Genetika dan <i>Simulated Annealing</i> ..	39
3.5 Contoh Persoalan.....	41
3.6 Contoh Perhitungan Manual.....	42
3.7 Implementasi	52
3.7.1 Inisialisasi parameter	52
3.7.2 Inisialisasi Populasi Awal.....	52
3.7.3 Proses Crossover.....	53
3.7.4 Proses Mutation	55
3.7.5 Proses Populasi Gabungan.....	56
3.7.6 Proses Penalti.....	58
3.7.7 Proses Fitness.....	58
3.7.8 Proses Evaluasi dan Seleksi.....	59
3.7.9 Proses <i>Simulated Annealing</i>	60
3.7.10 Proses Neighborhood.....	62
BAB IV UJI COBA DAN PEMBAHASAN	64
4.1 Hasil Pengujian.....	64
4.1.1 Input Data dan Parameter	64
4.1.2 Hasil Pengujian Populasi Awal	65
4.1.3 Hasil Pengujian <i>Crossover</i>	65
4.1.4 Hasil Pengujian <i>Mutation</i>	65
4.1.5 Hasil Pengujian Populasi Gabungan.....	66
4.1.6 Hasil Pengujian Seleksi dan <i>Simulated Annealing</i>	66
4.1.7 Hasil Pengujian Individu Terbaik.....	67
4.1.8 Output Sistem	67
4.1.9 Hasil Pengujian dan Analisis Ukuran Populasi	68
4.1.10 Hasil Pengujian dan Analisis Jumlah Generasi	70
4.1.11 Hasil Pengujian dan Analisis Kombinasi C_r dan M_r	71
4.1.12 Hasil Pengujian dan Analisis Temperatur awal (T_0)	75

4.1.13 Hasil Pengujian dan Analisis Alpha (α)	76
4.2 Pembahasan	78
BAB V KESIMPULAN DAN SARAN.....	79
5.1 Kesimpulan	79
5.2 Saran	79
DAFTAR PUSTAKA	

DAFTAR TABEL

2.1 Komposisi Bahan Makanan Pokok	7
2.2 Komposisi Bahan Makanan Nabati.....	8
2.3 Komposisi Bahan Makanan Hewani	9
2.4 Komposisi Bahan Makanan Sayuran	10
2.5 Komposisi Bahan Makanan Buah	10
2.6 Kategori Aktivitas Fisik atau Pekerjaan.....	13
2.7 Kategori Berat Badan	14
2.8 Pemetaan <i>Physical Annealing</i> ke <i>Simulated Annealing</i>	21
3.1 Data Penderita Diabetes Melitus.....	41
3.2 Representasi Kromosom	43
3.3 Kromosom Awal	43
3.4 Detail kromosom Awal Pada Individu P1.....	44
3.5 Populasi Awal	46
3.6 Detail Populasi Awal	47
3.7 Proses <i>Crossover</i>	47
3.8 Proses Mutasi P2	48
3.9 Proses Mutasi P3.....	48
3.10 Populasi Gabungan	49
3.11 Hasil Seleksi.....	49
3.12 Hasil Optimasi <i>Hybrid</i> Algoritma Genetika dan <i>Simulated Annealing</i>	51
4.2 Hasil Pengujian Ukuran Populasi (<i>Popsiz</i> e)	69
4.3 Hasil Pengujian Banyaknya Generasi	71
4.4 Hasil Pengujian Kombinasi C_r dan M_r	72
4.4 Hasil Pengujian Temperatur awal (T_0)	74
4.5 Hasil Pengujian Alpha (α).....	75
4.6 Perbandingan Nilai <i>Fitness</i> Algoritma Genetika dan Hybrid GA-SA.....	77

DAFTAR GAMBAR

2.1 Siklus Algoritma Genetika.....	18
3.1 Alur Penelitian.....	29
3.2 Perancangan Sistem	32
3.3 Flowchart Proses Algoritma Genetika	33
3.4 Flowchart Proses <i>Simulated Annealing</i>	38
3.5 Diagram Alur Hybrid GA-ZA.....	40
3.6 Proses <i>Neighborhood-1</i>	50
3.7 Kode Program untuk Inisialisasi Populasi Awal.....	52
3.8 Kode Program untuk Crossover	54
3.9 Kode Program Insertion Mutation	55
3.4 Kode Program Populasi Gabungan	57
3.5 Kode Program Untuk Penalti	58
3.6 Kode Program Perhitungan Fitness.....	58
3.7 Kode Program Untuk Evaluasi dan Seleksi	59
3.8 Kode Program untuk Proses <i>Simulated Annealing</i>	61
3.9 Kode Program untuk proses <i>Neighborhood</i>	63
4.1 Input Data Pasien Dan Parameter	65
4.2 Hasil Populasi Awal.....	65
4.3 Hasil Proses Crossover.....	66
4.4 Hasil Proses Mutation	66
4.5 Hasil Populasi Gabungan	67
4.6 Hasil Seleksi dan <i>Simulated Annealing</i>	67
4.7 Hasil Individu Terbaik	68
4.8 Output Hasil Hybrid Algoritma Genetika dan <i>Simulated Annealing</i>	68
4.9 Grafik Rata-rata Fitness Berdasarkan Ukuran Populasi.....	70
4.10 Grafik Rata-rata Fitness berdasarkan Banyaknya Generasi.....	72
4.11 Grafik Rata-rata Fitness Berdasarkan Kombinasi C_r dan M_r	73
4.12 Grafik Rata-rata Fitness Berdasarkan Temperatur Awal (T_0)	75
4.13 Grafik Rata-rata Fitness Berdasarkan Alpha (α).....	76

ABSTRAK

Ramadhan, Febrian Kurnia. 2020. **Optimasi Komposisi Bahan Makanan Bagi Penderita Diabetes Melitus Menggunakan Metode *Hybrid Algoritma Genetika Dan Simulated Annealing***. Skripsi. Jurusan Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang.

Pembimbing : (1) Fatchurrochman, M.Kom. (2) Roro Inda Melani, S.Kom., M.Sc

Kata Kunci : diabetes melitus, algoritma genetika, *simulated annealing*

Keseimbangan pola makan pada penderita diabetes melitus untuk memenuhi kebutuhan energi harus selalu terjaga. Oleh karena itu diperlukan komposisi makanan yang sesuai untuk memenuhi kebutuhan tersebut, sehingga dapat membantu penderita diabetes melitus untuk menentukan menu makannya. Permasalahan komposisi bahan makanan untuk penderita diabetes melitus dapat diselesaikan dengan hybrid algoritma genetika dan simulated annealing. Proses penyelesaian permasalahan ini menggunakan metode crossover yaitu one-cut point, mutasi menggunakan insertion mutation, seleksi menggunakan elitism, dan simulated annealing. Berdasarkan hasil pengujian dari parameter yang digunakan dalam sistem optimasi bahan makanan menggunakan Hybrid algoritma genetika dan simulated annealing didapatkan nilai parameter terbaik jumlah populasi = 40, jumlah generasi = 75, Cr = 0,5, Mr = 0,5, T0 = 1, Tn = 0,5, alpha = 0,1. Penggabungan kedua metode ini menghasilkan solusi yang lebih baik dibanding hanya dengan menggunakan metode algoritma genetika saja. Didapatkan hasil rata-rata nilai fitness yang dihasilkan dari pengujian menggunakan algoritma genetika sebesar 0.771567577, sedangkan dengan *hybrid* algoritma genetika dan *simulated annealing* sebesar 0.829818337.

ABSTRACT

Ramadhan, Febrian Kurnia. 2020. **Optimization of Food Ingredients for Diabetes**

Mellitus Patients Using Hybrid Genetic Algorithm and Simulated Annealing Methods. Thesis. Department of Informatics, Faculty of Science and Technology Maulana Malik Ibrahim State Islamic University of Malang.

Supervisor : (1) Fatchurrochman, M.Kom. (2) Roro Inda Melani, S.Kom., M.Sc

Keywords: diabetes mellitus, genetic algorithm, *simulated annealing*

Eating balance in people with diabetes mellitus to meet energy needs must always be maintained. Therefore we need a suitable food composition to meet those needs, so that it can help people with diabetes mellitus to determine their diet. The problem of food composition for diabetics can be solved by hybrid genetic algorithms and simulated annealing. The process of solving this problem uses a crossover method that is one-cut point, mutation using insertion mutation, selection using elitism, and simulated annealing. Based on the test results of the parameters used in the food ingredients optimization system using Hybrid genetic algorithms and simulated annealing, the best parameter values are population = 40, number of generations = 75, Cr = 0.5, Mr = 0.5, T0 = 1, Tn = 0.5, alpha = 0.1. The combination of these two methods produces a better solution than just using the genetic algorithm method. Obtained the results of the average fitness value generated from testing using a genetic algorithm of 0.771567577, while with a hybrid genetic algorithm and simulated annealing of 0.829818337.

ملخص

رمضان ، فبريان كورنيا. 2020. تحسين المكونات الغذائية لمرض السكري مرضى
الدهن باستخدام الخوارزمية الجينية الهجينة والتلين المحاكي طرق. مقال.
قسم المعلوماتية بكلية العلوم والتكنولوجيا جامعة مولانا مالك إبراهيم الإسلامية
الحكومية في مالانج
المستشار: (1) فتور الرحمن (2) رورو إندا ميلاني

الكلمات المفتاحية: داء السكري ، الخوارزمية الجينية ، التلدين بالمحاكاة

يجب دائماً الحفاظ على توازن النظام الغذائي لدى مرضى السكري لتلبية احتياجات الطاقة. لذلك نحن بحاجة إلى تركيبة غذائية مناسبة لتلبية هذه الاحتياجات ، بحيث يمكن أن تساعد الأشخاص المصابين بداء السكري في تحديد نظامهم الغذائي. يمكن حل مشكلة تكوين الغذاء لمرضى السكري باستخدام خوارزمية وراثية هجينة وتحمية محاكاة. تستخدم عملية حل هذه المشكلة طريقة التقاطع ، أي النقطة المقطوعة ، والتلدين المحاكي. elitism والطفرة باستخدام طفرة الإدراج ، والاختيار باستخدام بناءً على نتائج اختبار المسمات المستخدمة في نظام تحسين المواد الغذائية باستخدام الخوارزميات الوراثة الهجينة والتلين المحاكي ، فإن أفضل قيم المسمات هي السكان = $Mr. = 0.5$ ، $T0 = 1$ ، $Mr = 0.5$ ، $Cr = 0.5$ ، عدد الأجيال = 75 ، ألفا = 0.1. يؤدي الجمع بين هاتين الطريقتين إلى حل أفضل من استخدام 0.5 طريقة الخوارزمية الجينية وحدها. متوسط نتيجة قيمة الملاءمة من الاختبار باستخدام الخوارزمية الجينية هو 0.771567577 ، بينما الخوارزمية الجينية الهجينة والتلدين المحاكي 0.829818337.

BAB I

PENDAHULUAN

Pada bab ini akan dijelaskan tentang latar belakang, identifikasi masalah, tujuan penelitian, manfaat penelitian, batasan masalah dari penelitian yang akan dilakukan.

1.1 Latar Belakang

Pola makan masyarakat sekarang ini cenderung instan sehingga kurang memperhatikan kandungan gizi yang cukup terutama kadar gula atau glukosa yang berlebih dalam bahan makanannya. Jika hal ini tetap berlangsung setiap hari akan mengakibatkan penumpukan glikogen yang disimpan dalam tubuh. Hal ini meningkatkan kemungkinan terjangkit penyakit Diabetes Melitus (DM).

Mengonsumsi karbohidrat yang tinggi atau berlebihan dapat meningkatkan resiko terkena diabetes militus sebanyak 10,28 kali. Selain itu, orang dengan konsumsi lemak yang tinggi beresiko 5,25 kali lebih besar untuk terkena diabetes, dibandingkan dengan orang yang konsumsi lemaknya rendah dan mengonsumsi karbohidrat yang tinggi akan semakin meningkatkan risiko diabetes melitus jika diiringi asupan serat yang rendah (Garnita, 2012).

Mengonsumsi karbohidrat yang tinggi atau berlebihan dapat meningkatkan resiko terkena diabetes militus sebanyak 10,28 kali. Selain itu, orang dengan konsumsi lemak yang tinggi beresiko 5,25 kali lebih besar untuk terkena diabetes, dibandingkan dengan orang yang konsumsi lemaknya rendah dan mengonsumsi karbohidrat yang tinggi akan semakin meningkatkan risiko diabetes melitus jika diiringi asupan serat yang rendah (Garnita, 2012). Seperti halnya pada dalil Al-Qur'an yang terkandung pada QS : Al-A'raf ayat 31, yang menunjukkan bahwa Allah menganjurkan manusia bahwa jangan makan dan minum yang berlebihan.

﴿يٰۤاٰدَمُ خُذُوْا زِيْنَتَكُمْ عِنْدَ كُلِّ مَسْجِدٍ وَكُلُوْا وَاشْرَبُوْا وَلَا تُسْرِفُوْا اِنَّهٗ لَا يُحِبُّ الْمُسْرِفِيْنَ ۝۳۱﴾

Artinya : “Hai anak Adam, pakailah pakaianmu yang indah di setiap (memasuki) masjid, makan dan minumlah dan jangan berlebih-lebihan. Sesungguhnya Allah tidak menyukai orang-orang yang berlebih-lebihan.” (QS. Al-A’raf: 31).

Dalam tafsir Ibnu Katsir menjelaskan bahwa Firman Allah Ta’ala selanjutnya: *وَكُلُوْا وَاشْرَبُوْا* (“Makan dan minumlah....”). Sebagian ulama salaf mengatakan, Allah Ta’ala telah menyatukan seluruh pengobatan pada setengah ayat ini, *وَكُلُوْا وَاشْرَبُوْا وَلَا تُسْرِفُوْا* “Makan dan minumlah dan janganlah kamu berlebih-lebihan.” Imam al-Bukhari meriwayatkan, Ibnu ‘Abbas berkata: “Makan dan berpakaianlah sesuka kalian, asalkan engkau terhindar dari dua sifat: berlebih-lebihan dan sombong.” (Abdul Ghoffar, 2003).

Penyakit diabetes melitus memiliki beberapa gejala seperti sering merasa lapar, sering merasa haus, sering buang air kecil dan penurunan berat badan secara drastis. Tandra mengatakan diabetes adalah penyakit yang memiliki pertumbuhan terpesat dan telah menyebabkan kematian lebih banyak dibandingkan dengan penyakit lain dan merupakan penyebab kematian no 4 terbesar di dunia. Sebanyak 4,8 juta penduduk di dunia pada tahun 2012 mengalami kematian dan setiap menit terdapat 6 orang yang meninggal dikarenakan menderita penyakit diabetes (Maryamah dkk., 2017).

Pengobatan penyakit diabetes melitus bisa dilakukan dengan beberapa cara diantaranya diet diabetes, latihan fisik, penyuluhan kesehatan masyarakat, obat *hipoglikemi* (OAD dan insulin), dan cangkok pankreas. Mengatur pola makan yang baik adalah dengan cara memberikan modifikasi diet sesuai dengan keadaan penderita diabetes, serta dianjurkan mengonsumsi makanan yang mengandung serat dan tidak mengonsumsi gula murni (Rahmawati, 2011).

Modifikasi diet dilakukan guna untuk membantu memperoleh bahan makanan seperti bahan makanan pokok, bahan makanan nabati, bahan makanan

hewani, bahan makanan sayuran dan bahan makanan buah yang diperlukan pasien diabetes melitus dengan harga terjangkau dan ketersediaan bahannya di sekitaran. Modifikasi diet mengganti bahan makanan kaya akan kandungan dan harganya mahal dengan bahan makanan yang memiliki kandungan yang sama namun dengan harga yang terjangkau.

Pada penelitian sebelumnya mengenai optimasi komposisi makanan yang dilakukan pada penderita penyakit diabetes melitus menggunakan algoritma genetika mendapatkan hasil kombinasi makanan yang dibutuhkan oleh penderita diabetes melitus dan biaya yang minimum dengan ukuran populasi sebesar 160 individu dengan rata-rata fitness sebesar 0.0774665, 100 generasi dengan rata-rata fitness sebesar 0.0774665 dan kombinasi $cr = 0.4$ dan $mr = 0.6$ dengan rata-rata fitness sebesar 0.0780737. Nilai fitness merupakan suatu ukuran baik tidaknya suatu solusi yang dinyatakan sebagai satu individu. Output yang dihasilkan berupa list makan pagi, siang, dan malam (Rianawati & Mahmudy, 2015). Namun untuk memperbesar tingkat keakuratan dari metode algoritma genetika yang sudah digunakan dalam penelitian sebelumnya, akan dilakukan pengolahan data lebih lanjut dengan metode *simulated annealing* yang nantinya akan diharapkan ada peningkatan keakuratan nilai fitness pada optimasi bahan makanan untuk penderita diabetes melitus (Musanah, 2017).

Seperti pada hadist yang diriwayatkan oleh Imam Bukhari di dalam shahihnya, dari sahabat Abu Hurairah bahwasannya Nabi bersabda,

مَا أَنْزَلَ اللَّهُ دَاءً إِلَّا أَنْزَلَ لَهُ شِفَاءً

Artinya : “Tidaklah Allah turunkan penyakit kecuali Allah turunkan pula obatnya”.

Optimasi sebagai sebuah persoalan komputasi, tidaklah hanya sekedar bahwa optimasi selain dapat dikerjakan secara prosedural dan rigid dalam langkah-langkah yang dikerjakan dengan pensil dan kertas serta menghasilkan sebuah keputusan tentang yang mana yang terbaik atau optimum (Munirah &

Subanar, 2017). Optimasi dalam pemilihan komposisi bahan makanan bagi penderita diabetes sangat berguna bagi penderita diabetes.

Maka dari itu untuk menyelesaikan permasalahan pemilihan komposisi bahan makanan bagi penderita diabetes melitus. Dilakukan penelitian dengan judul optimasi komposisi bahan makanan bagi penderita diabetes melitus menggunakan metode *hybrid* algoritma genetika dan *simulated annealing* yang dapat membantu untuk mempermudah penderita diabetes melitus dalam memilih komposisi bahan makanan yang memenuhi nutrisi yang baik dan gizi yang cukup.

1.2 Identifikasi Masalah

1. Bagaimana mengukur nilai *fitness* yang dihasilkan dari *hybrid* algoritma genetika dan *simulated annealing* untuk optimasi komposisi bahan makanan bagi penderita diabetes melitus.
2. Apakah sistem yang dibangun mampu menghasilkan komposisi bahan makanan yang tepat bagi penderita diabetes melitus.
3. Apakah *simulated annealing* mampu meningkatkan hasil optimasi dari algoritma genetika.

1.3 Tujuan Penelitian

1. Mengukur nilai *fitness* yang dihasilkan dari *hybrid* algoritma genetika dan *simulated annealing* untuk optimasi komposisi bahan makanan bagi penderita diabetes melitus.
2. Mengetahui mampu tidaknya sistem menghasilkan komposisi bahan makanan yang tepat bagi penderita diabetes melitus.
3. Mengetahui mampu tidaknya *simulated annealing* mampu meningkatkan hasil optimasi dari algoritma genetika.

1.4 Manfaat Penelitian

1. Untuk penderita diabetes melitus, penelitian ini diharapkan dapat membantu memenuhi kebutuhan gizi yang sesuai kebutuhan dengan biaya atau harga yang minimal.

2. Untuk pembaca mendapatkan wawasan dalam pengimplementasian dari metode Algoritma Genetika dan *Simulated Annealing* pada aplikasi sistem ini.

1.5 Batasan Masalah

1. Bahan makanan sejumlah 75 bahan yang diambil dari Daftar Komposisi Bahan Makanan (DKBM)
2. Asupan gizi hanya meliputi energi, protein, karbohidrat, dan lemak
3. Data Pasien diambil dari Klinik Al-Fattah Kota Pasuruan

BAB II

STUDI PUSTAKA

Pada bab ini akan dijelaskan mengenai kajian pustaka, algoritma yang digunakan serta literatur terkait dengan penelitian yang dilakukan.

2.1 Diabetes Melitus

Diabetes melitus merupakan kumpulan gejala yang timbul pada diri seseorang yang disebabkan oleh adanya peningkatan glukosa darah akibat kekurangan insulin baik absolut maupun relatif (Rianawati dkk., 2015). Pada orang yang sehat, karbohidrat dalam makanan akan diubah menjadi glukosa yang akan didistribusikan ke seluruh sel tubuh untuk dijadikan energi dengan bantuan insulin. Pada orang yang menderita diabetes melitus, glukosa sulit masuk ke dalam sel karena sedikit atau tidak adanya zat insulin dalam tubuh. Akibatnya kadar glukosa dalam darah menjadi tinggi yang nantinya dapat memberikan efek samping yang bersifat negatif atau merugikan (WHO, 2009).

Diabetes melitus merupakan penyakit yang dapat dimiliki orang dalam jangka waktu yang panjang (penyakit menahun) karena tidak mudah untuk menurunkan kadar gula dalam darah sehingga membutuhkan penanganan yang tepat. Diabetes adalah kondisi tubuh tidak dapat memproduksi insulin dengan benar atau tubuh mengalami kekurangan insulin sehingga glukosa dalam darah menumpuk (Maryamah dkk., 2017).

2.2 Berat Badan Ideal

Untuk menentukan jumlah kalori yang dibutuhkan penderita Diabetes melitus diantaranya adalah dengan memperhitungkan kebutuhan kalori basal yang besarnya 25-30 kalori/kgBB ideal, ditambah atau dikurangi bergantung pada beberapa faktor seperti jenis kelamin, umur, aktivitas dan berat badan (Pranoto dkk., 2011).

Perhitungan Berat Badan Ideal (BBI) dengan rumus Brocca yang dimodifikasi pada Persamaan 2.1.

$$\text{BBI} = 0,9 \times (\text{TB} - 100) \times 1\text{kg} \quad (2.1)$$

Pada laki-laki yang tingginya < 160 cm atau perempuan yang tingginya < 150 cm berlaku Persamaan 2.2.

$$\text{BBI Khusus} = (\text{TB} - 100) \times 1\text{ kg} \quad (2.2)$$

Keterangan :

TB = Tinggi Badan (cm)

BBI = Berat Badan Ideal (kg)

Perhitungan berat badan ideal menurut Indeks Massa Tubuh (IMT) dapat di hitung dengan Persamaan 2.3 (Pranoto dkk., 2011).

$$\text{IMT} = \text{BB} / (\text{TB})^2 \quad (2.3)$$

Keterangan :

BB = Berat Badan (kg)

TB = Tinggi Badan (m)

2.3 Daftar Komposisi Bahan Makanan

Komposisi bahan makanan yang terdiri dari sumber pokok, sumber nabati, sumber hewani, sayuran dan buah sudah disesuaikan untuk penderita diabetes melitus sejumlah 75 komposisi bahan makanan yang diambil dari Daftar Komposisi Bahan Makanan (DKBM) Indonesia yang telah diterbitkan oleh Depkes tahun 2005. Daftar komposisi bahan makanan pokok ditunjukkan pada Tabel 2.1.

Tabel 2.1 Komposisi Bahan Makanan Pokok

No.	Sumber	Energi (kkal)	Protein (kkal)	Lemak (kkal)	Karbohidrat (kkal)
1	Singkong	146	1.2	0.3	34.7
2	Beras jagung	345	9.1	2	76.5
3	Beras ketan hitam	356	7	0.7	78

4	Beras ketan putih	362	6.7	0.7	79.4
5	Beras merah tumbuk	352	7.3	0.9	76.2
6	Ubi jalar putih	124	1,9	0.8	28
7	Ubi jalar kuning	114	0.8	0.5	26.7
8	Ubi jalar merah	123	1.8	0.7	27.9
9	Jagung kuning	366	9.8	7.3	69.1
10	Kentang	83	2	0.1	19.1
11	Roti kukus	249	5.1	2.1	52.5
12	Sagu ambon	338	0.6	0.3	83.1
13	Soun	351	11.3	0	76.5
14	Bihun	360	4.7	0.1	82.1
15	Talas	98	1.9	0.2	23.7

Daftar komposisi bahan makanan yang berasal dari sumber nabati sebanyak 15 komposisi bahan makanan ditunjukkan pada Tabel 2.2.

Tabel 2.2 Komposisi Bahan Makanan Nabati

No.	Sumber	Energi (kkal)	Protein (kkal)	Lemak (kkal)	Karbohidrat (kkal)
1	Tempe kedelai	201	20.8	8.8	13.5
2	Tahu	68	7.8	4.6	1.6
3	Coklat pahit (batang)	504	5.5	52.9	29.2
4	Kacang ijo	345	23.8	1.4	60.2
5	Kacang merah	336	23.1	1.7	59.5
6	Kacang kedelai	381	40.4	16.7	24.9
7	Kacang kapri	98	6.7	0.4	17.7
8	Melinjo	66	5	0.7	13.3
9	Macaroni	363	8.7	0.4	78.7
10	Kecap	46	5.7	1.3	9
11	Toge	255	6.6	3.4	4.8

12	Kecipir	35	2.9	0.2	5.8
13	Koro	332	24	3	55
14	Kluwak	273	10	24	13.5
15	Kacang Arab	330	23.8	1.4	60.2

Daftar komposisi bahan makanan yang berasal dari sumber hewani sebanyak 15 komposisi bahan makanan ditunjukkan pada Tabel 2.3.

Tabel 2.3 Komposisi Bahan Makanan Hewani

No.	Sumber	Energi (kkal)	Protein (kkal)	Lemak (kkal)	Karbohidrat (kkal)
1	Bandeng	129	20	4.8	0
2	Ikan gambus	74	25.2	1.7	0
3	Ikan asin kering	193	42	1.5	0
4	Ikan mas	86	42	1.5	0
5	Ikan mujair	89	18.7	1	0
6	Ikan tongkol	117	23.2	2.7	0
7	Kerang	59	8	1.1	3.6
8	Telur ayam	162	12.8	11.5	0.7
9	Telur bebek	189	13.1	14.3	0.8
10	Telur puyuh	168	12.3	12.7	1.2
11	Teri	74	10.3	1.4	4.1
12	Udang segar	91	21	0.2	0.1
13	Belut air tawar	82	6.7	1	10.9
14	Cumi-cumi	75	16.1	0.7	0.1
15	Kepiting	151	13.8	3.8	14.1

Daftar komposisi bahan makanan yang berasal dari sayuran sebanyak 15 komposisi bahan makanan ditunjukkan pada Tabel 2.4.

Tabel 2.4 Komposisi Bahan Makanan Sayuran

No.	Sumber	Energi (kkal)	Protein (kkal)	Lemak (kkal)	Karbohidrat (kkal)
1	Bayam	36	3.5	0.5	6.5
2	Kool merah	24	1.4	0.2	5.3
3	Kool kembang	25	2.4	0.2	4.9
4	Sawi	22	2.3	0.3	4
5	Daun pepaya	79	8	2	11.9
6	Daun singkong	73	6.8	1.2	13
7	Kangkung	29	3	0.3	5.4
8	Seledri	20	1	0.1	4.6
9	Rebung	27	2.6	0.3	5.2
10	Kacang panjang	44	2.7	0.3	7.8
11	Terong	24	1.1	0.2	5.5
12	Lobak	19	0.9	0.1	4.2
13	Wortel	42	1.2	0.3	9.3
14	Tomat	20	1	0.3	4.2
15	Selada	15	1.2	0.2	2.9

Daftar komposisi bahan makanan yang berasal dari buah sebanyak 15 komposisi bahan makanan ditunjukkan pada Tabel 2.5.

Tabel 2.5 Komposisi Bahan Makanan Buah

No.	Sumber	Energi (kkal)	Protein (kkal)	Lemak (kkal)	Karbohidrat (kkal)
1	Anggur	50	0.5	0.2	12.8
2	Apel	58	0.3	0.4	14.9
3	Semangka	28	0.5	0.2	6.9
4	Rambutan	69	0.9	0.1	18.1

5	Belimbing	36	0.4	0.4	8.8
6	Jambu biji	49	0.9	0.3	12.2
7	Jambu air	46	0.6	0.2	11.8
8	Jeruk manis	45	0.9	0.2	11.2
9	Jeruk nipis	37	0.8	0.1	12.3
10	Sirsak	65	1	0.3	16.3
11	Nanas	52	0.4	0.2	13.7
12	Mangga gadung	44	0.7	0.2	11.2
13	Pepaya	46	0.5	0	12.2
14	Salak	77	0.4	0	20.9
15	Manggis	63	0.6	0.6	15.6

2.4 Gizi

Gizi dikenal dengan istilah nutrition atau nutrisi yang artinya zat makanan. Gizi merupakan komponen kebutuhan tubuh yang sangat penting. Untuk memperoleh zat gizi tentunya didapatkan dari makanan yang dikonsumsi sehari-hari guna memenuhi kebutuhan dalam tubuh. Untuk mencapai keperluan zat dalam tubuh, makanan yang dikonsumsi perlu diperhatikan dikarenakan makanan yang sesuai porsi kebutuhan dalam tubuh akan menghasilkan energi yang tepat (Irianto, 2007). Pola makanan yang dibutuhkan dengan mengonsumsi makanan 4 sehat 5 sempurna yaitu nasi, sayur, lemak, buah dan susu. Pengkonsumsian makanan yang sehat dapat membantu pertumbuhan dan kebutuhan energi untuk melakukan aktivitas serta kebutuhan gizi. Energi yang dihasilkan dari pengkonsumsian makanan sehari-hari adalah lemak, karbohidrat dan protein. Ketiga zat tersebut merupakan zat organik yang mengandung karbon yang dibutuhkan oleh tubuh. (Almatsier, 2009)

2.4.1 Energi

Tubuh memerlukan energi untuk dapat melakukan aktivitas sehari-hari. Satuan energi disebut kilokalori (Kkal) untuk menyatakan jumlah kalori. Makanan

yang dikonsumsi dapat menghasilkan sebuah energi untuk memenuhi kebutuhan sesuai ukuran tubuh dan komposisi serta untuk kebutuhan energi pada saat melakukan aktifitas. Basal Metabolic Rate (BMR) merupakan komponen utama dalam menghitung kebutuhan energi.

Perhitungan dengan menambahkan atau mengurangi dengan faktor-faktor yang menentukan kebutuhan kalori yang meliputi jenis kelamin, umur, aktivitas fisik atau pekerjaan dan berat badan (Pranoto dkk., 2011).

1. Jenis Kelamin

Kebutuhan kalori berdasarkan jenis kelamin disebut juga energy basal (*Basal Energy Expenditure* / BEE atau Angka Metabolisme Basal / AMB). Angka Metabolisme Basal adalah kebutuhan energi minimal untuk menjalankan aktivitas vital dalam tubuh baik pria maupun wanita. Perhitungan Metabolisme Basal dengan Persamaan 2.4 dan 2.5.

$$\text{AMB Pria} = \text{BBI} \times 30 \text{ Kkal/kg BB} \quad (2.4)$$

$$\text{AMB Wanita} = \text{BBI} \times 25 \text{ Kkal/kg BB} \quad (2.5)$$

Keterangan :

BBI = Berat Badan Ideal (kg)

AMB = Angka Metabolisme Basal (Kkal)

2. Umur

Perhitungan kalori berdasarkan umur dibagi berdasarkan dekade umurnya yang dihitung dengan Persamaan 2.6 sampai 2.8.

$$40 - 59 \text{ tahun (Kkal)} = - 5 \% \times \text{AMB} \quad (2.6)$$

$$60 - 69 \text{ tahun (Kkal)} = - 10 \% \times \text{AMB} \quad (2.7)$$

$$\geq 70 \text{ tahun (Kkal)} = - 20 \% \times \text{AMB} \quad (2.8)$$

3. Aktivitas Fisik atau Pekerjaan

Pengelompokan dalam aktivitas fisik dan pekerjaan dapat dilihat pada Tabel 2.6 dibawah ini.

Tabel 2.6 Kategori Aktivitas Fisik atau Pekerjaan

Kategori	Aktivitas
<i>Bed Rest</i>	Koma
Ringan	Pegawai kantor, pegawai toko, guru, ahli hukum, ibu rumah tangga
Sedang	Pegawai di industri ringan, mahasiswa, militer yang sedang tidak perang
Berat	Petani, buruh, militer dalam keadaan latihan, penari, atlet
Sangat Berat	Tukang becak, tukang gali, pandai besi

Sumber : (Rianawati & Mahmudy, 2015)

Jumlah kalori yang dibutuhkan perhari sesuai aktivitasnya dapat dihitung dengan persamaan 2.9 sampai 2.13.

$$TEE \text{ Bed rest} = 10 \% \times \text{AMB} \quad (2.9)$$

$$TEE \text{ Ringan} = 20 \% \times \text{AMB} \quad (2.10)$$

$$TEE \text{ Sedang} = 30 \% \times \text{AMB} \quad (2.11)$$

$$TEE \text{ Berat} = 40 \% \times \text{AMB} \quad (2.12)$$

$$TEE \text{ Sangat Berat} = 50 \% \times \text{AMB} \quad (2.13)$$

Keterangan :

TEE = *Total Energy Expenditure* (Kkal)

4. Berat Badan

Pengelompokan berat badan ditentukan menggunakan Indeks Massa Tubuh (IMT) dan dapat dilihat pada Tabel 2.7 dibawah ini.

Tabel 2.7 Kategori Berat Badan

Kategori	Indeks Massa Tubuh (IMT)
Kurus	< 18,5
Gemuk	18,5-29,9
Obesitas	> 30

(Sumber : Maryamah dkk., 2017)

Perhitungan kalori berdasarkan berat badan dihitung dengan persamaan 2.14 sampai 2.16.

$$\text{Kurus (Kkal)} = + 20 \% \times \text{AMB} \quad (2.14)$$

$$\text{Gemuk (Kkal)} = - 20 \% \times \text{AMB} \quad (2.15)$$

$$\text{Obesitas (Kkal)} = - 30 \% \times \text{AMB} \quad (2.16)$$

Menghitung kalori total perhari dengan persamaan 2.17.

$$\begin{aligned} \text{Total Kalori (Kkal)} = & \text{kalori jenis kelamin} \pm \text{kalori umur} \pm \text{kalori aktivitas} \\ & \pm \text{kalori berat badan} \end{aligned} \quad (2.17)$$

Setelah menghitung total kalori, maka untuk menghitung jumlah karbohidrat, protein dan lemak yang diperlukan dengan persamaan sebagai berikut.

$$\text{Protein (Kkal)} = 15\% \times \text{Total Kalori} \quad (2.18)$$

$$\text{Lemak (Kkal)} = 25\% \times \text{Total Kalori} \quad (2.19)$$

$$\text{Karbohidrat (Kkal)} = 60\% \times \text{Total Kalori} \quad (2.20)$$

2.4.2 Protein

Protein merupakan sebuah molekul kecil yang terdiri dari oksigen, hydrogen, karbon dan nitrogen. Setiap satu gram protein terdapat 4 Kkal. Fungsi dari protein antara lain menyeimbangkan antara asam dan basa, menjaga

metabolisme tubuh, enzim, pertumbuhan sel, kulit, rambut dan yang paling besar adalah sumber energi. Selain itu protein juga mempunyai peran penting untuk menstabiliskan pH cairan tubuh. Berdasarkan sumbernya protein dibagi menjadi dua yaitu protein nabati dan hewani. Protein nabati merupakan protein yang dapat ditemui pada kandungan bahan makanan yang berasal dari tumbuhan. Protein nabati mengandung asam amino yang lebih sedikit dibanding protein hewani. Selain itu nutrisi dan vitamin juga terkandung dalam protein nabati. Bahan makanan yang mengandung protein nabati seperti tahu, tempe, dan lain sebagainya. Sedangkan protein hewani merupakan protein yang banyak mengandung asam amino, selain itu juga mengandung lemak jahat. Sumber protein hewani dari bahan makanan dapat diperoleh dari telur, susu, semua jenis ikan, dan daging (Almatsier, 2009).

2.4.3 Lemak

Fungsi dari lemak antara lain memelihara suhu tubuh, mengganti sel-sel yang rusak, dan membangun sel-sel jaringan tubuh serta penghasil asam lemak esensial. Lemak menghasilkan 9 Kkal setiap gramnya selain itu energi yang dihasilkan tiga kali lebih besar dari pada energi yang terdapat dalam karbohidrat dan protein dengan jumlah yang sama. Cadangan energi untuk tubuh paling besar adalah lemak (Almatsier, 2006). Kandungan lemak dapat ditemukan dalam minyak yang berasal dari tumbuhan seperti kelapa sawit, minyak kelapa, kacang kedelai dan lain sebagainya. Selain itu susu, keju, daging, dan kuning telur juga mengandung lemak. Menurut World Health Organization (WHO), 1990) Kebutuhan lemak yang baik bagi kesehatan sebanyak 20-30% dari semua kebutuhan energi total. Mengonsumsi lemak dalam sehari sebesar 8% dari kebutuhan energi total yang berasal dari lemak jenuh, dan 3-7% dari lemak tak jenuh. Kolesterol yang baik bisa dikonsumsi sebanyak ≤ 300 mg/hari.

2.4.4 Karbohidrat

Karbohidrat merupakan sumber energi yang dibutuhkan pertama bagi tubuh manusia. Karbohidrat juga dapat menjadi glukosa didalam sirkulasi darah serta dapat diubah menjadi lemak yang bermanfaat bagi tubuh. Karbohidrat dapat

melancarkan sistem pencernaan, metabolisme lemak dan pemanis yang alami. Karbohidrat terdiri dari unsur senyawa karbon (C), Oksigen (O₂) dan Hidrogen (H). Karbohidrat sangat baik bagi tubuh sebagai sumber energi (Musnah, 2017). Karbohidrat terdapat 2 jenis yaitu karbohidrat kompleks dan sederhana. Dari 2 jenis karbohidrat tersebut nantinya didalam tubuh akan diolah menjadi gula darah yang menghasilkan energi dalam tubuh. Kebutuhan karbohidrat sebanyak 60% sangat dianjurkan untuk kebutuhan kalori dalam sehari (Soenardi, 2006).

2.5 Algoritma Genetika

Algoritma genetika adalah salah satu algoritma dengan teknik pencarian heuristik berdasarkan mekanisme evolusi biologis. Pada umumnya, algoritma genetika digunakan untuk pemecahan masalah yang kompleks (Gen dalam Uyun dan Hartati, 2011).

Algoritma genetika mempunyai tahapan yaitu inisialisasi kromosom, reproduksi berupa crossover dan mutasi, dan evaluasi. Inisialisasi kromosom berupa gen-gen yang merepresentasikan solusi. Proses penentuan gen dilakukan secara random (Rianawati & Mahmudy, 2015). Proses selanjutnya yaitu reproduksi yang melibatkan beberapa parent di dalam crossover dan mutasi. Pada tahap evaluasi akan memilih solusi mana yang akan digunakan pada generasi selanjutnya dengan menggunakan nilai fitness. Evaluasi memiliki banyak jenis seperti elitism, roulette wheel, dan lain sebagainya. Hasil dari algoritma genetika biasanya dalam suatu pemecahan masalah tidak selalu mendapatkan solusi yang terbaik namun mendekati terbaik (Musnah, 2017).

Menurut Michalewicz dan Mahmudy, 1996 proses pencarian algoritma genetika berbeda dengan pencarian dan optimasi pada umumnya (Musnah, 2017). Perbedaan tersebut antara lain adalah :

1. Adanya kode dari himpunan parameter dalam manipulasi atau disebut juga kromosom.
2. Proses pencarian dilakukan dari beberapa titik dalam satu populasi, tidak hanya dari satu titik.

3. Menggunakan informasi serta fungsi tujuan pada proses pencarian.
4. Stokastik operator digunakan untuk pencariannya yang bersifat probabilitas, tidak dengan menggunakan aturan deterministik.

2.5.1 Parameter Algoritma Genetika

Berikut adalah parameter yang digunakan dalam algoritma genetika diantaranya adalah :

1. Ukuran populasi (*popsize*)

Ukuran populasi menunjukkan jumlah kromosom yang terdapat populasi dalam 1 generasi. Jumlah populasi sangat berpengaruh pada waktu komputasi. Jumlah populasi yang sedikit berpengaruh pada solusi yang kurang optimum karena ruang solusi yang dihasilkan kurang mampu merepresentasikan permasalahan yang terkait. Sedangkan jika jumlah populasi banyak, maka waktu yang dibutuhkan untuk memproses data juga akan membutuhkan waktu yang lama, tapi dapat mencegah terjadinya konvergensi.

2. *Crossover Rate*

Crossover Rate atau disebut C_r adalah perbandingan antara jumlah kromosom dalam populasi dengan jumlah keturunan yang dihasilkan. Tukar silang terjadi saat sebanyak C_r dikalikan dengan *popsize* dalam populasi. Jika nilai C_r semakin besar, maka alternative solusi yang dihasilkan lebih bervariasi. Sebaliknya, nilai C_r yang kecil, dapat menghasilkan solusi yang kurang optimum dikarenakan waktu komputasi yang lama dan proses tukar silang yang banyak.

3. *Mutation Rate*

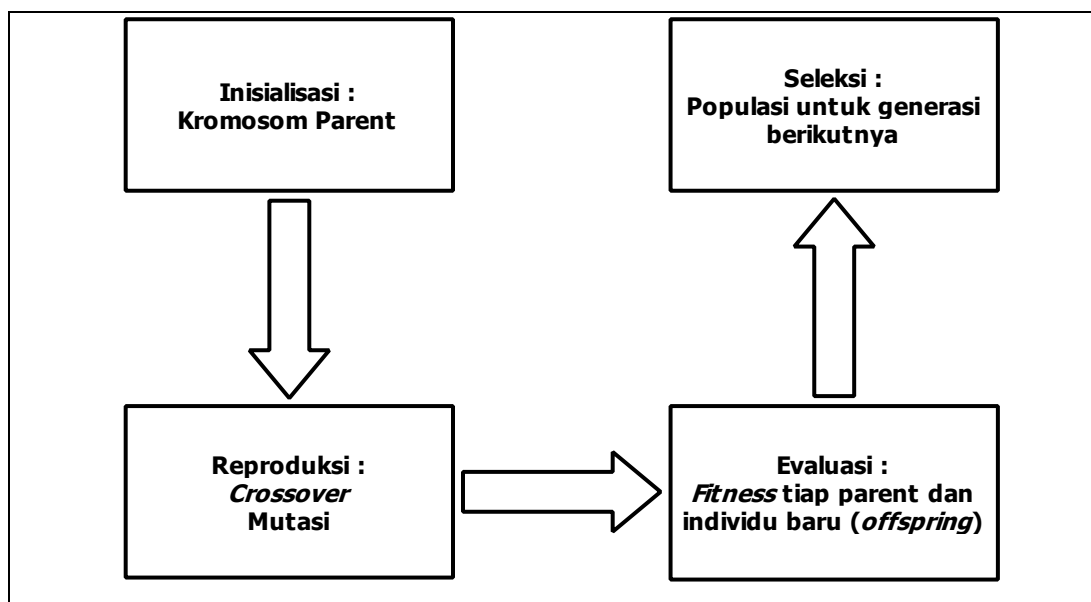
Mutation Rate atau disebut M_r merupakan parameter yang mempunyai fungsi sebagai penentu banyaknya kromosom dalam populasi yang akan dimutasi. Jika *Mutation Rate* semakin kecil, maka memungkinkan kromosom tidak melakukan proses mutasi.

4. Jumlah Generasi

Jumlah generasi ditentukan oleh banyaknya jumlah iterasi yang dilakukan pada proses evaluasi setiap generasi. Proses dalam menentukan kondisi *break* adalah dengan menentukan jumlah generasi yang tepat agar proses algoritma genetika tidak melakukan perulangan.

2.5.2 Struktur Algoritma Genetika

Penerapan algoritma genetika memiliki siklus yang terdiri dari beberapa tahapan seperti pada Gambar 2.1 dibawah ini :



Gambar 2.1 Struktur Algoritma Genetika

Sumber : (Mahmudy, 2013)

A. Inisialisasi

Inisialisasi bertujuan untuk menjelaskan nilai dari parameter dalam algoritma genetika yang terdiri dari sejumlah kromosom yang menjadi satu di dalam suatu himpunan atau disebut populasi. Populasi menunjukkan sekumpulan individu atau kromosom yang bergabung menjadi satu. Ukuran populasi (*popsi*) adalah banyaknya jumlah kromosom atau individu didalam sebuah populasi. dalam fase ini ditentukan kromosom yang akan menjadi *parent*. Ukuran populasi (*popsi*) dan harus di tentukan saat proses inisialisasi (Mamudy, 2013).

B. Representasi kromosom

Representasi kromosom merupakan proses dalam memetakan solusi dari sebuah permasalahan menjadi kromosom. Hasil algoritma genetika dalam memberikan solusi salah satunya dipengaruhi oleh representasi kromosom. Terdapat beberapa model dalam merepresentasikan sebuah kromosom diantaranya representasi integer, permutasi, real, dan biner. Dalam sebuah permasalahan memiliki perbedaan dalam merepresentasikan kromosom dikarenakan tidak semua bentuk representasi cocok untuk suatu permasalahan yang ada.

C. Reproduksi

Fase reproduksi ini mempunyai fungsi untuk membentuk individu atau kromosom baru. Terdapat dua fase reproduksi yaitu mutasi dan tukar silang. Berikut penjelasan lebih lengkapnya :

1. Mutasi

Mutasi merupakan cara untuk menghasilkan individu baru sebagai anak (*offspring*) sehingga, individu dalam populasi tambah beragam dengan mengubah gen dari keturunan melalui proses acak. Terdapat dua cara mutasi dalam representasi permutasi adalah *insertion mutation* dan *Reciprocal exchange*. Proses mutasi dari *insertion mutation* adalah dengan memilih gen secara random, selanjutnya gen yang terpilih tersebut diselipkan pada posisi yang dipilih secara acak. Sedangkan proses mutasi dari *reciprocal exchange* adalah dengan memilih dua posisi tempat secara random kemudian menukarkan nilai pada posisi keduanya. Dengan hal itu, maka yang akan menghasilkan individu baru. Proses mutasi dilakukan setelah proses *crossover*. Perannya adalah untuk mengganti gen yang hilang dari populasi akibat proses seleksi yang memungkinkan tidak muncul pada inisialisasi populasi (Musnah, 2017).

2. Tukar Silang (*crossover*)

Crossover merupakan proses memilih dua *parent* secara random dan terpisah menjadi 2 segmen, sehingga penukaran segmen kromosom *parent*

tersebut yang akan menghasilkan keturunan. Setelah terpilih proses selanjutnya melakukan penyilangan untuk menghasilkan anak (*offspring*). Pada umumnya salah satu metode tukar silang yang sering digunakan adalah *one-cut point crossover*. Metode ini merepresentasikan bilangan biner dan permutasi dalam menghasilkan individu baru. *Real Code Genetic Algorithms (RCGA)* mempunyai fungsi khusus untuk mengkonversi dari bilangan biner ke bilangan real atau sebaliknya, karena dengan kromosom dalam bentuk bilangan *real* akan membutuhkan waktu yang lama (Mahmudy, 2013). Dalam RCGA *crossover* yang sering dipakai yaitu *Intermediate Crossover* yang mengaplikasikan dengan *realcode*.

D. Evaluasi

Evaluasi merupakan proses untuk menghitung *fitness* setiap kromosom. Dalam menentukan kualitas dari suatu individu pada algoritma genetika nilai *fitness* yang menjadi sebuah acuan dalam pencapaian nilai paling optimum. Pada proses ini setiap populasi akan di evaluasi dengan menghitung nilai *fitness* dari setiap gen dan evaluasi tersebut akan terjadi perulangan sampai kriteria berhenti terpenuhi. Kriteria berhenti antara lain yaitu nilai *fitness* tidak berubah setelah dalam beberapa generasi secara berturut-turut, kondisi berhenti saat n generasi tidak mendapatkan nilai *fitness* terbesar, dan *break* pada generasi tertentu. Bila kriteria dalam situasi berhenti dan belum terpenuhi, maka akan terjadi proses pembentukan individu baru. Jika nilai *fitness* yang dihasilkan semakin besar maka semakin baik kromosom tersebut untuk menjadi solusi terbaik dalam suatu permasalahan.

E. Seleksi

Fase ini bertujuan untuk memilih individu dalam populasi dan individu baru (*offspring*) yang mampu bertahan hidup dan lolos dalam proses seleksi. Langkah pertama yang dilakukan adalah mencari nilai *fitness*. Nantinya, pada tahap-tahap selanjutnya nilai *fitness* ini akan digunakan sebagai acuan. Jika nilai *fitness* sebuah kromosom besar maka semakin besar pula peluang untuk terpilih, sehingga generasi berikutnya akan lebih baik dari generasi sebelumnya. Terdapat

beberapa metode seleksi yang sering digunakan antara lain seleksi *elitism*, *roulette wheel* dan *binary tournament*. Metode *elitism* memungkinkan individu yang bernilai *fitness* tertinggi akan terpilih. Akan tetapi, kemungkinan individu tersebut akan rusak (nilai *fitness*nya menurun) karena terjadinya pindah silang. Sedangkan Metode *roulette wheel* atau disebut juga *stochastic sampling with replacement*. Pada tahap ini, individu-individu dipetakan dalam suatu segmen secara teratur agar individu memiliki ukuran yang sama dengan ukuran *fitness*-nya. Teknik bilangan acak akan dibangkitkan dan individu yang memiliki segmen dalam kawasan bilangan random tersebut akan diseleksi. *Binary Tournament selection* memiliki implementasi yang sederhana dalam algoritma genetika. Dalam tahap ini, n individu dipilih secara acak sebanyak *popsiz*e. Individu yang terpilih dibandingkan nilai *fitness*-nya. Nilai *fitness* yang lebih tinggi akan lolos pada generasi berikutnya.

2.6 *Simulated Annealing*

Simulated annealing merupakan algoritma optimasi yang mempunyai sifat *generik*. Dengan berbasis mekanika statistik dan probabilitas, algoritma *simulated annealing* ini mempunyai kelebihan dalam pencarian pendekatan suatu permasalahan dengan solusi optimum global dan melalui proses *annealing* (pendinginan) yang terdiri dari kaca atau baja. Umumnya suatu masalah yang membutuhkan algoritma *simulated annealing* adalah optimasi kombinatorial, seperti halnya tidak memungkinkan solusi optimum untuk sebuah permasalahan dalam ruang pencarian solusi yang begitu kompleks (Mahmudy, 2014). Berikut adalah Tabel 2.8 pemetaan dari *physical annealing* ke *simulated annealing* :

Tabel 2.8 Pemetaan *Physical Annealing* ke *Simulated Annealing*

Fisika (terkodinamika)	Simulated Annealing
Keadaan sistem	Solusi yang pasti
Perubahan Keadaan	Solusi tetangga
Energi	Biaya

Temperatur	Parameter Kontrol
Keadaan Beku	Solusi <i>Heuristik</i>

Sumber : (Musannah, 2017)

Penurunan temperatur secara signifikan pada benda padat yang sebelumnya telah dipanaskan hingga mencapai titik beku, kemudian proses pendinginan. *Simulated annealing* menggunakan beberapa parameter yaitu sebagai berikut :

1. Temperatur Awal (T_0) : Memulai iterasi proses *Simulated annealing*. Temperatur awal akan terus berkurang hingga $\leq$ atau sama dengan nilai temperatur akhir.
2. Temperatur Akhir (T_n) : Penanda batas akhir iterasi.
3. Alpha (α) : Faktor reduksi temperatur sehingga temperatur dapat dikurangi secara bertahap (Cahyaningyas, Ratnawati, & Sutrisno, 2016)

Tahapan pertama pada *simulated annealing* adalah inisialisasi parameter dan individu awal. Teknik inisialisasi yang digunakan sama dengan teknik algoritma genetika untuk populasi awal. Tahapan selanjutnya adalah *neighborhood*.

Proses *Neighborhood* merupakan proses modifikasi individu awal menjadi individu baru. Cara modifikasi dengan menukarkan nilai pada salah satu gen ke gen lain pada kromosom yang sama. Kemudian akan melalui proses evaluasi terhadap kedua individu. Dari individu awal dan individu baru nantinya nilai *cost*-nya akan diperbandingkan. Jika selisih (ΔE) berada diantara nilai *cost* individu baru dengan individu awal ≤ 0 , maka individu baru akan terpilih menggantikan individu awal. Sebaliknya, jika selisih dari (ΔE) > 0 , maka bilangan random antara 0 hingga 1 akan dibangkitkan. Jika dari probabilitas Boltzman lebih dari bilangan tersebut, maka individu baru akan di terima (Orkcu, 2013). Berikut adalah rumus dari probabilitas Boltzman menggunakan bilangan acak.

$$\text{Exp}\left(\frac{\Delta E}{T}\right) \geq \delta \quad (2.21)$$

Keterangan : Δf = Selisih nilai *cost*

T = Temperatur Sekarang

δ = Bilangan Random antara 0-1

Jika nilai random mempunyai nilai lebih besar dibandingkan probabilitas Boltzman, maka akan terjadi penurunan temperatur. Berikut adalah persamaan penurunan temperatur.

$$T_{0+n} = \alpha \times T_s \quad (2.22)$$

Keterangan : T_{0+n} = Tempertur iterasi berikutnya

α = Alpha

T_s = Temperatur Sekarang

Temperatur akan berhenti jika temperatur akhir sudah mencapai akhir.

2.7 *Hybrid Algoritma Genetika dan Simulated Annealing*

Kombinasi metode antara algoritma genetika dan *simulated annealing* atau lebih dikenal dengan sebutan GAZA dari beberapa penelitian digunakan dalam permasalahan optimasi. Menurut Sofianti cara kerja yang paralel merupakan kelebihan dari algoritma genetika sendiri, dengan banyaknya keberagaman individu dalam populasi, memungkinkan algoritma genetika akan terjebak pada kondisi ekstrim lokal saat bekerja didalam ruang pencarian dan operator-operator genetika tidak dapat menghasilkan *offspring* lebih baik dari *parentnya* (Musnah, 2017). Dari kelemahan tersebut, pada metode *simulated annealing* yang mampu bertahan pada lokal optimum dengan mengendalikan penurunan temperatur suhu sehingga dapat menutupi kelemahan dari algoritma genetika tersebut. Sebaliknya dengan kelemahan dari metode *simulated annealing* yang hanya bisa menghasilkan satu solusi, sedangkan solusi yang diabaikan tersebut mungkin saja akan lebih baik dari solusi yang dipilih dapat ditutupi dengan kelebihan algoritma genetika. Metode algortima genetika dan *simulated annealing* keduanya memiliki kekurangan dan kelebihan masing-masing, oleh karena itu penggabungan kedua algoritma ini bertujuan untuk menutupi kekurangan satu sama lain dan mampu

memberikan solusi yang paling optimum pada permasalahan kebutuhan gizi bagi penderita diabetes melitus.

2.8 Penelitian Terkait

Metode Algoritma Genetika dan *Simulated Annealing* sudah banyak digunakan dalam pengoptimalan solusi, seperti penelitian Algoritma Genetika untuk optimasi komposisi makanan bagi penderita diabetes mellitus yang dilakukan oleh Rianawati & Mahmudy (2015). Pada proses algoritma genetika menggunakan representasi permutasi bilangan integer dengan panjang kromosom 15 dimana setiap angka pada gennya merepresentasikan nomor makanan, metode *crossover* yaitu *single-point crossover*, metode mutasi dengan *reciprocal exchange mutation* dan diseleksi dengan elitism selection. Dari pengujian yang dilakukan memperoleh hasil dengan parameter optimal yaitu ukuran populasi sebesar 160 individu dengan rata-rata fitness sebesar 0.0774665, 100 generasi dengan rata-rata fitness sebesar 0.0774665 dan kombinasi $cr = 0.4$ dan $mr = 0.6$ dengan rata-rata fitness sebesar 0.0780737. Hasil akhir berupa kombinasi bahan makanan untuk makan pagi, siang dan malam dengan kandungan gizi yang mencukupi kebutuhan pasien dan biaya minimal. Perbedaan pada penelitian yang akan dilakukan menggunakan dua metode yaitu algoritma genetika dan *simulated annealing* untuk melakukan optimasi.

Penelitian berikutnya menggunakan metode algoritma genetika untuk mengoptimasi asupan gizi ibu hamil oleh (Sari, Mahmudy, dan Dewi, 2014). Penelitian yang dilakukan ini menggunakan *crossover* satu titik (*Single Point Crossover*), sedangkan untuk mutasi dengan *Reciprocal Exchange Mutation*, serta seleksi *elitism*. Pada penelitian ini menggunakan jumlah data uji 150. Data seperti bahan makanan untuk memenuhi asupan gizi bagi penderita diabetes melitus direpresentasikan dengan panjang kromosom antara 5-15. Nilai *fitness* dihasilkan dengan menghitung selisih antara harga bahan setiap kromosom, harga maksimal (total 10 max harga bahan) dan penalti. Generasi yang mendekati optimum sesuai

kebutuhan gizi yang dibutuhkan bagi penderita diabetes melitus ditunjukkan pada generasi ke 1500 diperoleh rata-rata nilai *fitness* yang dihasilkan 142670.

Penelitian terkait selanjutnya menggunakan algoritma genetika dalam optimasi komposisi makanan untuk penderita kolesterol yang dilakukan oleh Wahid & Mahmudy (2014). Dalam proses algoritma genetika ini menggunakan representasi kromosom permutasi bilangan integer dan memiliki panjang kromosom 9, dimana angka tersebut merepresentasikan nomor urut makanan. Metode *crossover* yang digunakan adalah *single point crossover*, metode mutasi yang digunakan adalah *reciprocal exchange mutation* dan metode seleksi menggunakan *elitism selection*. Dari pengujian yang telah dilakukan menghasilkan solusi optimal yaitu ukuran populasi 100 dengan nilai rata-rata *fitness* 0,1862463, ukuran generasi 90 dengan nilai rata-rata *fitness* 0,1838946, dan kombinasi *crossover rate* dan *mutation rate* adalah $cr = 0,7$ dan $mr = 0,3$ dengan nilai rata-rata *fitness* 0,18575847. Hasil yang diperoleh pada penelitian ini adalah berupa bahan makanan untuk penderita kolesterol.

Pada Penelitian yang dilakukan oleh Suci, Mahmudy dan Putri (2015) tentang optimasi biaya untuk memenuhi gizi dan nutrisi pada lanjut usia dengan menggunakan Algoritma Genetika. Pada penelitian ini menggunakan representasi kromosom permutasi dengan bilangan integer dengan panjang 14 gen yang terdiri dari makan pagi, snack, makan siang, dan makan malam. Nilai *fitness* didapatkan dari perhitungan 10 bahan makanan dengan harga yang optimum dikurangi dengan total harga bahan makanan pada tiap kromosom dan penalti. Didapatkan hasil dengan generasi sebanyak 500, rata-rata nilai *fitness* adalah 110175.49, ukuran populasi sama dengan 120 dengan rata-rata *fitness* 1198113.462, $C_r = 0,3$ $M_r = 0,7$ dengan rata-rata *fitness* adalah 109795.06.

Penelitian berikutnya yang dilakukan oleh Maryamah, Putri dan Wicaksono (2017) tentang optimasi komposisi makanan pada penderita diabetes melitus dan komplikasinya menggunakan algoritma genetika. Mereka mengatakan dengan bantuan sistem proses komputasi akan berlangsung dengan cepat dan jika diterapkan suatu algoritma perhitungan akan menghasilkan solusi yang lebih

optimal, salah satu algoritma tersebut adalah algoritma genetika. Algoritma genetika merupakan metode heuristik yaitu suatu metode pencarian, dalam pelaksanaannya terdapat aturan-aturan untuk memperoleh solusi yang lebih baik daripada solusi sebelumnya. Dan hasil yang didapatkan dari penelitian yang dilakukan adalah individu dalam populasi yang optimal berjumlah 250 individu dengan jumlah generasi 145 dan kombinasi C_r dan M_r paling optimal adalah 0.7 dan 0.3 dengan fitness 0.01857.

Begitu pula pada penelitian yang dilakukan oleh Elisa, Cholissodin dan Fauzi (2017) menggunakan algoritma genetika untuk sistem rekomendasi bahan makanan bagi penderita penyakit jantung. Data yang mereka gunakan dalam melakukan penelitian adalah data bahan makanan penyusun diet yang terdiri dari 8 jenis bahan makanan yaitu karbohidrat, protein hewani, protein nabati, sayur, buah, susu, gula dan minyak. Dalam mengubah bahan makanan menjadi kromosom digunakan representasi kromosom real code. Metode *crossover* yang digunakan adalah *extended intermediate crossover*, untuk metode mutasi yang digunakan adalah random mutation dan metode seleksi digunakan *elitism selection*. Dari hasil pengujian didapatkan nilai parameter algoritma genetika yang optimal, yaitu jumlah populasi sebesar 280 dengan rata-rata nilai fitness 103,7, nilai C_r dan M_r adalah 0,5 dan 0,5 dengan rata-rata nilai fitness 103,3 dan untuk jumlah generasi sebesar 100 dengan rata-rata nilai fitness 111,2. Hasil keluaran dari sistem merupakan rekomendasi bahan makanan dengan 5 kali waktu makan per hari, yaitu makan pagi, snack, makan siang, snack dan makan malam dengan jumlah hari sesuai pilihan pengguna.

Tak hanya menggunakan algoritma genetika saja, pada penelitian yang dilakukan oleh Sari, Mahmudy dan Dewi (2017) mereka menggunakan dua metode *hybrid* algoritma genetika dan *simulated annealing* dalam mengoptimasi kebutuhan gizi pada komposisi bahan makanan untuk balita. Representasi kromosom yang digunakan ada dua segmen, segmen pertama menggunakan bilangan integer dan segmen kedua menggunakan bilangan real code. Untuk proses reproduksi menggunakan metode *extended intermediate crossover* dan

metode *random mutation*. Hasil pengujian menghasilkan nilai fitness sebesar 0.10106 dengan parameter ukuran populasi = 100, generasi = 50, kombinasi C_r dan M_r = 0.8 dan 0.3, nilai α = 0.8, nilai T_0 = 2 dan T_n = 0.2. Hasil dari penelitian ini berupa rekomendasi bahan makanan sesuai kebutuhan gizi yang mendekati kebutuhan gizi balita yang sebenarnya dengan mempertimbangkan berat bahan makanan dan harga yang minimal dalam satu hari.

Pada penelitian lain yang dilakukan oleh Hamidah (2016) dengan melakukan penggabungan dua metode algoritma genetika dan *simulated annealing* tentang optimasi susunan bahan makanan bagi keluarga bertujuan untuk menghasilkan rekomendasi bahan makanan sehari, kandungan gizi serta harga total secara keseluruhan. Hasil optimum yang diperoleh dari penelitian dengan penggabungan algoritma genetika dan *simulated annealing*, mendapatkan jumlah generasi sebanyak 100 dengan *popsiz*e sebesar 280, C_r sebesar 0,6, M_r sebesar 0,4, temperatur awal sebesar 1, temperatur akhir sebesar 0,2 dan α sebesar 0,9. Dengan menggunakan *hybrid* algoritma genetika dan *simulated annealing* dapat menghemat biaya konsumsi bahan makanan keluarga tersebut dengan rata-rata yang dihasilkan sebesar 42,13%.

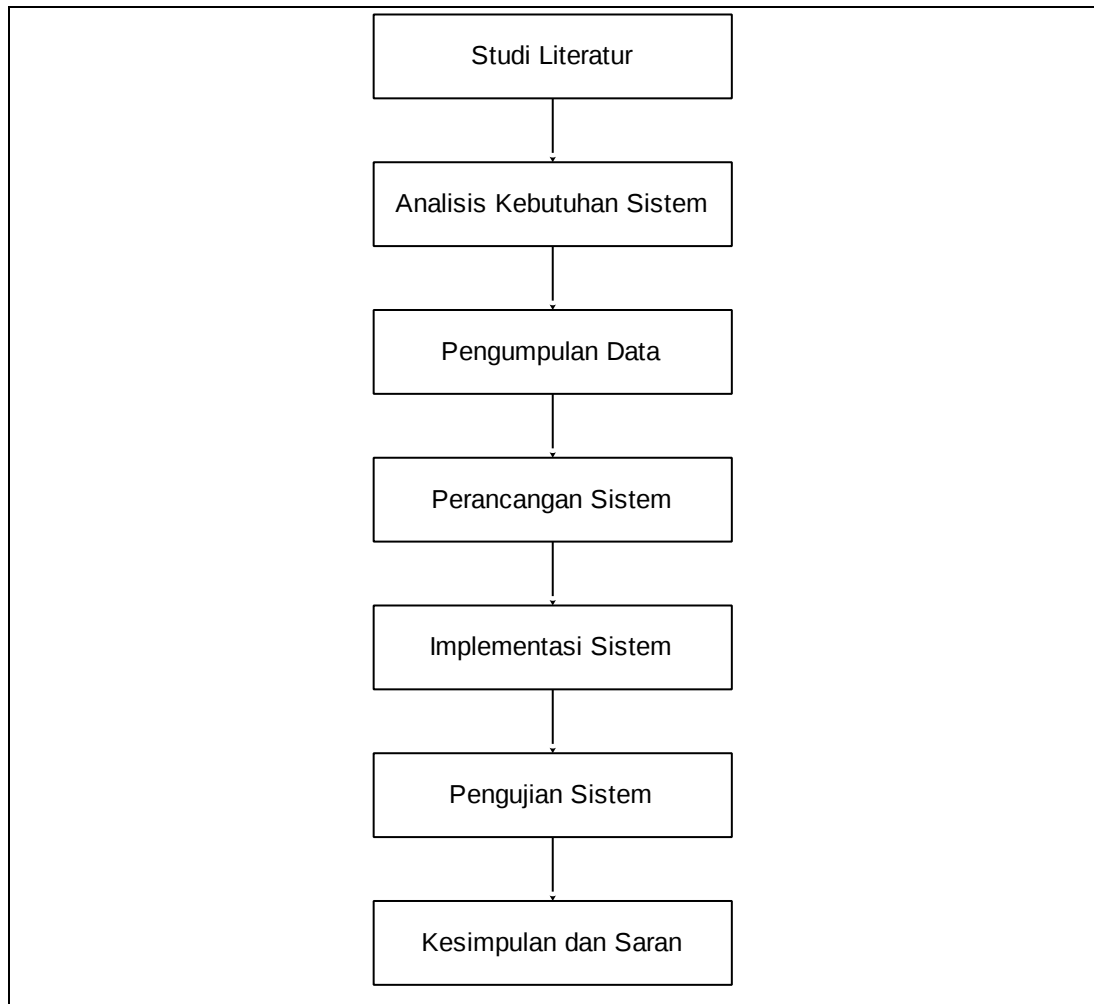
Penelitian lain juga melakukan kombinasi dengan dua metode yaitu algoritma genetika dan *simulated annealing*. Sofianti (2004) melakukan sebuah pengoptimalan sistem terdistribusi yang dilakukan dengan pengoptimalan transportasi perjalanan dalam tugas mengantarkan barang ke tangan konsumen tepat waktu untuk menyelesaikan sebuah permasalahan sistem terdistribusi pada suatu perusahaan yang bergerak dalam bidang ekspedisi maupun logistik. Ada salah satu yang merupakan permasalahan dalam transportasi yaitu *Vehicle Routing Problem with Stochastic Demands* (VRPSD). Pada penelitian tersebut nilai *fitness*, dipilih N solusi dengan *makespan* terkecil, kemudian dari N solusi tersebut hanya satu solusi saja yang dipilih yaitu yang paling optimum. Terdapat pengulangan sebanyak 6 kali pada kromosom yang mempunyai *makespan* terbaik. Jika *makespan* terbaik tidak berubah, maka terjadi penurunan temperatur. Kemudian dilakukan perbandingan antara kromosom terbaik dengan kromosom baru dari

hasil penurunan temperatur. Hasil dari penelitian yang menggunakan *hybrid* algoritma genetika dan *simulated* annealing yang diperoleh dapat dikatakan bahwa pengkombinasian kedua metode tersebut memiliki solusi yang paling optimum dibandingkan dengan hanya menggunakan algoritma genetika saja pada kasus yang sama.

BAB III

METODOLOGI PENELITIAN

Pada bab ini menjelaskan tentang tahapan penelitian yang akan dilakukan dan penyelesaian masalah terhadap pengujian keakuratan metode *hybrid* algoritma genetika dan *simulated annealing* dalam optimasi komposisi bahan makanan untuk penderita diabetes. Adapun tahapan metodologi yang dilakukan selama penelitian dapat dilihat pada Gambar 3.1, yang mana merupakan proses yang dimulai dari studi literatur hingga diperoleh kesimpulan.



Gambar 3.1 Alur Penelitian

Dalam melakukan suatu penelitian terdapat tahapan atau proses-proses yang harus dilakukan. Tahapan-tahapan dalam penelitian akan dilakukan secara rinci akan dijelaskan sebagai berikut :

3.1 Studi Literatur

Dalam proses penelitian, diperlukan pengumpulan pengetahuan dengan cara mempelajari literatur dari beberapa bidang ilmu yang berhubungan dengan optimasi komposisi bahan makanan bagi penderita diabetes melitus menggunakan metode *hybrid* algoritma genetika dan *simulated annealing*, yaitu diantaranya :

- Diabetes Melitus
- Komposisi bahan makanan bagi penderita diabetes melitus
- Algoritma Genetika
- *Simulated Annealing*
- *Hybrid* algoritma genetika dan *simulated annealing*

Literatur yang telah disebutkan diatas dapat diperoleh dari jurnal, e-book, dan penelitian sebelumnya. Hal ini dilakukan untuk dasar referensi terkait dengan tema penelitian yang dilakukan.

3.2 Analisis Kebutuhan Sistem

Dalam analisa sistem bertujuan mengidentifikasi sistem yang akan dirancang, yang meliputi perangkat lunak serta perangkat keras. Tahapan-tahapan yang menyusun analisa sistem ini adalah analisis data yang dipakai, spesifikasi kebutuhan sistem, spesifikasi pengguna, perancangan basis data, dan perancangan antarmuka.

3.2.1 Data

Data pengujian yang digunakan sebagai data penelitian adalah data dari penelitian terkait, dimana kasus tersebut dapat berisi tentang komposisi bahan makanan seperti kandungan gizi diantaranya karbohidrat, protein, lemak.

3.2.2 Spesifikasi Sistem

Analisa kebutuhan merupakan analisis terhadap komponen-komponen yang digunakan untuk pembuatan sistem. Dalam hal ini, komponen yang dibutuhkan terbagi menjadi dua macam, yaitu komponen perangkat lunak dan perangkat keras.

3.2.3 Spesifikasi Pengguna

Aplikasi ini digunakan oleh semua pihak terutama orang dewasa yang sedang memiliki keluhan terhadap kondisi kesehatan ataupun orang yang sedang menderita penyakit diabetes melitus.

3.2.4 Lingkungan Operasi

Perangkat lunak yang dibangun ini membutuhkan perangkat lunak lain sebagai penunjang agar sistem berjalan sesuai dengan fungsinya. Kebutuhan tersebut diantaranya:

1. Antarmuka pemakai

Sebagai penunjang antarmuka pemakai dari perangkat lunak, diperlukan aplikasi untuk membuat desain rancangan *interface* atau *mock up* dan lain-lain

2. Aplikasi server

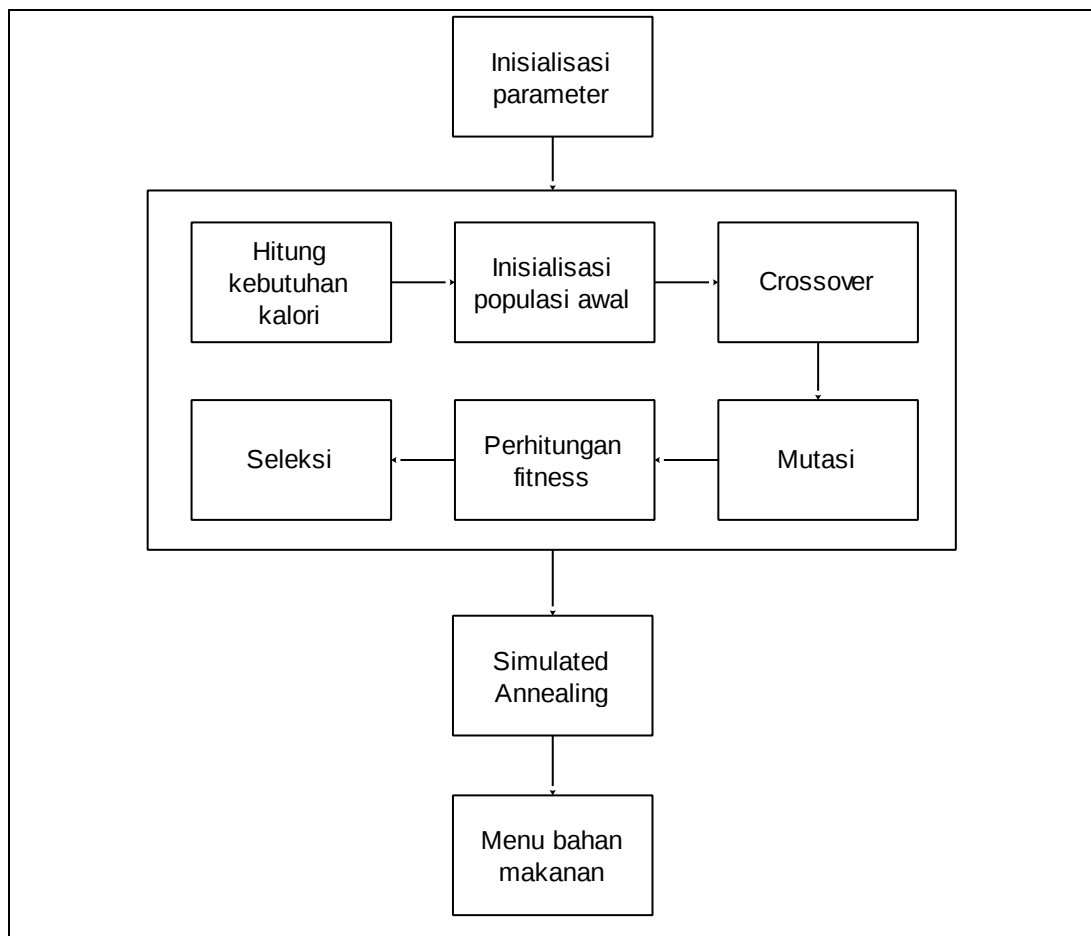
Untuk memusatkan proses dari perangkat lunak, menggunakan database server firebase untuk media penyimpanan data dari proses yang dilakukan oleh pengguna maupun data sebagai acuan sistem.

3.3 Pengumpulan Data

Tahap ini dilakukan untuk atau mengetahui informasi mengenai kebutuhan kandungan gizi diantaranya karbohidrat, protein, lemak bagi penderita diabetes melitus. Sedangkan data komposisi bahan makanan sejumlah 75 bahan. Untuk pengumpulan data komposisi bahan makanan diperoleh dari Daftar Komposisi Makanan (DKBM) Indonesia yang diterbitkan Depkes pada tahun 2005. Serta data pasien yang diambil dari klinik Al-Fattah Kota Pasuruan.

3.4 Perancangan Sistem

Perancangan sistem dilakukan untuk mempermudah implementasi, pengujian, serta analisis sistem yang dirancang. Sistem akan dibangun menggunakan bahasa pemrograman *Java* sebagai antarmuka yang dapat memudahkan pengguna untuk menggunakan sistem yang akan dibangun. Perancangan sistem direpresentasikan seperti Gambar 3.2



Gambar 3.2 Perancangan Sistem

Dalam perancangan sistem, secara garis besar langkah-langkah pembuatan sistem ini yaitu dengan dua metode ialah algoritma genetika dan *simulated annealing*. Kedua metode tersebut digunakan dalam optimasi komposisi bahan makanan penderita diabetes melitus. Metode algoritma genetika akan menyelesaikan pemilihan komposisi bahan makanan yang menghasilkan individu terbaik. Sedangkan metode *simulated annealing* akan melanjutkan hasil yang

didapatkan dari metode algoritma genetika yang akan meningkatkan keakuratan untuk menyelesaikan permasalahan optimasi komposisi bahan makanan bagi penderita diabetes melitus. Berikut adalah spesifikasi perancangan sistem yang akan dibangun:

1. Inisialisasi parameter

Menginisialisasi parameter data penderita diabetes mellitus yang meliputi jenis kelamin, umur, tinggi badan, berat badan dan aktifitas fisik. Serta untuk parameter fungsi optimasi meliputi *popsize*, *crossover rate* (C_r) *mutation rate* (M_r), jumlah generasi, temperatur awal (T_0), temperatur akhir (T_n) dan *alpha* (α)

2. Hitung kebutuhan kalori

Proses ini menghitung kalori yang di butuhkan untuk penderita diabetes melitus untuk mengetahui batasan kalori yang dibutuhkan diantaranya energy, protein, lemak dan karbohidrat.

3. Inisialisasi populasi awal

Menginialisasi populasi awal sejumlah *popsize* dan juga menghitung nilai *penalti* yang nantinya akan digunakan untuk menghitung nilai fitness pada proses Hitung *fitness*

4. *Crossover*

Proses crossover ini melakukan perhitungan terhadap nilai *offspring* yang digunakan untuk menentukan jumlah parent dan child yang dihasilkan. Dengan menggunakan metode *one cut point crossover* bertujuan untuk menyilangkan gen antar parent.

5. Mutasi

Proses mutasi ini juga dilakukan untuk menghitung nilai *offspring*. Dengan menggunakan metode *insertion mutation* yakni dengan menukar gen yang dipilih secara random kemudian menyisipkan gen ke posisi yang dipilih secara random.

6. Perhitungan *fitness*

Proses ini melakukan perhitungan nilai fitness dari populasi gabungan yaitu populasi awal dan *offspring* yang nantinya akan digunakan di proses selanjutnya.

7. Seleksi

Proses terakhir dari metode algoritma genetika ini melakukan penyeleksian terhadap populasi gabungan yang didapat dari proses sebelumnya dengan menggunakan metode *elitsm*, dan dapatkan individu terbaik.

8. *Simulated Annealing*

Proses ini untuk melanjutkan hasil yang didapat dari proses sebelumnya dengan menjadikannya populasi awal dari proses ini. Proses akan terus berlanjut hingga mencapai iterasi maksimum sesuai dengan jumlah generasi yang sudah ditentukan dan individu terbaik akan dihasilkan jika iterasi sudah mencapai maksimum.

9. Menu Bahan Makanan

Output yang dihasilkan dari optimasi berupa menu bahan makanan bagi penderita diabetes melitus untuk sehari diantaranya pagi, siang dan malam.

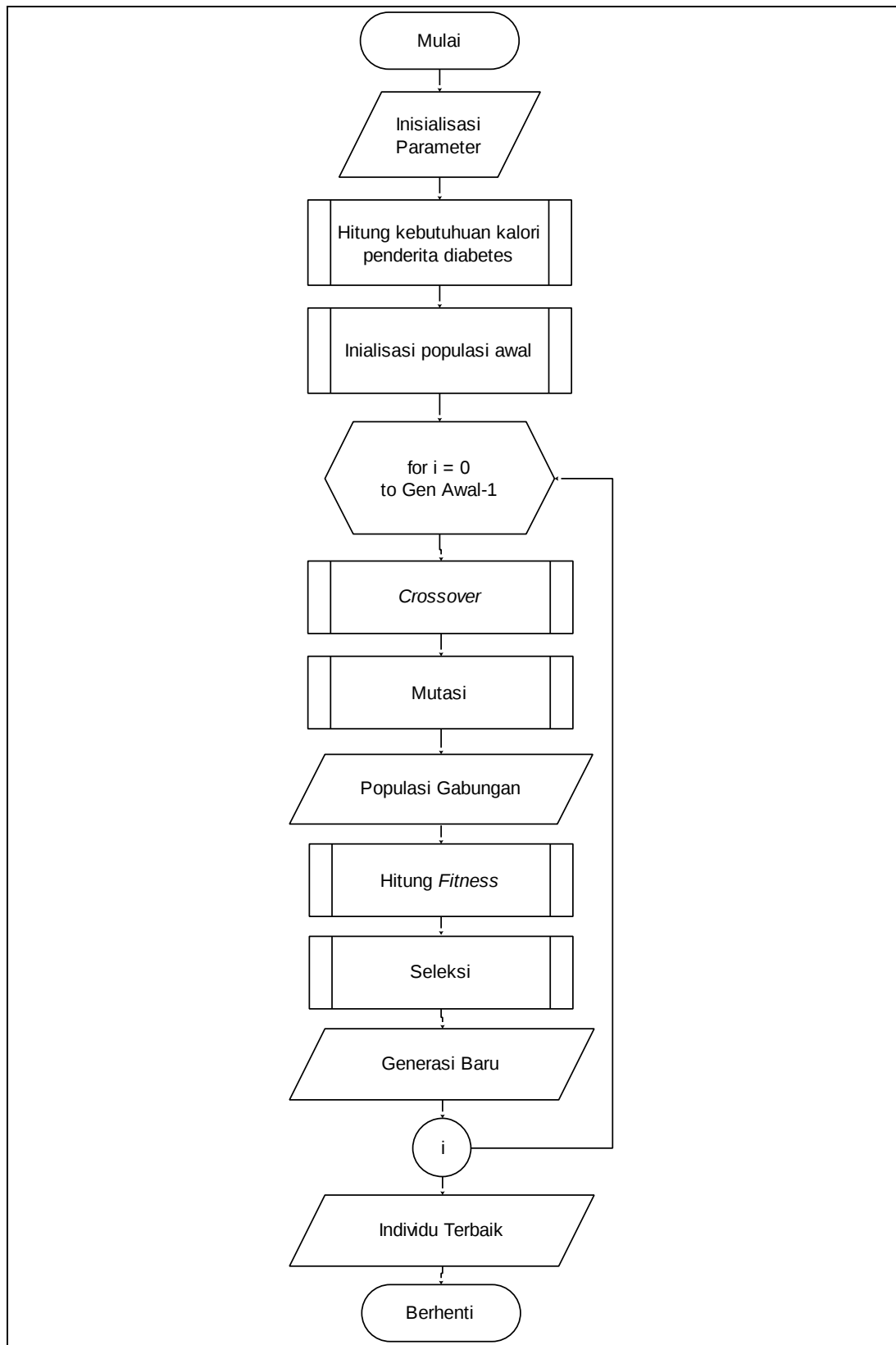
3.4.1 Siklus Algoritma Genetika

Siklus ini membahas tentang proses penyelesaian yang dilakukan dengan menggunakan algoritma genetika. Penjelasan proses optimasi menggunakan algoritma genetika diuraikan sebagai berikut :

- a) Menginisialisasi parameter antara lain data penderita diabetes melitus, *popsize*, probabilitas *crossover* (C_r), probabilitas mutasi (M_r), dan jumlah generasi.
- b) Menghitung kebutuhan kalori penderita diabetes melitus
- c) Inisialisasi populasi awal secara *random*
- d) Melakukan proses reproduksi *crossover*
- e) Melakukan proses mutasi dengan *insertion mutation*

- f) Menghitung nilai *fitness*
- g) Melakukan proses seleksi menggunakan metode *elitism*
- h) Populasi baru dengan menggabungkan individu induk dan *offspring* setelah memilih individu sebanyak populasi awal yang akan menjadi populasi baru
- i) Sesuai jumlah generasi, proses akan berlanjut hingga mencapai iterasi maksimum
- j) Jika kondisinya sudah terpenuhi maka akan dihasilkan individu terbaik yaitu dengan nilai *fitness* tertinggi dari populasi saat itu.

Langkah-langkah pada algoritma genetika akan ditunjukkan pada gambar 3.3 dibawah ini.

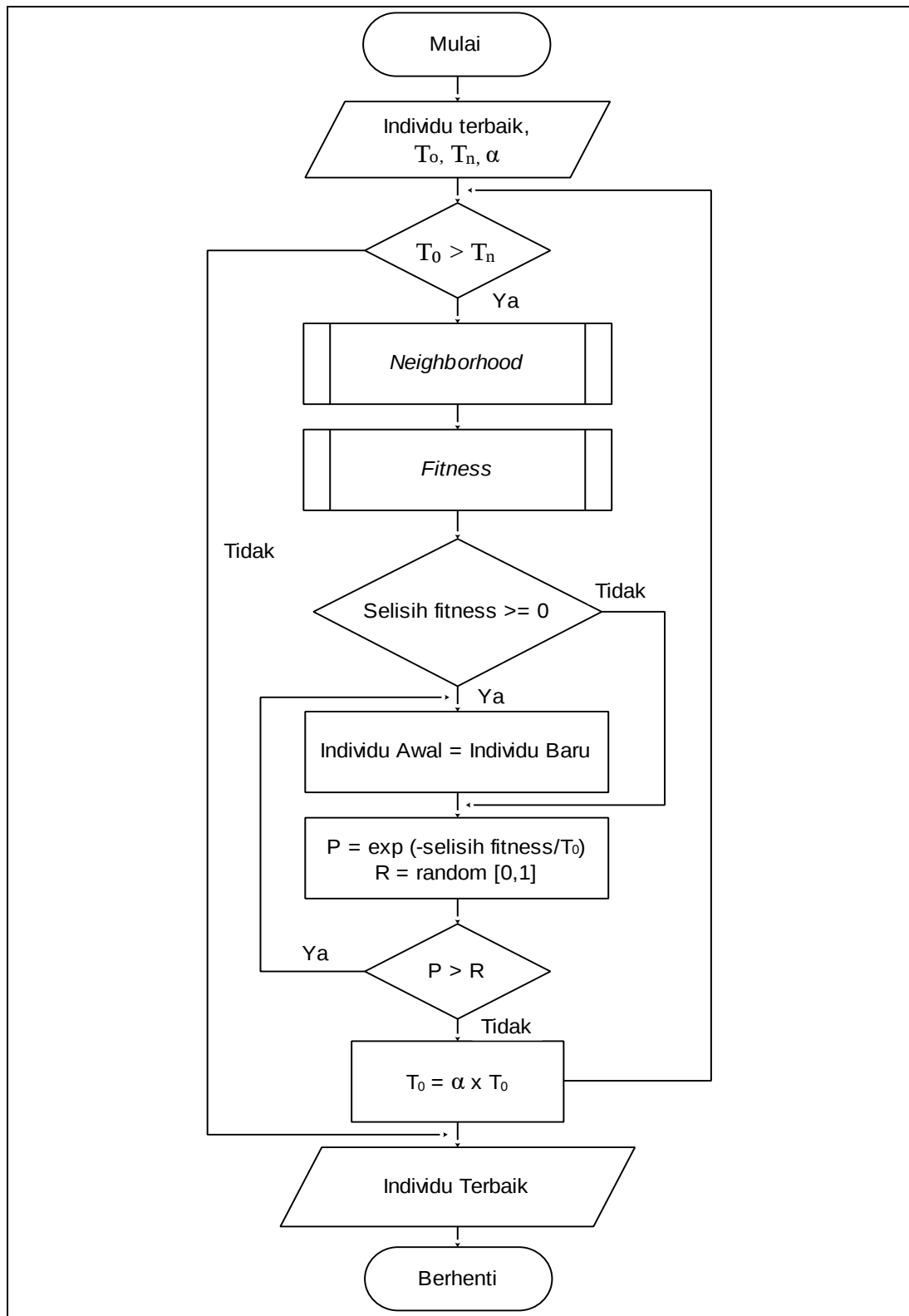


Gambar 3.3 Flowchart Proses Algoritma Genetika

3.4.2 Siklus Algoritma *Simulated Annealing*

Pada algoritma optimasi terdapat berbagai modifikasi, salah satunya algoritma *simulated annealing* terdapat proses yaitu *neighborhood*. Proses *neighborhood* berfungsi untuk memodifikasi suatu individu yang melalui proses *simulated annealing*. Adapun langkah-langkah dari proses *simulated annealing* ditunjukkan pada gambar 3.4 seperti berikut :

1. Memasukkan individu terbaik yang merupakan hasil yang didapatkan dari algoritma genetika dan beberapa parameter dari *simulated annealing* yakni seperti temperatur awal (T_0), temperatur akhir (T_n) serta alpha (α)
2. Melakukan perulangan dengan ketentuan untuk temperatur awal (T_0) merupakan batas awal sedangkan temperatur akhir (T_n) merupakan batas akhir
3. Melakukan Proses *neighborhood*
4. Melakukan perhitungan *fitness* pada individu awal dan pada individu baru
5. Jika selisih nilai *fitness* (Δf) ≥ 0 , maka individu baru akan menggantikan individu awal sebagai solusi baru
6. Jika selisih nilai *fitness* (Δf) < 0 , maka akan dibangkitkan bilangan antara 0 sampai 1 secara random dan melakukan perhitungan probabilitas boltzman
7. Jika persamaan pada (2.11) terpenuhi, maka individu baru akan diterima sebagai individu terbaik
8. Jika tidak, hal yang akan terjadi adalah diturunkan temperaturnya (2.11) dan kembali pada tahap ke 2
9. Jika kondisi sudah terpenuhi maka akan dihasilkan individu terbaik yaitu nilai *fitness* tertinggi dari populasi saat itu.

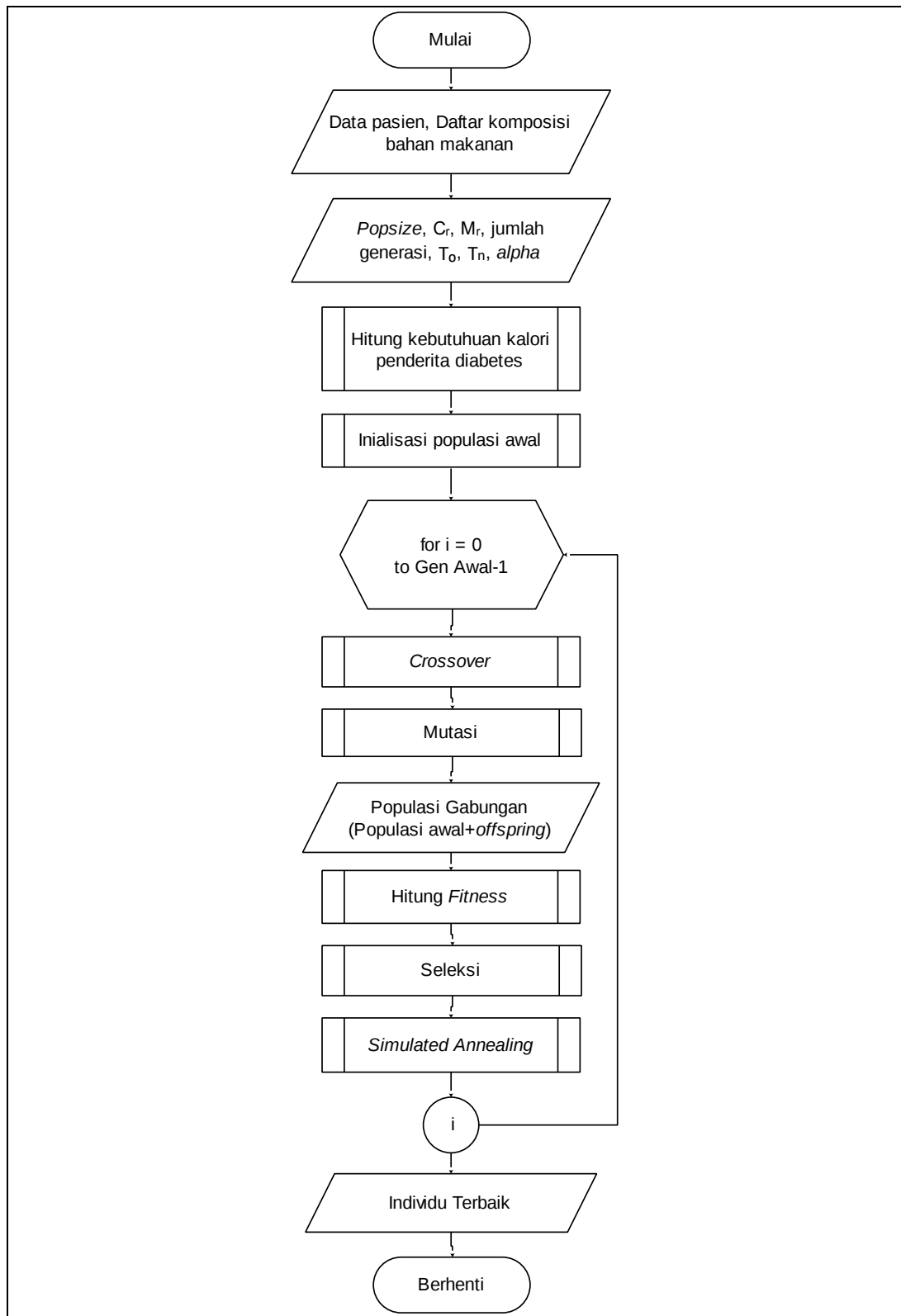


Gambar 3.4 Flowchart Proses *Simulated Annealing*

3.4.3 Siklus *Hybrid Algoritma Genetika* dan *Simulated Annealing*

Penjelasan tentang proses *hybrid* algoritma genetika dan *simulated annealing* dalam permasalahan optimasi kebutuhan gizi bagi penderita diabetes melitus terdapat pada langkah-langkah proses perhitungan yang tunjukkan pada gambar 3.5 berikut :

1. Melakukan inisialisasi parameter terhadap data pasien yang berupa jenis kelamin, tinggi badan, berat badan, usia, dan aktivitas fisik. Serta untuk parameter fungsi optimasi berupa *popsiz*, *crossover rate* (C_r) *mutation rate* (M_r), jumlah generasi, temperatur awal (T_0), temperatur akhir (T_n) dan *alpha* (α)
2. Melakukan inisialisasi populasi awal sejumlah *popsiz*
3. Melakukan proses *crossover* dengan menggunakan metode *one cut point crossover*
4. Melakukan proses mutasi dengan menggunakan metode *insertion mutation*
5. Menghitung nilai *fitness* dari populasi gabungan yaitu populasi awal dan *offspring*
6. Melakukan seleksi pada populasi gabungan dengan menggunakan metode *elitsm*
7. Menghasilkan individu terbaik dan selanjutnya akan diteruskan pada proses *simulated annealing*
8. Proses terus berlanjut sampai mencapai iterasi yang maksimum sesuai jumlah generasi yang sudah ditentukan
9. Menghasilkan individu terbaik jika iterasi yang dilakukan sudah mencapai maksimum dari hasil optimasi *hybrid* algoritma genetika dan *simulated annealing*, yang berupa menu bahan makanan bagi penderita diabetes dalm sehari diantaranya pagi, siang dan malam.



Gambar 3.5 Diagram Alur *Hybrid GA-ZA*

3.5 Contoh Persoalan

Berikut adalah contoh persoalan data penderita diabetes melitus yang diambil dari klinik Al-Fattah Kota Pasuruan seperti Tabel 3.1.

Tabel 3.1 Data Penderita Diabetes Melitus

Jenis Kelamin	Umur	Tinggi Badan	Berat Badan	Aktivitas
Laki-laki	55	170	80	Sedang

Pada persamaan Berat Badan Ideal (BBI), maka dapat dihitung berat badan ideal dengan persamaan 2.1.

$$\text{BBI} = 0,9 \times (170-100) \times 1\text{kg} = 63 \text{ kg}$$

Dalam menentukan kategori berat badan berdasarkan Indeks Massa Tubuh (IMT), maka digunakan persamaan 2.3.

$$\text{IMT} = 80 / (1.7)^2 = 27,6816609$$

Kategori : Gemuk (24,9 - 29,9)

Pada tahap selanjutnya adalah perhitungan dengan menambahkan atau mengurangi dengan faktor-faktor yang menentukan kebutuhan kalori yang meliputi jenis kelamin, umur, aktivitas fisik atau pekerjaan dan berat badan dengan persamaan 2.6, 2.11, 2.15 dan 2.17.

Jenis Kelamin

$$\text{AMB Pria} = 63 \times 30 \text{ Kkal/kg BB} = 1890 \text{ Kkal}$$

Umur

$$55 \text{ tahun} = -5\% \times 1890 = -94,5 \text{ Kkal}$$

Aktivitas

$$\text{TEE Sedang} = 30\% \times 1890 = 567 \text{ Kkal}$$

Keterangan :

TEE : *Total Energy Expenditure*

Berat Badan

$$\text{Gemuk} = -20\% \times 1890 = -378 \text{ Kkal}$$

Total Kalori

$$\text{Total} = 1890 - 94,5 + 567 - 378 = 1984,5 \text{ Kkal}$$

Dari total kalori yang didapatkan, langkah selanjutnya adalah menghitung total kebutuhan protein, lemak dan karbohidrat. Maka dengan menggunakan persamaan 2.18 sampai 2.20 sebagai berikut.

$$\text{Protein} = 15\% \times 1984,5 = 297,675 \text{ Kkal}$$

$$\text{Lemak} = 25\% \times 1984,5 = 496,125 \text{ Kkal}$$

$$\text{Karbohidrat} = 60\% \times 1984,5 = 1190,7 \text{ Kkal}$$

Berdasarkan data diatas dapat dilakukan penyusunan komposisi bahan makanan dalam satu hari menggunakan metode *hybrid* algoritma genetika dan *simulated annealing* bagi penderita diabetes melitus.

Inisialisasi Parameter Algoritma Genetika dan *Simulated Annealing*

Adapun parameter yang dibutuhkan dalam optimasi menggunakan *hybrid* algoritma genetika dan *simulated annealing* antara lain adalah *popsiz*e, *crossover rate* (C_r), *mutation rate* (M_r), jumlah generasi, temperatur awal, temperatur akhir dan *alpha*. Berikut ini adalah nilai parameter pada perhitungan manual :

1. jumlah generasi : 1
2. *popsiz*e : 3
3. *crossover rate* (C_r) : 0,6
4. *mutation rate* (M_r) : 0,4
5. temperatur awal (T_0) : 5

6. temperatur akhir (T_n) : 3

7. α (α) : 0,4

Representasi Kromosom dan Perhitungan Fitness

Terdapat 75 komposisi bahan makanan yang dibagi menjadi lima jenis yakni sumber pokok, sumber protein hewani, sumber protein nabati, sayuran dan buah seperti yang telah ditampilkan pada sub bab 2.3. Representasi permutasi menggunakan bilangan integer yang berisi nomor makanan yang akan dikonsumsi. Satu kromosom terdapat 15 gen yang menyusun komposisi bahan makanan selama sehari 3 kali yaitu pagi, siang, dan malam dengan 5 jenis komposisi bahan makanan. Pada tabel 3.2 menunjukkan representasi kromosom pada penelitian ini.

Tabel 3.2 Representasi Kromosom

Kromosom														
Pagi					Siang					Malam				
SP	SN	SH	S	B	SP	SN	SH	S	B	SP	SN	SH	S	B
2	7	5	12	9	1	8	15	11	6	3	14	4	10	13

Keterangan :

SP : Sumber pokok

SN : Sumber nabati

SH : Sumber hewani

S : Sayuran

B : Buah

Tabel 3.3 adalah contoh sebuah individu P1 dengan kromosom awal secara acak.

Tabel 3.3 Kromosom Awal

P1	2	7	5	12	9	1	8	15	11	6	3	14	4	10	13
----	---	---	---	----	---	---	---	----	----	---	---	----	---	----	----

Tabel 3.4 adalah detail dari kromosom awal pada individu P1 dan total kebutuhan gizi yang dibutuhkan.

Tabel 3.4 Detail kromosom Awal Pada Individu P1

Waktu Makan	Jenis Bahan Makanan	Gen	Nama Bahan	Energi	Protein	Lemak	Karbohidrat
Pagi	SP	2	Beras jagung	345	9,1	2	76,5
	SN	7	Kacang kapri	98	6,7	0,4	17,7
	SH	5	Ikan mujair	89	18.7	1	0
	S	12	Lobak	19	0.9	0.1	4.2
	B	9	Salak	77	0.4	0	20.9
Siang	SP	1	Singkong	146	1.2	0.3	34.7
	SN	8	Melinjo	66	5	0,7	13,3
	SH	15	Kepiting	151	13.8	3.8	14.1
	S	11	Terong	24	1.1	0.2	5.5
	B	6	Jambu biji	49	0,9	0,3	12,2
Malam	SP	3	Beras ketan hitam	356	7	0,7	78
	SN	14	Kluwak	273	10	24	13,5

	SH	4	Ikan mas	86	42	1.5	0
	S	10	Kacang panjang	44	2.7	0.3	7.8
	B	13	Pepaya	46	0.5	0	12.2
Total				1869	120	35,3	310,6

Setelah melakukan perhitungan nilai total kebutuhan sehari dari energi, protein, lemak dan karbohidrat dengan berat masing-masing jenis bahan makanan adalah 100 gram. Maka selanjutnya adalah melakukan perhitungan masing-masing nilai penalti setiap kebutuhan gizi yang nantinya dilakukan perbandingan pada perhitungan manual kebutuhan gizi penderita diabetes dengan persamaan 4.2 sampai 4.6 seperti berikut ini :

$$Penalti = penalti1 + penalti2 + penalti3 + penalti4$$

$$Penalti_{Energi} = \begin{cases} 0, TotalEnergi < KebEnergi \\ TotalEnergi - KebEnergi, TotalEnergi \geq KebEnergi \end{cases} \quad (3.1)$$

$$Penalti_{Protein} = \begin{cases} 0, TotalProtein < KebProtein \\ TotalProtein - KebProtein, TotalProtein \geq KebProtein \end{cases} \quad (3.2)$$

$$Penalti_{Lemak} = \begin{cases} 0, TotalLemak < KebLemak \\ TotalLemak - KebLemak, TotalLemak \geq KebLemak \end{cases} \quad (3.3)$$

$$Penalti_{Karbo} = \begin{cases} 0, TotalKarbo < KebKarbo \\ TotalKarbo - KebKarbo, TotalKarbo \geq KebKarbo \end{cases}$$

(3.4)

Sehingga nilai penalti yang diperoleh setiap energi, lemak, protein, dan karbohidrat sebagai berikut.

$$Penalti\ Energi = |(1869 - 1984,5)| = 115,5$$

$$Penalti\ Protein = |(120 - 297,675)| = 177,675$$

$$\text{Penalti Lemak} = |(35,3 - 496,125)| = 460,825$$

$$\text{Penalti Karbohidrat} = |(310,6 - 1190,7)| = 880,1$$

Dengan perhitungan diatas, maka penalti dari masing-masing kebutuhan gizi diperoleh nilai pinalti yaitu sebesar = 1634.1. Selanjutnya adalah rumus menghitung nilai *fitness* didapatkan dari persamaan 3.5 sebagai berikut.

$$Fitness = C \div (Penalti) \quad (3.5)$$

Keterangan :

C = Konstanta yang ditentukan (1000)

Penalti = Nilai hasil pembobotan

Menggunakan persamaan 3.5, maka hasil dari *fitness* sebagai berikut.

$$Fitness = 1000 \div (1634.1) = 0,611957653$$

Inisialisasi Populasi Awal

Berikut sebuah contoh populasi awal secara acak menggunakan integer dari 1-15 dan jumlah *popsiz*e sebesar 3 yang sudah ditentukan di awal. Tabel 3.5 menunjukkan populasi awal.

Tabel 3.5 Populasi Awal

Individu	SP	S N	S H	S	B	S P	S N	S H	S	B	S P	S N	S H	S	B	<i>fitness</i>
P1	2	7	5	1 2	9	1	8	15	1 1	6	3	14	4	1 0	1 3	0.61195765 3
P2	13	10	4	1 4	3	6	11	15	8	1	9	12	5	7	2	0.53616428 1
P3	15	14	13	1 2	1 1	1 0	9	8	7	6	5	4	3	2	1	0.59780009 6

Individu p1, p2, dan p3 adalah populasi awal. Panjang kromosom adalah sebanyak 15 gen. Setiap gen menyusun komposisi bahan makanan yang terdiri dari lima jenis yaitu sumber pokok, nabati, hewani, sayuran serta buah. Setiap lima jenis bahan makanan tersebut mengandung energi, protein, lemak, dan karbohidrat dengan berat sebesar 100 gram, maka setiap individu dihitung total

dari kebutuhan gizi tersebut selama sehari. Tabel 3.6 menunjukkan detail populasi awal beserta hasil nilai *fitness* dengan menggunakan perhitungan dari persamaan 3.5 untuk masing-masing individu.

Tabel 3.6 Detail Populasi Awal

Individu	Energi	Protein	Lemak	Karbohidrat	Penalti	<i>Fitness</i>
P1	1869	120	35.3	310,6	1634.1	0.611957653
P2	1708	119	20.8	256.1	1865,1	0.536164281
P3	2174	122.3	42.3	336.6	1672.8	0.597800096

Reproduksi

Pada tahapan ini terdapat dua proses yakni *crossover* dengan menggunakan menggunakan *one-cut point* dan mutasi menggunakan *insertion mutation*.

Crossover

Proses *crossover* dilakukan untuk menentukan berapa jumlah *offspring*. Nilai *offspring* didapatkan dari perkalian antara *popsiz*e dengan *crossover rate* (C_r). Nilai C_r yang digunakan 0,6 maka jumlah child yang digunakan adalah

$$\text{Offspring} = 3 \times 0,6 = 1,80$$

Hasil dari perkalian tadi dibulatkan menjadi 2, maka proses *crossover* harus memperoleh 2 child. Setiap induk menghasilkan satu keturunan. Sehingga jika harus menghasilkan 2 child, maka membutuhkan 2 parent yang akan diproses *crossover*.

Penyilangan dilakukan dengan memilih titik potong gen secara random. Proses *crossover* ditunjukkan pada Tabel 3.7 dibawah ini.

Tabel 3.7 Proses *Crossover*

P1	2	7	5	12	9	1	8	15	11	6	3	14	4	10	13
P2	13	10	4	14	3	6	11	15	8	1	9	12	5	7	2
C1	2	7	5	12	9	6	11	15	8	1	9	12	5	7	2

C2	13	10	4	14	3	1	8	15	11	6	3	14	4	10	13
----	----	----	---	----	---	---	---	----	----	---	---	----	---	----	----

C1 adalah *offspring* yang dihasilkan proses *crossover* antara induk p1 gen 1-5 dan p2 gen 6-15. Sedangkan C2 adalah *offspring* yang dihasilkan proses *crossover* antara induk p2 gen 1-5 dan p2 gen 6-15.

Mutasi

Proses mutasi menghasilkan *offspring* sama seperti proses *crossover*. *Offspring* dihasilkan dengan melakukan perkalian antara *popsiz* sebanyak 3 dengan M_r sebesar 0,4. Sehingga menghasilkan nilai *offspring* sebesar 1,2 yang juga dibulatkan keatas menjadi 2. Dikarenakan hasilnya 2, maka persilangan yang dilakukan harus menghasilkan 2 *offspring*.

$$\text{Offspring} = 3 \times 0,4 = 1,2$$

Mutasi yang digunakan ialah *insertion mutation*. Segmen akan dipilih satu gen secara random. Gen tersebut disisipkan diposisi yang juga dipilih secara random proses mutasi ditunjukkan oleh Tabel 3.8 dan 3.9.

Tabel 3.8 Proses Mutasi P2

	x					p									
P2	13	10	4	14	3	6	11	15	8	1	9	12	5	7	2
C3	13	10	6	4	14	3	11	15	8	1	9	12	5	7	2

x p

Tabel 3.9 Proses Mutasi P3

P3	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
C4	15	14	13	12	11	10	9	8	7	3	6	5	4	2	1

Child C3 adalah hasil mutasi dari induk P2, sedangkan child C4 adalah hasil dari mutasi P3. P adalah gen yang terpilih, sedangkan x merupakan tempat yang terpilih menyisipkan p. Pada induk P2, p terdapat pada gen ke-6, sedangkan pada induk P3, p terletak pada gen-13. Serta x pada P2 berada pada gen ke-3 sedangkan x pada P3 berada pada gen ke-10. Dengan hal itu, maka gen yang berada diantara x dan p akan bergeser satu tempat. Dikarenakan proses reproduksi sudah dilakukan dengan menghasilkan 4 *offspring* dan jumlah *popsi* 3 maka didapatkan hasil populasi gabungan antara induk dan keturunan seperti yang ditunjukkan pada Tabel 3.10.

Tabel 3.10 Populasi Gabungan

Individu	SP	SN	SH	S	B	SP	SN	SH	S	B	SP	SN	SH	S	B	<i>Fitness</i>
P1	2	7	5	12	9	1	8	15	11	6	3	14	4	10	13	0.611957653
P2	13	10	4	14	3	6	11	15	8	1	9	12	5	7	2	0.536164281
P3	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0.597800096
C1	2	7	5	12	9	6	11	15	8	1	9	12	5	7	2	0.550055006
C2	13	10	4	14	3	1	8	15	11	6	3	14	4	10	13	0.578268664
C3	13	10	6	4	14	3	11	15	8	1	9	12	5	7	2	0.635364381
C4	15	14	13	12	11	10	9	8	7	3	6	5	4	2	1	0.621310966

Seleksi

Pada proses ini menggunakan *elitism selection* sehingga diambil 3 individu dalam populasi dengan nilai *fitness* yang paling tinggi untuk masuk ke generasi selanjutnya. Urutan-urutan nilai *fitness* tertinggi adalah *offspring* C3, *offspring* C4 dan individu P1. Berikut Tabel 3.11 dibawah menunjukkan hasil seleksi Individu dengan *elitism*.

Tabel 3.11 Hasil Seleksi

Individu	SP	SN	SH	S	B	SP	SN	SH	S	B	SP	SN	SH	S	B	<i>Fitness</i>
C3	13	10	6	4	14	3	11	15	8	1	9	12	5	7	2	0.635364381
C4	15	14	13	12	11	10	9	8	7	3	6	5	4	2	1	0.621310966
P1	2	7	5	12	9	1	8	15	11	6	3	14	4	10	13	0.611957653

Proses *Simulated Annealing*

Metode *simulated annealing* bertujuan untuk memperkecil error dari metode algoritma genetika. Proses *simulated annealing* dilakukan setelah proses algoritma genetika selesai. Pada *simulated annealing* nilai *fitness* terbaik yang didapatkan dari hasil proses algoritma genetika akan digunakan sebagai individu awal. Parameter yang dibutuhkan *simulated annealing* adalah temperatur awal (T_0) sebesar 5, temperatur akhir (T_n) sebesar 3, dan *alpha* (α) sebesar 0,4. Pada proses algoritma genetika diketahui bahwa individu terbaik dengan nilai *fitness* tertinggi adalah C3 dengan nilai *fitness* yang dihasilkan sebesar 0.635364381 dan akan dijadikan sebagai individu awal pada proses *simulated annealing*.

Proses selanjutnya adalah melakukan proses *neighborhood* dengan memilih dua gen secara random. Gambar 3.6 menunjukkan proses *neighborhood* dimana x_n adalah individu C3 dan x_p adalah individu baru hasil modifikasi.

	X								P							
X_n	13	10	6	4	14	3	11	15	8	1	9	12	5	7	2	
X_p	13	10	8	4	14	3	11	15	6	1	9	12	5	7	2	

Gambar 3.6 Proses *Neighborhood-1*

Populasi baru yang dihasilkan dari pertukaran nilai gen ke-x ke nilai gen ke-y. Individu baru menghasilkan nilai *fitness* sebesar 0.60331825. Kemudian menghitung selisih *fitness* antara *offspring* C3 dengan individu baru. Individu baru dapat diterima menggantikan individu lama jika Δf lebih besar atau sama dengan 0 serta memenuhi persamaan (2.11) yaitu dengan melakukan perbandingan antara probabilitas Boltzman dengan bilangan random antara 0 hingga 1. Selisih nilai *fitness* individu x_n dan individu x_p sebagai berikut.

$$\begin{aligned}
 \Delta f &= f(x_p) - f(x_n) \\
 &= 0.60331825 - 0.635364381 \\
 &= -0.032046131
 \end{aligned}$$

Karena hasil dari selisih *fitness* baru dengan *fitness* lama diperoleh nilai *fitness* kurang dari 0, maka selajutnya akan dilakukan perbandingan antara probabilitas Boltzman dengan angka random. Angka randomnya dimisalkan adalah 0,997.

$$\text{Exp} \left(\frac{\Delta f}{T} \right) \geq \text{random} (0,1)$$

$$\text{Exp} \left(-\frac{0.032046131}{0,25} \right) > 0.997 \dots$$

$$0.879 > 0.997$$

Diketahui bahwa nilai dari probabilitas Boltzman kurang dari nilai random, maka individu x_p tidak dapat menggantikan individu x_n , sehingga terjadi penurunan temperatur menggunakan persamaan (2.22).

$$T_1 = 0,4 \times 5 = 2,0$$

Setelah temperatur diturunkan menjadi T_1 , maka dilakukannya pengecekan apakah temperatur tersebut sudah mencapai temperatur akhir. Jika nilai dari T_1 masih diatas T_n maka proses *simulated annealing* akan melakukan perulangan kembali, namun jika $T_1 < T_n$ maka proses akan berhenti. Sehingga individu x_p akan menggantikan individu x_n serta dijadikan sebagai solusi baru dan merupakan hasil dari proses optimasi menggunakan hybrid algoritma genetika dan *simulated annealing*. Namun jika saat kondisi $T_1 > T_n$ maka akan dilakukan perulangan kedua dengan diawali proses neighborhood ke-2 hingga mencapai solusi optimum.

Sehingga dipilih individu baru dengan mengganti individu lama. Dengan menghasilkan individu terbaik yaitu individu baru dari hasil optimasi *hybrid* algoritma genetika dan *simulated annealing*. Ditunjukkan pada Tabel 3.12

Tabel 3.12 Hasil Optimasi *Hybrid* Algoritma Genetika dan
Simulated Annealing

	SP	SN	SH	S	B	SP	SN	SH	S	B	SP	SN	SH	S	B
Xp	13	10	8	4	14	3	11	15	6	1	9	12	5	7	2

Dengan komposisi bahan makanan untuk satu hari, komposisi bahan makanan pagi diantaranya soun, kecap, telur ayam, sawi dan salak. Komposisi bahan makanan siang diantaranya beras ketan hitam, toge, kepiting, daun singkong dan anggur. Komposisi bahan makanan malam diantaranya jagung kuning, kecipir, ikan mujair, kangkung dan apel.

3.7 Implementasi

Sistem optimasi komposisi bahan makanan bagi penderita diabetes melitus pada penelitian ini menggunakan metode algoritma genetika dan *simulated annealing* untuk membantu mempermudah pasien dalam memilih bahan makanan agar pemilihan kalori pada bahan makanannya tidak berlebihan terhadap kalori yang dibutuhkan pasien. Langkah pertama yang dilakukan adalah inisialisasi parameter.

3.7.1 Inisialisasi Parameter

Sebelum mengimplentasikan metodenya, terlebih dahulu ada proses inisialisasi parameter yang meliputi data pasien diantaranya jenis kelamin, umur, tinggi badan, aktivitas dan berat badan. Serta juga beberapa parameter yang dibutuhkan seperti *popsize*, *crossover* (C_r), mutasi (M_r), jumlah generasi, temperatur awal (T_0), temperatur akhir (T_n) dan *alpha* (α).

3.7.2 Inisialisasi Populasi Awal

Pada proses ini dimulai dengan melakukan pemanggilan method inisialisasi yang memiliki parameter yaitu *popsize* dan kromosom. Dapat dilihat kode program dari inisialisasi populasi awal pada Gambar 4.1

1	public void Inisialisasi(int PopSize) {
2	int satu_hari = 3;
3	Chromosome = satu_hari * 5;
4	this.Populasi_Awal = new int[PopSize][Chromosome];
5	for (int i = 0; i < PopSize; i++) {
6	for (int j = 0; j < Chromosome; j++) {

7	this.Populasi_Awal[i][j]=random.nextInt(15-1)+1;
8	}
9	}
10	}
11	

Gambar 3.7 Kode Program untuk Inisialisasi Populasi Awal

Penjelasan dari Gambar 3.7 adalah sebagai berikut.

1. Pada baris 2-3 digunakan untuk menentukan berapa jumlah kromosom.
2. Pada baris 4 digunakan mengalokasikan array untuk kromosom induk sesuai dengan jumlah popsize dan jumlah kromosom.
3. Pada baris 5-7 adalah proses perulangan array sejumlah popsize dan kromosom.

3.7.3 Implementasi Crossover

Kemudian dilakukan proses crossover sebanyak crossover rate sesuai input pengguna. Dapat dilihat kode program dari proses crossover pada Gambar 3.8 dibawah.

1	Public void CrossOver(int PopAwal[][], double Pcr) {
2	this.Off_CrossOver=(int) (Math.round(Pcr*PopAwal.length
3	));
4	this.Hasil_CrossOver=new
5	int[Off_CrossOver][Chromosome];
6	int hitung = 0;
7	
8	while (hitung < Off_CrossOver) {
9	int Parent1 = random.nextInt(PopAwal.length);
10	int Parent2 = random.nextInt(PopAwal.length);
11	int OneCut_Point = random.nextInt(Chromosome);
12	
13	while (Parent1 == Parent2) {
14	Parent1 = random.nextInt(PopAwal.length);
15	Parent2 = random.nextInt(PopAwal.length);
16	}
17	
18	if (Off_CrossOver - hitung == 1) {
19	for(int i=0;i<Chromosome;i++) {
20	this.Hasil_CrossOver[hitung][i]=
21	this.Populasi_Awal[Parent1][i];
22	}

```

23     for (int j = OneCut_Point; j < Chromosome; j++) {
24         this.Hasil_CrossOver[hitung][j] =
25 this.Populasi_Awal[Parent2][j];
26     }
27     hitung++;
28 }
29 else {
30     int C1 = hitung;
31     for (int i = 0; i < Chromosome; i++) {
32         this.Hasil_CrossOver[C1][i] =
34 this.Populasi_Awal[Parent1][i];
35     }
36     for (int j = OneCut_Point; j < Chromosome; j++) {
37         this.Hasil_CrossOver[C1][j] =
38 this.Populasi_Awal[Parent2][j];
39     }
40     hitung++;
41     int C2 = hitung;
42     for (int i = 0; i < Chromosome; i++) {
43         this.Hasil_CrossOver[C2][i] =
44 this.Populasi_Awal[Parent2][i];
45     }
46     for (int j = OneCut_Point; j < Chromosome; j++) {
47         this.Hasil_CrossOver[C2][j] =
48 this.Populasi_Awal[Parent1][j];
49     }
50     hitung++;
51 }
52 }
53 }
54 }
55
56

```

Gambar 3.8 Kode Program untuk Crossover

Penjelasan dari Gambar 3.8 adalah sebagai berikut.

1. Pada baris 1 adalah pendeklarasian method dan parameternya.
2. Pada baris 3 adalah proses untuk menghitung banyaknya child crossover yang bisa dihasilkan dengan Crossover rate yang di inputkan.
3. Pada baris 5-9 proses untuk mengalokasikan array anak sejumlah off_crossover.
4. Pada baris 10-17 adalah proses pengambilan induk secara random.
5. Pada baris 19-51 adalah proses untuk penentuan anak kromosom.

3.7.4 Implementasi Mutasi

Potongan kode program yang digunakan untuk proses mutasi dengan menggunakan insertion mutation sebanyak mutation rate sesuai inputan awal dapat dilihat pada Gambar 3.9 dibawah.

```

1 public void Mutasi(int[][] PopAwal, double Pmr) {
2     this.Off_Mutasi = (int) (Math.ceil((Math.round(Pmr *
3     100.0) / 100.0) * PopAwal.length));
4     this.Hasil_Mutasi = new int[Off_Mutasi][Chromosome];
5
6     for (int i = 0; i < Off_Mutasi; i++) {
7         int Parent = random.nextInt(PopAwal.length);
8
9         int Titik_Mutasi_1 = random.nextInt(Chromosome);
10        int Titik_Mutasi_2 = random.nextInt(Chromosome);
11
12
13        while (Titik_Mutasi_1 == Titik_Mutasi_2) {
14            Titik_Mutasi_1 = random.nextInt(Chromosome);
15            Titik_Mutasi_2 = random.nextInt(Chromosome);
16        }
17
18        for (int j = 0; j < Chromosome; j++) {
19            this.Hasil_Mutasi[i][j] =
20            this.Populasi_Awal[Parent][j];
21        }
22
23
24        InsertionMutation(i, Hasil_Mutasi, Titik_Mutasi_1,
25        Titik_Mutasi_2, Parent);
26    }
27
28 }
29
30 private void InsertionMutation(int i, int[][]
31 Hasil_Mutasi, int Titik_Mutasi_1, int Titik_Mutasi_2,
32 int Parent) {
33     int nilai_1 =
34     this.Populasi_Awal[Parent][Titik_Mutasi_1];
35     int nilai_2 =
36     this.Populasi_Awal[Parent][Titik_Mutasi_2];
37     this.Hasil_Mutasi[i][Titik_Mutasi_1] = nilai_2;
38     this.Hasil_Mutasi[i][Titik_Mutasi_2] = nilai_1;
39
40 }
41

```

42	
43	

Gambar 3.9 Kode Program Insertion Mutation

Penjelasan dari Gambar 3.9 adalah sebagai berikut.

1. Pada baris 1 adalah pendeklarasian method dan parameternya.
2. Pada baris 2-3 adalah proses untuk menghitung banyaknya child mutasi yang bisa didapatkan dengan Mutation rate yang di inputkan.
3. Pada baris 4 digunakan untuk mengalokasikan array anak sejumlah off_mutation.
4. Pada baris 8-17 adalah proses pengambilan induk secara random.
5. Pada baris 25-40 adalah proses Insertion mutation.

3.7.5 Implementasi Populasi Gabungan

Kemudian adalah populasi gabungan merupakan proses penggabungan antara populasi awal dengan semua child yang didapatkan dari proses *crossover* dan mutasi, maka akan menghasilkan populasi gabungan. Kode program untuk proses populasi gabungan dapat dilihat pada Gambar 3.10 dibawah.

1	<code>public void PopAkhir(int Populasi_Awal[][], int</code>
2	<code>Hasil_CrossOver[][], int Hasil_Mutasi[][]) {</code>
3	<code>int Panjang_PopAwal = Populasi_Awal.length;</code>
4	<code>int Panjang_Hasil_CrossOver = Hasil_CrossOver.length;</code>
5	<code>int Panjang_Hasil_Mutasi = Hasil_Mutasi.length;</code>
6	
7	<code>this.Pop_Akhir = Panjang_PopAwal +</code>
8	<code>Panjang_Hasil_CrossOver + Panjang_Hasil_Mutasi;</code>
9	<code>this.Populasi_Akhir = new int[Pop_Akhir][Chromosome];</code>
10	<code>int a = 0, b = 0, c = 0;</code>
11	<code>for (int i = 0; i < Pop_Akhir; i++) {</code>
12	<code>for (int j = 0; j < Chromosome; j++) {</code>
13	
14	<code>if (i < Panjang_PopAwal) {</code>
15	<code> this.Populasi_Akhir[i][j] = Populasi_Awal[a][j];</code>
16	<code>}</code>
17	
18	<code>if (i > Panjang_PopAwal - 1 && i < Panjang_PopAwal +</code>

```

19 Panjang_Hasil_CrossOver) {
20     this.Populasi_Akhir[i][j] = Hasil_CrossOver[b][j];
21 }
22
23     if (i > (Panjang_PopAwal + Panjang_Hasil_CrossOver) -
24 1) {
25         this.Populasi_Akhir[i][j] = Hasil_Mutasi[c][j];
26     }
27 }
28
29     if (i < Panjang_PopAwal) {
30         a++;
31     }
32
33     if (i > Panjang_PopAwal - 1 && i < Panjang_PopAwal +
34 Panjang_Hasil_CrossOver) {
35         b++;
36     }
37 }
38
39     if (i > (Panjang_PopAwal + Panjang_Hasil_CrossOver) - 1)
40 {
41     c++;
42 }
43 }
44 }
45 }
46 }
47 }
48 }

```

Gambar 3.10 Kode Program Populasi Gabungan

Penjelasan dari Gambar 3.10 adalah sebagai berikut.

1. Pada baris 1-2 adalah pendeklarasian method populasi akhir dan parameter berupa populasi awal, hasil *crossover* dan hasil mutasi.
2. Pada baris 3-5 adalah inisialisasi variabel-variabel yang dibutuhkan dalam proses populasi gabungan.
3. Pada baris 14-15 digunakan untuk memasukkan populasi awal ke variabel populasi akhir.
4. Pada baris 18-20 digunakan untuk memasukkan offspring *crossover* ke variabel populasi akhir.

5. Pada baris 23-25 digunakan untuk memasukkan offspring mutasi ke variabel populasi akhir.
6. Pada baris 29-41 adalah proses untuk menginkremenkan setiap variabel yang dibutuhkan untuk mengambil nilai populasi awal, offspring *crossover* dan mutasi.

3.7.6 Implementasi Penalti

Melakukan perhitungan penalti bertujuan mendapatkan nilai *fitness*. Perhitungan penalti dimulai dari menghitung total kandungan gizi dari setiap kromosom. Kemudian akan dihitung nilai penalti dari masing-masing kebutuhan gizi. Maka total penalti didapatkan dari selisih total kandungan dengan nilai penalti dari masing-masing kandungan gizi (Baris 9-10).

```

1 public void Hitung_Penalty(double[][] total_kandungan) {
2     this.total_penalty = new
3     double[total_kandungan.length];
4     double nilai_penalty[] = {2095.85, 314.3764, 523.9625,
5     1257.51};
6     for (int i = 0; i < total_kandungan.length; i++) {
7         for (int j = 0; j < total_kandungan[0].length; j++){
8             this.total_penalty[i] +=
9             Math.abs(total_kandungan[i][j] - nilai_penalty[j]);
10        }
11    }
12
13    int pembulat = 5;
14    for (int i = 0; i < total_penalty.length; i++) {
15        BigDecimal bd = new BigDecimal(total_penalty[i]);
16        bd = bd.setScale(pembulat, BigDecimal.ROUND_UP);
17        total_penalty[i] = bd.doubleValue();
18    }
19 }
20
21 
```

Gambar 3.11 Kode Program Untuk Penalti

3.7.7 Implementasi Fitness

Kode program untuk perhitungan nilai fitness dapat dilihat pada Gambar 3.12. Nilai fitness didapatkan dari nilai 1000 dibagi dengan hasil penjumlahan penalti dengan total harga (baris 5) yang sebelumnya dilakukan proses penjumlahan (baris 4).

1	private double Hitung_Fitness_Xp(double penalty, double
2	total_harga) {
3	double fitness = 0;
4	double tot = penalty + total_harga;
5	fitness = 1000.0 / tot;
6	return fitness;
7	}

Gambar 3.12 Kode Program Perhitungan Fitness

3.7.8 Implementasi Evaluasi dan Seleksi

Pada Gambar 3.13 merupakan kode program dari proses evaluasi dan seleksi, dilakukan dengan melibatkan individu induk dan anak yang nantinya akan diurutkan berdasarkan dari nilai *fitness* tertinggi hingga terkecil.

1	public void Seleksi(double Fitness[], int
2	PopulasiAkhir[][], double Penalty[]) {
3	ArrayList<simpan_data> list = new ArrayList<>();
4	this.Fitness = new double[Populasi_Awal.length];
5	for (int i = 0; i < PopulasiAkhir.length; i++) {
6	list.add(new menyimpan_data(i, Penalty[i],
7	Fitness[i]));
8	}
9	
10	boolean butuh_pelewatan = true;
11	for (int i = 1; i < list.size() && butuh_pelewatan; i++)
12	{
13	butuh_pelewatan = false;
14	for (int j = 0; j < list.size() - i; j++) {
15	if (list.get(j).fitness > list.get(j + 1).fitness)
16	{
17	simpan_data temp = list.get(j);
18	list.set(j, list.get(j + 1));
19	list.set(j + 1, temp);
20	butuh_pelewatan = true;

21	}
22	}
23	}
24	for (int i = 0; i < this.Populasi_Awal.length; i++) {
25	for (int j = 0; j < this.Populasi_Awal[0].length; j++)
26	{
27	this.Populasi_Awal[i][j] =
28	PopulasiAkhir[list.get(i).asal_individu][j];
29	}
30	this.Fitness[i] = list.get(i).fitness;
31	}
32	}
33	}
34	
35	
36	

Gambar 3.13 Kode Program Untuk Evaluasi dan Seleksi

Penjelasan dari Gambar 3.13 adalah sebagai berikut.

1. Pada baris 1-2 adalah pendeklarasian method untuk proses seleksi dan parameter berupa *fitness*, populasi akhir dan penalti.
2. Pada baris 3 adalah fungsi array list yang digunakan untuk menyimpan data.
3. Pada baris 6-8 digunakan untuk proses masukkan nilai *fitness*, penalti dan individu ke dalam array list.
4. Pada baris 11-21 merupakan fungsi untuk sorting array list berdasarkan nilai *fitness* dengan bubble sort.
5. Pada baris 26-32 adalah proses seleksi populasi awal dengan populasi akhir yang sudah diurutkan.

3.7.9 Implementasi Simulated Annealing

Pada metode simulated annealing, proses awal adalah mengambil hasil terbaik dari metode algoritma genetika dan mengambil temperature awal yang telah di inputkan user kemudian menukar salah satu titik acak di kromosom terbaik di algoritma genetika yang bisa dilihat pada Gambar 3.14 dibawah.

```

1 public void SimulatedAnealling(int Populasi_Awal[][],
2 double Fitness[], double T0, double Tn, double Alpha)
3 throws FileNotFoundException {
4     this.Xn = new int[Chromosome];
5     this.Xp = new int[Chromosome];
6     this.Fitness_Xn = Fitness[0];
7
8     for (int i = 0; i < Chromosome; i++) {
9         this.Xn[i] = Populasi_Awal[0][i];
10    }
11    for (int i = 0; i < Chromosome; i++) {
12        this.Xp[i] = Xn[i];
13    }
14    while (T0 > Tn) {
15        int titik1 = random.nextInt(Chromosome);
16        int titik2 = random.nextInt(Chromosome);
17
18        //acak titik2 jika titik1 = titik2
19
20        while (titik1 == titik2) {
21            titik1 = random.nextInt(Chromosome);
22            titik2 = random.nextInt(Chromosome);
23        }
24        this.Xp = Neighborhood(titik1, titik2);
25    }
26
27    this.Total_Kandungan_Xp = Hitung_Total_Kandungan(Xp);
28    this.Total_Harga_Xp = Hitung_Total_Harga(Xp);
29    this.Penalty_Xp = Hitung_penalty(Total_Kandungan_Xp);
30    this.Fitness_Xp = Hitung_Fitness_Xp(Penalty_Xp,
31    Total_Harga_Xp);
32
33    if (Fitness_Xp > Fitness_Xn) {
34        for (int j = 0; j < Chromosome; j++) {
35            this.Xn[j] = this.Xp[j];
36        }
37        this.Fitness_Xn = this.Fitness_Xp;
38    } else {
39        double Selisih_Fitness = this.Fitness_Xp -
40    this.Fitness_Xn;
41        double P = Math.exp(-(Selisih_Fitness / T0));
42        double R = random.nextDouble();
43        if (P > R) {
44            for (int j = 0; j < Chromosome; j++) {
45                this.Xn[j] = this.Xp[j];
46            }
47            this.Fitness_Xn = this.Fitness_Xp;

```

48	}
49	}
50	T0 = Alpha * T0;
51	}
52	
53	}
54	
55	
56	
57	
58	
59	
60	

Gambar 3.14 Kode Program untuk Proses Simulated Annealing

Penjelasan dari Gambar 3.14 adalah sebagai berikut.

1. Pada baris 5-10 adalah inisialisasi individu yang akan di eksekusi dengan simulated annealing.
2. Pada baris 13 adalah perulangan suhu awal yang sudah di tentukan sampai suhu akhir yang di inginkan.
3. Pada baris 17-19 adalah proses untuk penukaran partikel dalam individu yang terpilih.
4. Pada baris 23 adalah fungsi untuk membuat solusi baru
5. Pada baris 30 adalah fungsi untuk menghitung fitness baru dari hasil penukaran partikel di individu.
6. Pada baris 36 - 44 ialah proses perbandingan fitness baru dengan fitness lama jika fitness baru lebih besar dari fitness lama maka fitness dan solusi baru yang akan di pakai jika tidak maka fitness lama yang akan di pakai.
7. Pada baris 45 ialah proses pengurangan suhu.

3.7.10 Implementasi Proses Neighborhood

Pada proses *neighborhood* merupakan lanjutan dari proses simulated annealing dengan mengambil hasil terbaik dari algoritma genetika dan mengambil temperatur awal yang telah di inputkan user lalu menukar salah satu titik acak di

chromosome terbaik di algoritma genetika yang dapat dilihat pada Gambar 3.15 dibawah.

1	boolean flag = random.nextBoolean();
2	if (flag) {
3	int titik1 = random.nextInt(Chromosome);
4	int titik2 = random.nextInt(Chromosome);
5	//acak titik2 jika titik1 = titik2
6	while (titik1 == titik2) {
7	titik1 = random.nextInt(Chromosome);
8	titik2 = random.nextInt(Chromosome);
9	}
10	this.Xp = Neighborhood(titik1, titik2);
11	}
12	

Gambar 3.15 Kode Program untuk proses Neighborhood

Penjelasan dari Gambar 3.15 adalah sebagai berikut.

1. Pada baris 1 digunakan untuk penentuan flag
2. Pada baris 3-8 adalah pencarian titik yang di tukar.
3. Pada Bbris 9-11 adalah proses untuk penukaran partikel dalam individu yang terpilih.

BAB IV

UJI COBA DAN PEMBAHASAN

4.1 Hasil Pengujian

Pengujian yang dilakukan yaitu pengujian sistem, pengujian ukuran populasi, banyaknya generasi, kombinasi antara *crossover rate* (C_r) dan *mutation rate* (M_r), temperatur awal (T_0), serta pengujian α (α). Hasil pengujian akan dianalisis sehingga dapat disimpulkan nilai-nilai parameter yang menghasilkan nilai *fitness* terbaik.

4.1.1 Pengujian Sistem

Pengujian terhadap sistem optimasi bahan makanan bagi penderita diabetes melitus menggunakan metode hybrid algoritma genetika dan simulated annealing yang meliputi input data dan parameter, pengujian populasi awal, pengujian *crossover*, pengujian *mutation*, pengujian populasi gabungan, pengujian seleksi dan *simulated annealing*, pengujian individu terbaik serta output sistem.

4.1.1.1 Input Data dan Parameter

Proses input data pasien diabetes melitus. Data pasien yang di input adalah berat badan, tinggi badan, jenis kelamin, umur, dan aktivitas yang dilakukan pasien. Parameter yang di input berupa jumlah generasi, *popsiz*e, *crossover rate* (C_r), *mutation rate* (M_r), temperatur awal (T_0), temperatur akhir (T_n), serta *alpha* (α).

Gambar 4.1 Input Data Pasien Dan Parameter

4.1.1.2 Hasil Pengujian Populasi Awal

Gambar 4.2 Menunjukkan hasil populasi awal yang didapatkan dari pemasukan data satu pasien dan parameter yang sudah ditetapkan diatas. Terdapat 3 individu karena menginputkan parameter *pop size* sebanyak 3 diantaranya individu P1, P2, dan P3 dengan susunan bahan makanan secara random, serta menghitung nilai *fitness* dari masing-masing individu.

Populasi Awal																
Individu	SP	SN	SH	S	B	SP	SN	SH	S	B	SP	SN	SH	S	B	fitness
P1	11	5	4	6	2	5	14	2	7	2	8	8	1	11	5	0.6630024729992243
P2	7	10	11	3	9	5	9	4	9	9	10	14	2	5	12	0.7803416274778451
P3	1	7	5	2	2	2	3	14	2	9	2	1	11	11	7	0.5754137224664534

Gambar 4.2 Hasil Populasi Awal

4.1.1.3 Hasil Pengujian Crossover

Pada proses crossover ini menggunakan metode *one cut point crossover* dan memperoleh sebanyak 2 child dikarenakan perkalian antara *popsiz*e dengan *croosover rate* menghasilkan nilai *offspring* sebesar 1,80 maka dibulatkan

menjadi 2, dengan itu maka child yang dihasilkan berupa 2 Child yaitu C1 dan C2. Dapat dilihat pada gambar 4.3

Crossover																
Individu	SP	SN	SH	S	B	SP	SN	SH	S	B	SP	SN	SH	S	B	fitness
C1	11	5	4	6	2	5	3	14	2	9	2	1	11	11	7	0.5567649728020311
C2	1	7	5	2	2	2	14	2	7	2	8	8	1	11	5	0.6896076132680505

Gambar 4.3 Hasil Proses Crossover

4.1.1.4 Hasil Pengujian Mutation

Pada proses mutation ini menggunakan metode *insertion mutation* dan untuk menentukan nilai *offspring* sama seperti pada proses *crossover* dengan mengkalikan popsize dengan *mutation rate* dan menghasilkan nilai offspring sebesar 1,2 yang dibulatkan keatas menjadi 2. Sehingga menghasilkan child sejumlah 2 juga yaitu C3 dan C4. Dapat dilihat pada gambar 4.4 dibawah.

Mutasi																
Individu	SP	SN	SH	S	B	SP	SN	SH	S	B	SP	SN	SH	S	B	fitness
C3	7	10	11	3	9	5	9	4	9	9	10	14	2	5	12	0.7803416274778451
C4	1	7	1	2	2	2	3	14	2	9	2	5	11	11	7	0.5618040620579778

Gambar 4.4 Hasil Proses Mutation

4.1.1.5 Hasil Pengujian Populasi Gabungan

Hasil pupulasi gabungan didapatkan dari gabungan individu yang didapatkan dari proses *one cut point crossover* dan *insertion mutation*. Dengan penggabungan itu mengasilkan individu sebanyak 7 diantaranya dari populasi awal sebanyak 3, dari proses *crossover* sebanyak 2 dan dari proses *mutation* sebanyak 2. seperti pada gambar 4.5

Populasi Gabungan																	
Individu	SP	SN	SH	S	B	SP	SN	SH	S	B	SP	SN	SH	S	B	fitness	
P1	11	5	4	6	2	5	14	2	7	2	8	8	1	11	5	0.6630024729992243	
P2	7	10	11	3	9	5	9	4	9	9	10	14	2	5	12	0.7803416274778451	
P3	1	7	5	2	2	2	3	14	2	9	2	1	11	11	7	0.5754137224664534	
C1	11	5	4	6	2	5	3	14	2	9	2	1	11	11	7	0.5567649728020311	
C2	1	7	5	2	2	2	14	2	7	2	8	8	1	11	5	0.6896076132680505	
C3	7	10	11	3	9	5	9	4	9	9	10	14	2	5	12	0.7803416274778451	
C4	1	7	1	2	2	2	3	14	2	9	2	5	11	11	7	0.5618040620579778	

Gambar 4.5 Hasil Populasi Gabungan

4.1.1.6 Hasil Pengujian Seleksi dan Simulated Annealing

Pada gambar 4.6 menunjukkan hasil seleksi dari populasi gabungan yang diambilnya nilai *fitness* terbesar dari ketujuh individu yang didapatkan dan proses seleksi bisa diartikan proses akhir dari metode algoritma genetika. Kemudian dilanjutkan metode *simulated annealing* dengan proses neighborhood yang membandingkan individu terbaik dengan individu baru serta melakukan perbandingan boltzman dan penurunan temperatur.

Hasil Seleksi																	
Individu	SP	SN	SH	S	B	SP	SN	SH	S	B	SP	SN	SH	S	B	fitness	
C3	7	10	11	3	9	5	9	4	9	9	10	14	2	5	12	0.03499636062346968	
C3	7	10	11	3	9	5	9	4	9	9	10	14	2	5	12	0.7803416274778451	
C2	1	7	5	2	2	2	14	2	7	2	8	8	1	11	5	0.6896076132680505	

Gambar 4.6 Hasil Seleksi dan Simulated Annealing

4.1.1.7 Hasil Pengujian Individu Terbaik

Pengujian individu terbaik dari hybrid algoritma genetika dan *simulated annealing* mendapatkan hasil *fitness* tertinggi dengan nilai 0.7803416274778451.

Individu terbaik

Gener...	SP	SN	SH	S	B	SP	SN	SH	S	B	SP	SN	SH	S	B	fitness
1	7	10	11	3	9	5	9	4	9	9	10	14	2	5	12	0.7803416274778451

Gambar 4.7 Hasil Individu Terbaik

4.1.1.8 Output Sistem

Output yang di dapatkan berupa kebutuhan kalori seorang pasien seperti protein, lemak, karbohidrat dalam satuan kilo kalori (Kkal) serta hasil dari penggabungan metode algoritma genetika dan *simulated annealing* yang berupa menu bahan makanan sehari bagi penderita diabetes melitus yang meliputi menu makan pagi, siang, dan malam.

Umur : 55.0							Total Kalori : 1667.25 Kkal						
JK : Laki-laki							Protein : 250.08749999999998 Kkal						
TB : 165.0							Lemak : 416.8125 Kkal						
BB : 95.0							Karbohidrat : 1000.3499999999999 Kkal						
Aktivitas : Sedang													
Makan pagi													
Type	Sumber	Energi	Protein	Lemak	Karbohidrat	Biaya							
SP	Ubi jalar kuning	114.0	0.8	0.5	26.7	850.0							
SN	Tempe kedelai	201.0	20.8	8.8	13.5	1000.0							
SH	Telur ayam	162.0	12.8	11.5	0.7	850.0							
Makan Siang													
Type	Sumber	Energi	Protein	Lemak	Karbohidrat	Biaya							
SP	Beras merah tumbuk	352.0	7.3	0.9	76.2	1500.0							
SN	Keupir	35.0	2.9	0.2	5.8	650.0							
SH	Cumi cumi	75.0	16.1	0.7	0.1	2400.0							
Makan Malam													
Type	Sumber	Energi	Protein	Lemak	Karbohidrat	Biaya							
SP	Kentang	83.0	2.0	0.1	19.1	950.0							
SN	Kluwak	273.0	10.0	24.0	13.5	1600.0							
SH	Bebek	326.0	16.0	28.6	0.0	4500.0							

Gambar 4.8 Output Hasil Hybrid Algoritma Genetika dan *Simulated Annealing*

Dapat dilihat dari gambar 4.8 total kalori yang dibutuhkan pasien sebesar 1667.25 Kkal dengan kebutuhan protein sebesar 250.087, kebutuhan lemak

sebesar 416.812, dan karbohidrat sebesar 1000.349. Sedangkan sistem kurang bisa membantu dalam pemilihan menu makan sehari yang tepat untuk pasien dikarenakan berat bahan makanan yang di masukkan dalam sistem ini hanya 100 gram, jadi output yang dihasilkan sistem kurang mencukupi kebutuhan kalori pasien diabetes melitus.

4.1.2 Hasil Pengujian dan Analisis Ukuran Populasi

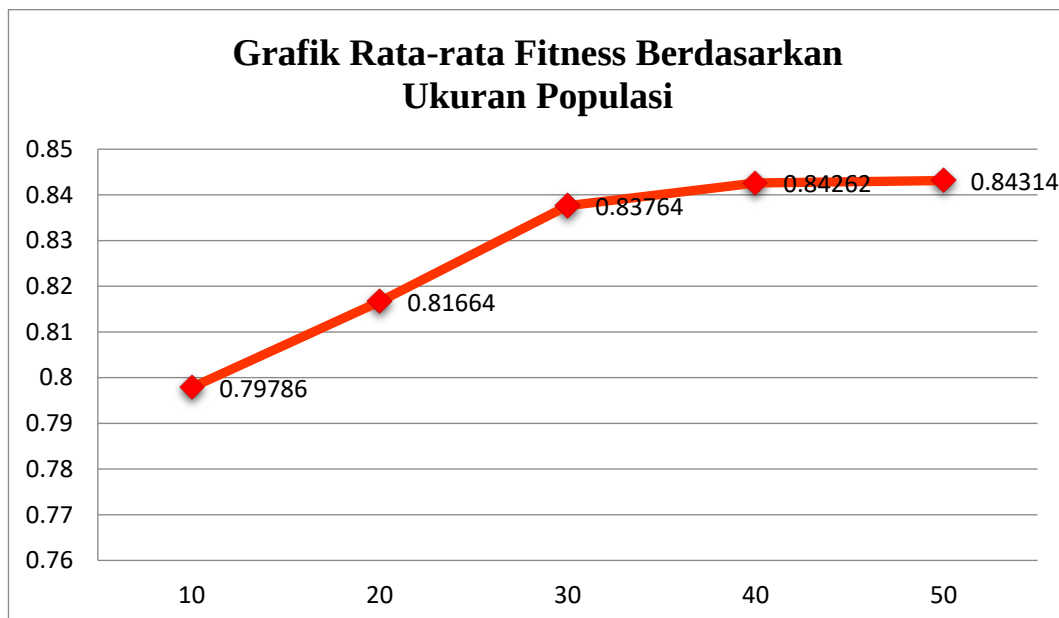
Data yang digunakan adalah 75 data dari bahan makanan yang digolongkan menjadi 5 jenis bahan makanan. Data pasien diabetes melitus yang digunakan antara lain adalah berat badan sebesar 80 kg, tinggi badan sebesar 170 cm, jenis kelamin adalah laki-laki, umur 55 tahun dan aktivitas fisik yang dilakukan adalah sedang. Tabel 4.2 menunjukkan hasil pengujian ukuran populasi (*popsiz*) terhadap perubahan nilai fitness. Dari pengujian ini akan memperoleh rata-rata *fitness* untuk setiap nilai dari *popsiz* dengan pengujian yang dilakukan sebanyak lima kali. Adapun parameter yang digunakan untuk pengujian awal antara lain dengan nilai generasi sebanyak 100 dengan *crossover* rate dan *mutation* rate adalah 0,5 serta temperatur awal adalah 0,8 dan temperatur akhir adalah 0,5 serta nilai alpha adalah 0,4.

Tabel 4.2 Hasil Pengujian Ukuran Populasi (*Popsiz*)

<i>Pengujian Ke-</i>	Ukuran Populasi (<i>Popsiz</i>)				
	10	20	30	40	50
1	0.8513	0.8437	0.8363	0.8493	0.8374
2	0.7759	0.8376	0.8391	0.8343	0.8452
3	0.8207	0.8309	0.8349	0.8467	0.8512
4	0.7719	0.7846	0.8372	0.8459	0.8473
5	0.7695	0.7864	0.8407	0.8369	0.8346
Rata-rata Fitness	0.79786	0.81664	0.83764	0.84262	0.84314

Dilihat dari Tabel 4.2, menunjukkan perubahan rata-rata nilai *fitness* dari ukuran populasi yang berbeda. Pada Gambar 4.9 bisa dilihat bahwa rata-rata *fitness* paling baik yaitu 0.84262 didapatkan dari generasi sebanyak 40 sementara pada generasi 10 menghasilkan rata-rata *fitness* paling kecil yaitu 0.79786. Hal ini menunjukkan apabila semakin besar ukuran populasi, maka nilai rata-rata *fitness* pun semakin besar. Semakin besarnya variasi induk juga dapat memperluas kemungkinan memperoleh individu-individu yang memiliki nilai *fitness* tinggi.

Tetapi tidak menutup kemungkinan pula bahwa nilai rata-rata *fitness* bisa berubah, karena populasi awal dibangkitkan secara acak. Dalam percobaan ini, nilai rata-rata *fitness* untuk ukuran populasi 60, 80 dan 100 tidak mengalami peningkatan yang signifikan. Jadi bisa dikatakan bahwa penambahan ukuran populasi yang lebih besar lagi, tidak meningkatkan rata-rata *fitness* secara signifikan. Perubahan nilai *fitness* ini juga didapatkan oleh pada Hamidah, CP, Cholissodin, I & Nurwansito, H (2016) yang menerapkan *hybrid* algoritma genetika dan *simulated annealing* untuk optimasi susunan bahan makanan untuk keluarga.



Gambar 4.9 Grafik Rata-rata Fitness Berdasarkan Ukuran Populasi

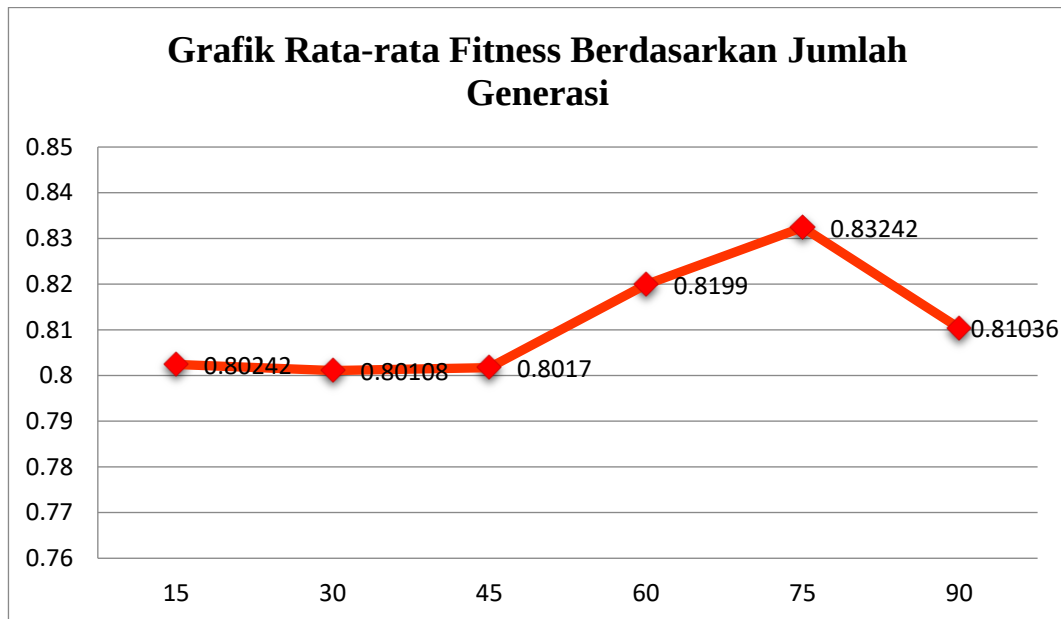
4.1.3 Hasil Pengujian dan Analisis Jumlah Generasi

Hasil pengujian jumlah generasi terhadap perubahan nilai fitness yang dihasilkan ditunjukkan dalam tabel 4.3. Data pasien diabetes melitus yang digunakan adalah berat badan 80 kg, tinggi badan 170 cm, jenis kelamin laki-laki umur 55 tahun dan aktivitas fisik yang dilakukan sedang. Uji coba ini akan dilakukan sebanyak 5 kali. Pengujian awal ini memerlukan parameter yaitu nilai *crossover rate* dan *mutation rate* 0,5 serta temperatur awal dan akhir 0,8 dan 0,4 lalu nilai alpha 0,4. Pada pengujian generasi ini menggunakan ukuran populasi (*popsiz*) sebanyak 40 sesuai dengan hasil pengujian yang sudah dilakukan sebelumnya yang dianggap bisa menghasilkan rata-rata nilai fitness terbaik. Banyaknya generasi yang diuji yaitu 15, 30, 45, 60, 75, dan 90.

Tabel 4.3 Hasil Pengujian Banyaknya Generasi

Pengujian Ke-	Jumlah Generasi					
	15	30	45	60	75	90
1	0.8197	0.7931	0.7856	0.8235	0.8494	0.8224
2	0.8013	0.8212	0.8325	0.8279	0.8455	0.8168
3	0.7952	0.7868	0.7914	0.8315	0.8461	0.8097
4	0.8068	0.8067	0.7936	0.8067	0.7992	0.7903
5	0.7891	0.7976	0.8054	0.8099	0.8219	0.8126
Rata-rata Fitness	0.80242	0.80108	0.8017	0.8199	0.83242	0.81036

Pada tabel 4.3 menunjukkan bahwa rata-rata *fitness* terkecil dengan nilai 0.80108 yang didapatkan dari jumlah generasi sebanyak 30, Sedangkan untuk *fitness* terbesar didapatkan dari jumlah generasi sebanyak 75 dengan rata-rata *fitness* 0.83242. Tetapi pada generasi sebanyak 90 mengalami penurunan.



Gambar 4.10 Grafik Rata-rata Fitness berdasarkan Banyaknya Generasi

4.1.4 Hasil Pengujian dan Analisis Kombinasi C_r dan M_r

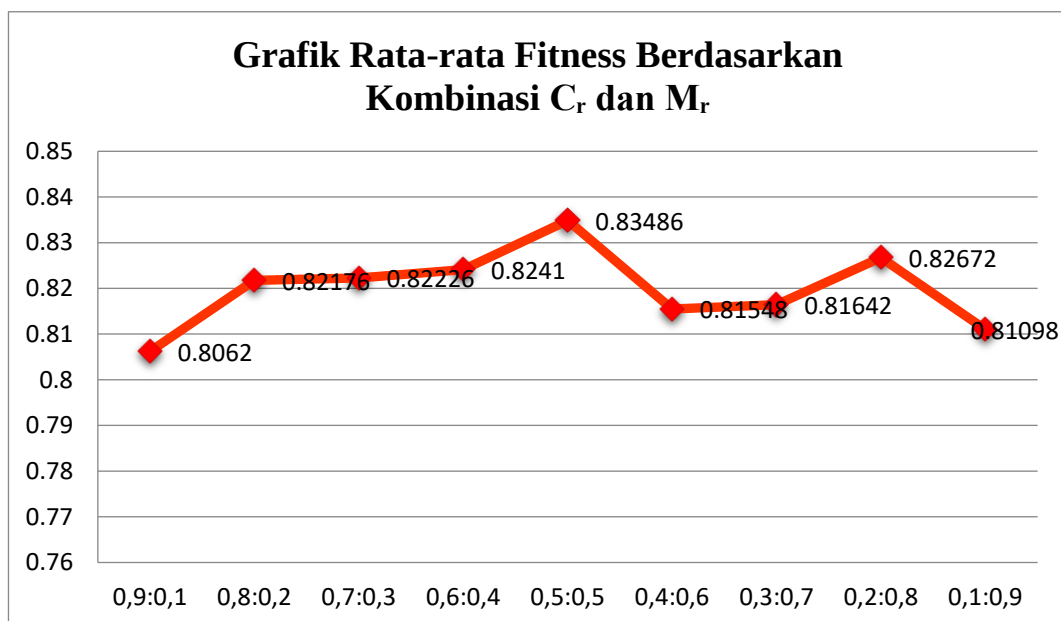
Hasil pengujian berdasarkan kombinasi C_r dan M_r dilakukan untuk mengetahui nilai *fitness* terbaik dari hasil kombinasi tersebut. Tabel 4.4 menunjukkan rata-rata *fitness* untuk setiap nilai dari probabilitas *crossover* dan mutasi dengan melakukan pengujian sebanyak 5 kali. Data pasien diabetes melitus yang digunakan adalah berat badan 80 kg, tinggi badan 170 cm, jenis kelamin laki-laki umur 55 tahun dan aktivitas fisik yang dilakukan sedang. Nilai C_r dan M_r yang digunakan antara 0,1 dan 0.9. Parameter yang dibutuhkan antara lain *popsiz* sebesar 40 nilai generasi sebanyak 75 dengan nilai temperatur awal dan akhir adalah 0,8 dan 0,5 kemudian alpha 0,4. Uji coba ini menggunakan hasil pengujian pada ukuran populasi dan generasi yang telah dilakukan sebelumnya.

Tabel 4.4 Hasil Pengujian Kombinasi C_r dan M_r

Kombinasi C_r dan M_r	Percobaan ke-					Rata-rata
	1	2	3	4	5	
0,9 ; 0,1	0.7968	0.8057	0.8204	0.8235	0.7846	0.8062
0,8 ; 0,2	0.8093	0.8207	0.8321	0.8121	0.8346	0.82176

0,7 ; 0,3	0.7982	0.8172	0.8441	0.8201	0.8317	0.82226
0,6 ; 0,4	0.8305	0.8238	0.8303	0.8258	0.8101	0.8241
0,5 ; 0,5	0.8475	0.8538	0.8093	0.8208	0.8429	0.83486
0,4 ; 0,6	0.8188	0.8024	0.8253	0.8214	0.8095	0.81548
0,3 ; 0,7	0.8375	0.7892	0.8264	0.8169	0.8121	0.81642
0,2 ; 0,8	0.8239	0.8421	0.8269	0.8328	0.8079	0.82672
0,1 ; 0,9	0.8128	0.8187	0.8116	0.7897	0.8221	0.81098

Hasil dari Tabel 4.4 dapat dilihat bahwa rata-rata nilai *fitness* terbesar terdapat pada kombinasi C_r 0,5 dan M_r 0,5 yaitu 0.83486 sedangkan kombinasi C_r 0,9 dan M_r 0,1 menghasilkan rata-rata nilai *fitness* terkecil yaitu sebesar 0.8062 sebagaimana ditunjukkan pada Gambar 6.1. Nilai *crossover* yang terlalu kecil dan mutasi yang terlalu besar menyebabkan menurunnya kemampuan algoritma genetika untuk belajar ke generasi sebelumnya dan tidak mampu mengeksplorasi daerah optimum lokal (Mahmudy, 2014). Selain itu nilai bila tingkat *crossover* rendah dan tingkat mutasi terlalu besar mengakibatkan tidak mampu mengeksplorasi daerah optimum lokal.



Gambar 4.11 Grafik Rata-rata Fitness Berdasarkan Kombinasi C_r dan M_r

4.1.5 Hasil Pengujian dan Analisis Temperatur awal (T_0)

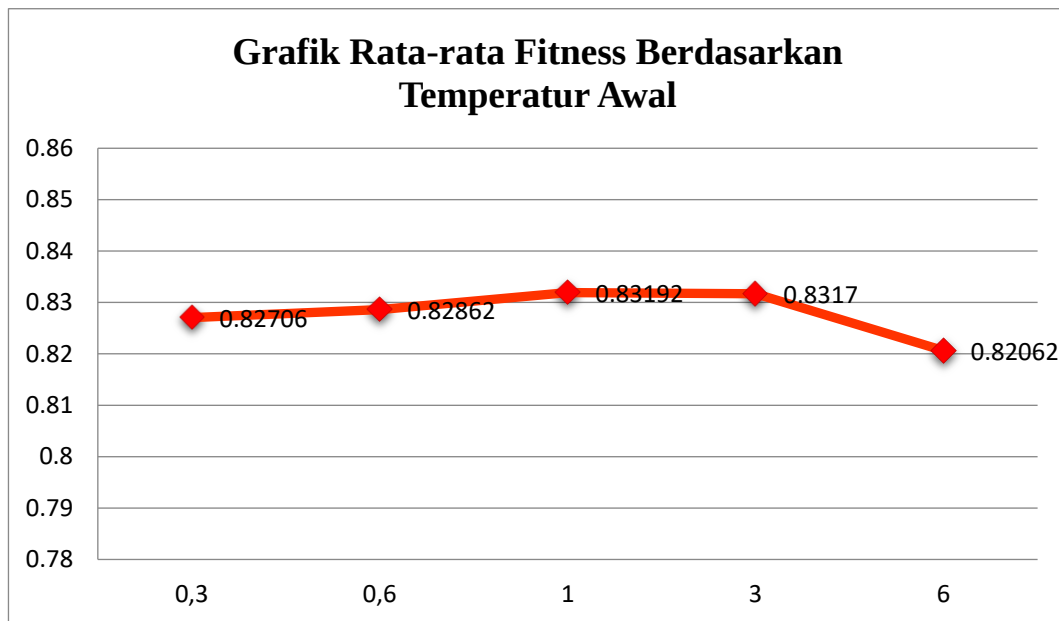
Pada tabel 4.5 menunjukkan hasil rata-rata *fitness* dari pengujian temperatur awal. Pengujian temperatur awal (T_0) berfungsi untuk mengetahui nilai (T_0) yang menghasilkan *fitness* terbaik. Pengujian dilakukan sebanyak 8 kali dan akan dihitung rata-rata nilai *fitness* yang dihasilkan. Data pasien diabetes melitus yang digunakan adalah berat badan 80 kg, tinggi badan 170 cm, jenis kelamin laki-laki umur 55 tahun dan aktivitas fisik yang dilakukan sedang. Parameter yang dibutuhkan antara lain *popsize* sebesar 40 nilai generasi sebanyak 75 dengan nilai temperatur akhir adalah 0,5 dan alpha sebesar 0,4 sesuai dengan hasil pengujian parameter yang dilakukan sebelumnya yang menghasilkan rata-rata *fitness* tertinggi.

Tabel 4.5 Hasil Pengujian Temperatur awal (T_0)

Pengujian Ke-	Temperatur Awal (T_0)				
	0,3	0,6	1	3	6
1	0.8184	0.8181	0.8087	0.8407	0.8501
2	0.8244	0.8332	0.8399	0.8431	0.8116
3	0.8297	0.8197	0.8286	0.8133	0.8152
4	0.8483	0.8194	0.8321	0.8568	0.8457
5	0.8145	0.8527	0.8503	0.8046	0.7805
Rata-rata fitness	0.82706	0.82862	0.83192	0.8317	0.82062

Pada Tabel 4.5 dapat dilihat bahwa nilai rata-rata *fitness* terbesar terdapat pada temperatur awal dengan nilai 1. Nilai rata-rata *fitness* mengalami penurunan pada nilai 0,3 hingga 1. Akan tetapi pada temperatur awal dengan nilai 1 hingga 6 mengalami penurunan sebagaimana ditunjukkan pada Gambar 4.12. Iterasi pada *simulated annealing* semakin banyak jika nilai temperatur awal lebih besar. Namun hal tersebut tidak menjamin jika solusi yang dihasilkan juga lebih baik. Penyebabnya adalah temperatur awal yang besar maka nilai probabilitas

Boltzman juga akan semakin besar. Hal ini dikarenakan solusi akan selalu diterima meskipun solusi tersebut buruk atau memiliki nilai *fitness* rendah. Nilai rata-rata *fitness* terbesar yaitu 0.83192 sedangkan rata-rata *fitness* terkecil yaitu 0.82062.



Gambar 4.12 Grafik Rata-rata Fitness Berdasarkan Temperatur Awal (T_0)

4.1.6 Hasil Pengujian dan Analisis Alpha (α)

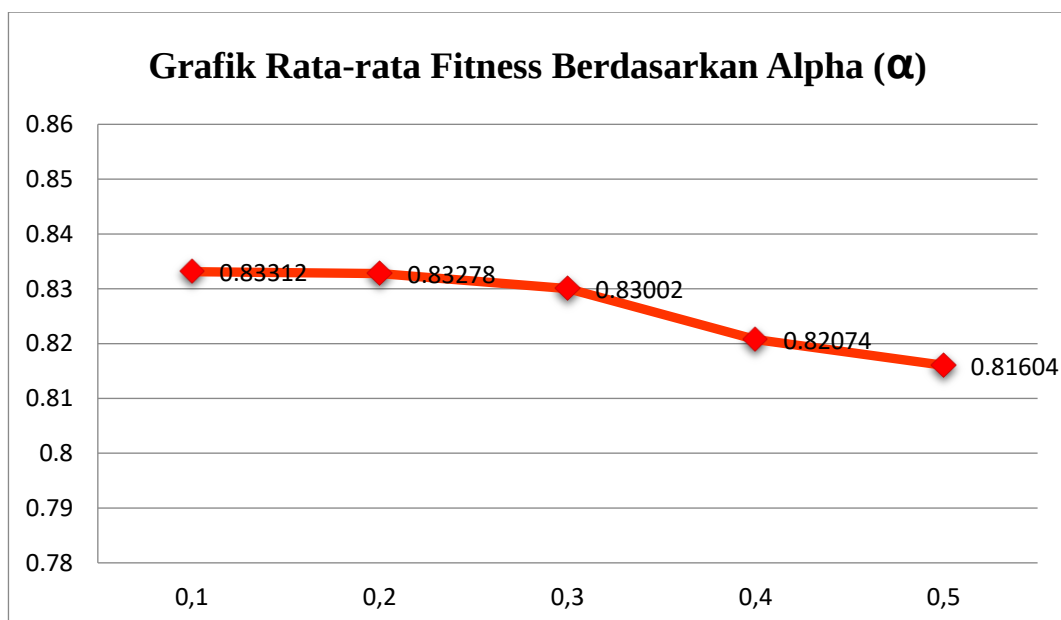
Pada tabel 4.6 menunjukkan hasil nilai *fitness* untuk pengujian alpha (α) sebanyak 5 kali pengujian. Pengujian temperatur awal (T_0) berfungsi untuk mengetahui nilai (T_0) yang menghasilkan *fitness* terbaik. Data pasien diabetes melitus yang digunakan adalah berat badan 80 kg, tinggi badan 170 cm, jenis kelamin laki-laki umur 55 tahun dan aktivitas fisik yang dilakukan sedang. Pengujian ini menggunakan popsize sebesar 40 jumlah generasi sebanyak 75 dengan nilai temperatur 1 sedangkan temperatur akhir adalah 0,5.

Tabel 4.6 Hasil Pengujian Alpha (α)

Pengujian Ke-i	Alpha (α)				
	0,1	0,2	0,3	0,4	0,5

1	0.8377	0.8161	0.8432	0.7999	0.7959
2	0.8539	0.8288	0.8068	0.8264	0.8315
3	0.8269	0.8389	0.8394	0.8494	0.8145
4	0.8286	0.8512	0.8273	0.8214	0.8432
5	0.8185	0.8289	0.8334	0.8066	0.7951
Rata-rata Fitness	0.83312	0.83278	0.83002	0.82074	0.81604

Dari Tabel 4.6 dapat disusun grafik untuk perubahan nilai rata-rata *fitness* sebagaimana ditunjukkan pada gambar 4.13. Gambar tersebut menunjukkan *fitness* terbesar berada pada nilai alpha sebesar 0,1 yaitu 0.83312 dan terkecil pada nilai 0,5 sebesar 0.81604. Hal ini terjadi karena nilai alpha sangat berpengaruh pada jumlah iterasi proses *simulated annealing*. Ruang pencarian semakin luas jika nilai alpha semakin kecil dan iterasi semakin banyak.



Gambar 4.13 Grafik Rata-rata *Fitness* Berdasarkan Alpha (α)

4.2 Pembahasan

Perbandingan hasil uji memakai data pasien diabetes melitus yang digunakan adalah berat badan 80 kg, tinggi badan 170 cm, jenis kelamin laki-laki umur 55 tahun dan aktivitas fisik yang dilakukan sedang. Serta hasil parameter dengan nilai fitness terbaik yang didapatkan dari pengujian sebelumnya antara lain dengan nilai *popsiz*e sebanyak 40 dengan generasi 75 dengan *crossover* rate dan *mutation* rate adalah 0,5 serta temperatur awal dan akhir adalah 1 dan 0,5 kemudian nilai alpha 0,1.

Tabel 4.7 Perbandingan Nilai *Fitness* Algoritma Genetika dan Hybrid GA-SA

Pengujian ke-	Nilai Fitness	
	Algoritma Genetika	GA-SA
1	0.7744133818632386	0.8498487197041785
2	0.7793138859761383	0.8325840077263796
3	0.7818669357706711	0.8234519036277017
4	0.7532956685499058	0.8190813116843481
5	0.7484021557843842	0.8160001239014588
6	0.7660070426515448	0.8223458830101133
7	0.7816654540769907	0.8379937029411303
8	0.7978448944007532	0.8421948900923124
9	0.7570654899544321	0.8182031704483174
10	0.7758008654890425	0.8364796548906541

Melihat hasil rata-rata nilai *fitness* antara metode algoritma genetika dengan *hybrid* algoritma genetika dan *simulated annealing* (GA-SA) yang menggunakan nilai parameter yang terbaik dari hasil uji coba sebelumnya menunjukkan bahwa nilai *fitness* dari GA-SA lebih tinggi daripada nilai *fitness* yang dihasilkan metode algoritma genetika saja. Maka dengan melakukan

penggabungan kedua metode algoritma genetika dan simulated annealing dapat meningkatkan nilai *fitness* dari penggunaan metode algoritma genetika saja.

Dalam riwayat Muslim, beliau shallallahu ‘alaihi wa sallam mengatakan itu bukanlah jenis makanan beliau,

كُلُوا فَإِنَّهُ حَلَالٌ وَلَكِنَّهُ لَيْسَ مِنْ طَعَامِي

“Makanlah oleh kalian, karena sesungguhnya daging ini halal. Akan tetapi bukan dari makananku” [HR. Muslim no. 3608]

Ibnu Taimiyyah menjelaskan dari hadits ini, bahwa disunnahkan makan dan berpakaian sesuai dengan kebiasaan kaumnya dan negerinya, beliau berkata,

فَسُنَّتُهُ فِي ذَلِكَ تَقْتَضِي أَنْ يَلْبَسَ الرَّجُلُ وَيَطْعَمَ مِمَّا
يَسَّرَهُ اللَّهُ بِبَلَدِهِ مِنَ الطَّعَامِ وَاللِّبَاسِ . وَهَذَا يَتَنَوَّعُ
بِتَنَوُّعِ الْأَمْصَارِ

“Sunnah dalam hal ini adalah hendaknya seseorang memakai pakaian dan memakan apa yang telah Allah mudahkan (tersedia) di negerinya/kaumnya berupa makanan dan pakaian. Hal ini berbeda-beda sesuai dengan (keadaan) negerinya.” [Majmu’ fatawa 22/310]

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Setelah melakukan beberapa uji coba sistem optimasi komposisi bahan makanan bagi penderita diabetes melitus menggunakan metode *hybrid* algoritma genetika dan *simulated annealing* dapat ditarik kesimpulan sebagai berikut :

- a. Penggabungan antara metode algoritma genetika dengan *simulated annealing* ini mendapatkan hasil lebih baik dibandingkan hanya menggunakan metode algoritma genetika.
- b. Rata-rata nilai fitness yang dihasilkan dari uji coba menggunakan algoritma genetika sebesar 0.771567577, sedangkan dengan *hybrid* algoritma genetika dan *simulated annealing* sebesar 0.829818337.
- c. Sistem optimasi komposisi bahan makanan bagi penderita diabetes melitus menggunakan metode *hybrid* algoritma genetika dan *simulated annealing* pada penelitian ini kurang mampu menghasilkan komposisi bahan makanan yang tepat bagi penderita diabetes melitus dikarenakan komposisi bahan makanan yang di inputkan hanya 100 gram.

5.2 Saran

Berdasarkan penelitian pada optimasi komposisi bahan makanan bagi penderita diabetes melitus menggunakan metode *hybrid* algoritma genetika dan *simulated annealing* yang telah di lakukan, disarankan untuk mengembangkan sistem yang lebih baik untuk penelitian selanjutnya dan sistem dapat bermanfaat bagi penderita diabetes melitus.

DAFTAR PUSTAKA

- Abdul Ghoffar, M., 2003. *Tafsir Ibnu Katsir Jilid 3*. Pustaka Imam asy-Syafi'i.
- Adi Bhasyudewo Kawi, Sri Widowati, Bedy Purnama, 2011. *Analisis Dan Implementasi Algoritma Genetika Dalam Sistem Pakar Untuk Merancang Bahan makanan Diet*.
- Almatsier, S., 2006. *Prinsip Dasar Ilmu Gizi*. Ed.6. Jakarta : Gramedia Pustaka Utama.
- Almatsier, S., 2009. *Prinsip Dasar Ilmu Gizi*. Jakarta : Gramedia Pustaka Utama.
- Artika Rianawati, Wayan Firdaus Mahmudy, 2015. *Implementasi Algoritma Genetika Untuk Optimasi Komposisi Makanan Bagi Penderita Diabetes Mellitus*. Universitas Brawijaya.
- Binti Robiyatul Musanah, 2017. *Optimasi Kebutuhan Gizi Untuk Ibu Hamil Dengan Menggunakan Hybrid Algoritma Genetika Dan Simulated Annealing*. Universitas Brawijaya.
- Cahyaningyas, Y.N., Ratnawati, D.E., & Sutrisno, 2016. *Implementasi Hybrid Algoritma Genetika dan Simulated Annealing untuk Penjadwalan Mata Pelajaran (Studi Kasus SMPN 1 Sukomoro)*. Malang.
- Departemen Kesehatan, 2005. *Daftar Komposisi Bahan Makanan (DKBM)*.
- Dewa Gede Sastra Ananta Wijaya, 2014. *Perencanaan Diet Pada Penderita Diabetes Melitus*. Politeknik Kesehatan Denpasar.
- Dita Garnita, 2012. *Faktor Risiko Diabetes Melitus Di Indonesia (Analisis Data Sakerti 2007)*. Universitas Indonesia.
- Elisa Julie Irianti Siahaan, Imam Cholissodin, M. Ali Fauzi, 2017. *Sistem Rekomendasi Bahan Makanan Bagi Penderita Penyakit Jantung Menggunakan Algoritma Genetika*. Universitas Brawijaya.

- Fitri Anggarsari, Wayan Firdaus Mahmudy, Candra Dewi, 2017. *Optimasi Kebutuhan Gizi untuk Balita Menggunakan Hybrid Algoritma Genetika dan Simulated Annealing*. Universitas Brawijaya.
- Fitri Rahmawati, MP., 2011. *Perencanaan Diet Untuk Penderita Diabetes Mellitus*. Universitas Negeri Yogyakarta.
- Hamidah, C.P., Cholissodin, I., & Nurwansito, H., 2016. *Optimasi Susunan Bahan Makanan Untuk Pemenuhan Gizi Keluarga Menggunakan Hybrid Algoritma Genetika Dan Simulated Annealing*. DORO : Repository Jurnal Mahasiswa PTIIK Universitas Brawijaya.
- Irianto, Pekik, J., 2007. *Panduan Gizi Lengkap Keluarga Dan Olahragawan*. Yogyakarta : Andi Offset.
- John. MF Adam, 2006. *Klasifikasi dan Kriteria Diagnosis Diabetes Melitus yang Baru*. Cermin Dunia Kedokteran.
- Kushardiana, Resthy, 2013. *Penentuan Komposisi Bahan makananan Untuk Penderita Diabetes Mellitus Menggunakan Algoritma Genetika*. Universitas Brawijaya.
- Kusumadewi, S. 2003. *Artificial Intelligent (Teknik dan Aplikasinya)*. Yogyakarta: Graha Ilmu.
- Maryamah, Rekyan Regasari Mardi Putri, Satrio Agung Wicaksono, 2017. *Optimasi Komposisi Makanan Pada Penderita Diabetes Melitus dan Komplikasinya Menggunakan Algoritma Genetika*. Universitas Brawijaya.
- Meicicho Mardiyah K, Marjono. 2012. *Optimalisasi Kebutuhan Gizi Pada Menu Makanan Penderita Diabetes Melitus Dengan Metode Branch And Bound*. Universitas Brawijaya.
- Muhammad Shafaat, Imam Cholissodin, Edy Santoso, 2017. *Optimasi Komposisi Makanan Diet bagi Penderita Hipertensi menggunakan Algoritma Genetika*. Universitas Brawijaya.

- Munirah, M. & Subanar, 2017. *Kajian terhadap beberapa metode optimasi*. UGM Yogyakarta.
- Nurbaiti Wahid, Wayan Firdaus Mahmudy, 2014. *Optimasi Komposisi Makanan Untuk Penderita Kolesterol Menggunakan Algoritma Genetika*. Universitas Brawijaya.
- Orkcu, H.H., 2013. *Subset Selection in Multiple Linier Regresion Models : A Hybrid of Genetic and Simulated Annealing Algorithm*. Applied Mathematics and Computation, vol. 129, no. 23, pp. 11018–11028.
- Pranoto, Agung, Sutjahjo, Ari, Tjokroprawiro, Askandar, Murtiwi, Sri, Wibisono, Soni, dkk, 2011. *Konsensus Pengelolaan dan Pencegahan Diabetes Melitus Tipe 2 Di Indonesia*. Perkumpulan Endokrinologi Indonesia.
- Rismayanti, & Haryanti, T., 2015. *Implementasi Algoritma Genetika Untuk Penjadwalan Mata Pelajaran di SMAN 1 Ciwedey*. Jurnal Ilmiah Komputer dan Informatika.
- Sari, A.P., Mahmudy, W.F., & Dewi, C., 2014. *Optimasi Asupan Gizi Pada Ibu Hamil Dengan Menggunakan Algoritma Genetika*. DORO : Repository Jurnal Mahasiswa PTIIK Universitas Brawijaya, vol.4, no.5
- Soegondo, Sidartawan, dkk, 2009. *Penatalaksanaan Diabetes Melitus Terpadu*. Jakarta: FKUI.
- Sofianti, T.D., 2004. *Penjadwalan Multipurpose Batch Chemical Plant Dengan Metode Optimasi Gabungan : Algoritma Genetika - Simulated Annealing*. Proceedingsa Komputer dan Sistem Intelejen, (pp297-308). Jakarta
- Suryanto, 2007. *Artificial Intelligence Searching, Reasonig, Planning dan Learning*. Bandung: Informatika.
- Territory Organisations (2012). *A Diabetes Information Series From Diabetes Scale*.
- Yogeswaran, M, Ponnambalam, SG & Tiwari, MK, 2009. *'An efficient hybrid evolutionary heuristic using genetic algorithm and simulated annealing*

algorithm to solve machine loading problem in FMS'. International Journal of Production Research, vol. 47, no.19, pp.5421-5448.