BAB II
TINJAUAN PUSTAKA

2.1 Landasan Teori

2.1.1 Algoritma Palgunadi

Algoritma Palgunadi berdasarkan pada permasalahan penyusunan jadwal yang terdiri dari beberapa elemen yaitu: *teacher*(T), *lessons*(L), *class*(C), *room*(R), dan alokasi *timeslots*(S) (Palgunadi, 2012).

Ada beberapa relasi yang terdapat dalam algoritma ini.

- Relasi α, $\alpha: T_x \times C \rightarrow T$, mengatakan bahwa untuk setiap pelajaran dalam kelas tertentu harus memiliki guru setidaknya satu.
- Relasi β, $\beta: T_x \times C \rightarrow R_x S$, mengatakan bahwa untuk setiap pelajaran di kelas harus dilakukan pada ruang yang tersedia dalam slot waktu tertentu.
- Relasi γ, $\gamma: R_x \times S \rightarrow \{0, 1\}$ adalah untuk memeriksa apakah ruangan di slot waktu sudah ditempati atau tidak.
- Relasi δ, $\delta: T_x \times S \rightarrow \{0, 1\}$ adalah untuk memeriksa apakah seorang guru di slot waktu telah ditugaskan untuk mengajar.

Dalam menyelesaikan algoritma ini dibutuhkan beberapa matriks yaitu:

- Matriks TS. Melambangkan hubungan antara Teacher dan timeslot yang berbentuk boolean matriks dan dipetakan dalam bentuk 0 dan 1. Awalnya diberi nilai nol jika belum terjadwal dan diset menjadi 1 jika sudah terjadwal. Conflict terjadi pabila $TS_1(t_1, s) = TS_2(t_2, s)$ dimana $t_1 = t_2$ maka terjadi penugasan seorang guru dalam timeslot yang sama.
- Matriks CS. Melambangkan hubungan antara class dan timeslot yang berbentuk boolean matriks. Hampir sama dengan matriks TS.
- Empat set Matriks penjadwalan, AS merupakan set elemen dalam {TxLxC} yang telah dijadwalkan, ANS set dari elemen {TxLxC} yang belum dijadwalkan, BS merupakan set elemen {RxS} yang telah dijadwalkan, dan ABS set elemen yang telah dijadwalkan secara benar.

Algoritma utama untuk menyelesaikan masalah penjadwalan ini adalah sebagai berikut:

Algorithm generalTCP (T, C, L, R, S)

(1) Create the a-assignment (A), where A = T x C x { L_r U L_b } and split A into two disjoint set A_r = T x C x L_r and A_b = T x C x L_b
(2) Create the b-assignment (B), where B = {R_r U R_b } x S and split B into two disjoint set B_r = R_r x S and B_b = R_b x S
(3) intialize the matrices TS and CS, AS = {} , ANS = {} BS = {}, ABS = {}
(4) while (There is an unexmined element in A) do begin
(4.1) select a be anunexamined element Є A; scheduled = false;
(4.2) if a Є A_r then begin {conducted in lecturing room}
(4.2.1) while (not(scheduled) or (there is an unexmined element in B_r) do begin
(4.2.1.1) select b be an unexamined element Є B_r
(4.2.1.2) if TS(a.t, b.s) = 0 and CS(a.c, b.s) = 0 then begin { check the conflict}
scheduled = true; no conflict}
AS = AS + {a} ; BS = BS + {b} ; ABS = ABS + {(a,b)};
TS(a.t, b.s) = 1 ; CS(a.c, b.s) = 1;
A = A – { a } , B = B – {b}; end
else select b.next ; {conflict still occur, try next room-timeslot}
end
(4.2.2) If not(scedulled) then begin
ANS = ANS + {a} ; A = A – {a} end
else {conducted in lab }
(4.2.3) while (not(scheduled) or (there is an unexmined element in B_b) do begin
(4.2.3.1) select b be an element Є B_b
(4.2.3.2) if TS(a.t, b.s) = 0 and CS(a.c, b.s) = 0 then begin
scheduled = true;
AS = AS + {a} ; BS = BS + {b} ; ABS = ABS + {(a,b)};
TS(a.t, b.s) = 1 ; CS(a.c, b.s) = 1;
A = A – { a } , B = B – {b}; end
else select b.next ;
end;
(4.2.4)) If not(scedulled) then begin
ANS = ANS + {a} ; A = A – {a} end
end;

2.1.2 Algoritma Genetika

Algoritma genetika merupakan metode heuristik adaptif yang memiliki dasar pemikiran atau gagasan untuk proses seleksi alam dan genetika berdasarkan penelitian Charles Darwin. Dengan kata lain pencarian solusi suatu masalah dengan algoritma genetika akan terus berevolusi (Kusumadewi, 2005). Algoritma

genetika pertama kali ditemukan oleh John Holland dari University of Michigan pada tahun 1960, yang diteruskan dengan terbitnya sebuah buku dengan judul *Adaption in Natural and Artificial Systems* pada tahun 1975.

Algoritma genetika yang dibuat Holland merupakan sebuah metode untuk memisahkan satu populasi kromosom (terdiri dari bit-bit 1 dan 0) ke populasi baru dengan menggunakan "seleksi alam" dan operator genetik seperti crossover, *mutation, invertion*. *Crossover* menukar bagian kecil dari dua kromosom, *mutation* mengganti secara acak nilai gen di beberapa lokasi pada kromosom, *invertion* membalikkan urutan beberapa gen yang berurutan dalam kromosom. Dasar teori inilah yang menjadi dasar kebanyakan program yang menggunakan algoritma genetika pada saat ini (Fariza, 2006).

Secara alami semua organisme terdiri dari sel, dimana setiap sel terdiri dari sekumpulan kromosom membentuk sekumpulan gen membuat satu kesatuan yang tersusun dalam rangkaian linear. Setiap gen mempunyai letak tersendiri di dalam kromosom yang disebut dengan lokus. Gen tersusun dari (DNA), yang membawa sifat-sifat keturunan. Setiap gen menyandi protein tertentu suatu sifat. Bagian tertentu dari gen di dalam genom disebut genotip. Beberapa sifat individu yang menunjukkan perbedaan gen dan berada pada bagian disebut alel (Fitri, 2002). Perbandingan istilah alam dengan Algoritma Genetika dapat ditunjukan pada Tabel 2.1.

Tabel 2.1. Perbandingan istilah alam dengan istilah Algoritma Genetika

Sistem Alamiah	Algoritma Genetika
Kromosom	String
Gen	Fitur, Karakter, atau detector
Allel	Nilai fitur
Locus	Posisi String
Genotip	Struktur
Fenotip	Set parameter, solusi alternatif, struktur yang di-decode
Epitasis	Non linieritas

Beberapa definisi penting dalam algoritma genetika, yaitu:

- *Genotype* (Gen) adalah sebuah nilai yang menyatakan satuan dasar yang membentuk suatu arti tertentu dalam satu kesatuan gen yang dinamakan kromosom. Dalam algoritma genetika, gen ini bisa berupa nilai biner, float, integer maupun karakter.
- *Allele* merupakan nilai yang berada dalam gen.
- *Locus* adalah letak suatu gen berada dalam suatu kromosom
- Kromosom adalah gabungan gen-gen yang membentuk nilai tertentu.
- Individu menyatakan satu nilai atau keadaan yang menyatakan salah satu solusi yang mungkin dari permasalahan yang diangkat
- Populasi merupakan sekumpulan individu yang akan diproses bersama dalam satu siklus proses evolusi.
- Generasi menyatakan satu-satuan siklus proses evolusi.
- Anak (*Offspring*) adalah generasi berikutnya yang terbentuk dari gabungan 2 kromosom. Generasi sekarang yang bertindak sebagai induk (parent) dengan menggunakan operator perkawinan (crossover) maupun operator mutasi
- Nilai Fitness menyatakan seberapa baik nilai dari suatu individu atau solusi yang didapatkan.

Algoritma genetika mempunyai karakteristik yang perlu diketahui sehingga dapat dibedakan proses pencarian atau optimasi yang lainnya. Karakteristik-karakteristik yang perlu diketahui sehingga dapat dibedakan dari prosedur pencarian atau optimasi yang lain, yaitu (Siahaan, 2012):

- Algoritma genetika dengan pengkodean dari himpunan solusi permasalahan berdasarkan parameter yang telah diterapkan dengan bukan parameter itu sendiri.
- Algoritma genetika pencarian pada sebuah solusi dari sejumlah individu-individu yang merupakan solusi permasalahan bukan hanya dari sebuah individu.

- Algoritma genetika informasi fungsi objektif (*fitness*), sebagai cara untuk mengevaluasi individu yang mempunyai solusi terbaik, bukan turunan dari suatu fungsi.
- Algoritma genetika menggunakan aturan-aturan transisi peluang, bukan aturan deterministik.

Algoritma genetika dapat menyelesaikan beberapa permasalahan yang mempunyai ciri-ciri seperti berikut (Achmad, 2003):

- Mempunyai fungsi tujuan optimalisasi non linear dengan banyak kendala yang juga non linear.
- Mempunyai kemungkinan solusi yang jumlahnya tak berhingga.
- Membutuhkan solusi "*real-time*" dalam arti solusi bisa didapatkan dengan cepat sehingga dapat diimplementasikan untuk permasalahan yang mempunyai perubahan yang cepat seperti optimasi pada pembebanan kanal pada komunikasi seluler.
- Mempunyai *multi-objective* dan *multi-criteria*, sehingga diperlukan solusi yang dapat secara bijak diterima oleh semua pihak.

Struktur umum dari suatu algoritma genetika dapat didefinisikan dengan langkah-langkah sebagai berikut:

- Membangkitkan populasi awal

  Populasi awal ini dibangkitkan secara random sehingga didapatkan solusi awal. Populasi itu sendiri terdiri dari sejumlah kromosom yang merepresentasikan solusi yang diinginkan.
- Membentuk generasi baru

  Dalam membentuk digunakan tiga operator yang telah disebut di atas yaitu operator reproduksi/seleksi, *crossover* dan mutasi. Proses ini dilakukan berulang-ulang sehingga didapatkan jumlah kromosom yang cukup untuk membentuk generasi baru dimana generasi baru ini merupakan representasi dari solusi baru.
- Evaluasi solusi

  Proses ini akan mengevaluasi setiap populasi dengan menghitung nilai fitness setiap kromosom dan mengevaluasinya sampai terpenuhi kriteria

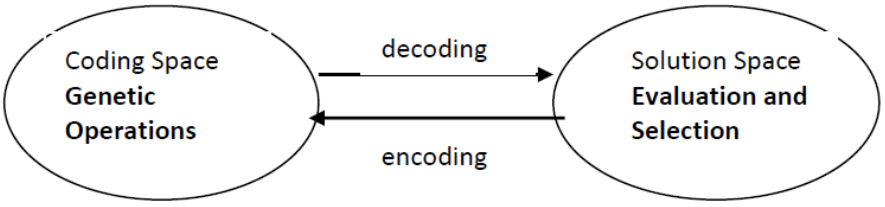
berhenti. Bila kriteria berhenti belum terpenuhi maka akan dibentuk lagi generasi baru dengan mengulangi langkah 2. Beberapa kriteria berhenti yang sering digunakan antara lain:

- Berhenti pada generasi tertentu.
- Berhenti setelah dalam beberapa generasi berturut-turut didapatkan nilai *fitness* tertinggi tidak berubah.
- Berhenti bila dalam n generasi berikut tidak didapatkan nilai fitness yang lebih tinggi.

Algoritma genetika memiliki komponen-komponen utama yang penting, yaitu:

- Teknik *Encoding*

Proses encoding adalah salah satu proses yang sulit dalam algoritma genetika. Hal ini disebabkan karena proses *encoding* untuk setiap permasalahan berbeda-beda karena tidak semua teknik *encoding* cocok untuk setiap permasalahan (Lukas, 2005). Dalam algoritma genetika, fungsi enkoding berguna untuk memetakan objek variabel kedalam kode *string*, dan memetakan kode *string* kedalam objek *variabel* dilakukan oleh proses *decoding* seperti pada Gambar 2.1.



Gambar 2.1. Proses enkoding dan dekoding

Berdasarkan struktur encoding, dapat diklasifikasikan kedalam 1-dimensi dan 2-dimensi. *Encoding* biner, oktal, heksadesimal, permutasi, dan *value encoding* adalah jenis 1-dimensi, sementara encoding pohon merupakan jenis 2-dimensi (Gen, 1996).

Menurut Goldbreg, fungsi *fitness* untuk skema *encoding* tertentu tergantung pada dua faktor yaitu nilai dan order. *Encoding* dapat

diklasifikasikan berdasarkan faktor evaluasi *fitness* yaitu (Goldberg, 1989):

- Skema encoding di mana *fitness* tergantung pada order saja: f (o) misalnya encoding permutasi.
- Skema encoding di mana *fitness* tergantung pada nilai dan order: f (v, o) misalnya encoding biner.
- Skema encoding di mana *fitness* tergantung hanya pada nilai saja: f (v) misalnya encoding nilai.

Jenis jenis enkoding sendiri akan dipaparkan sebagai berikut (Kumar, 2013):

- Enkoding biner, merupakan bentuk paling umum dari *encoding*. Dalam pengkodean ini setiap kromosom diwakili menggunakan *string* biner. Dalam pengkodean biner setiap kromosom adalah *string* bit, 0 atau 1 (Siahaan, 2019). Gambar 2.2 menunjukkan pengkodean binari.

Chromosome1	110101110010
Chromosome2	011010011101

Gambar 2.2. Pengkodean binari

Dalam pengkodean ini setiap bit menunjukkan beberapa karakteristik solusi. Di sisi lain setiap string biner mewakili nilai. Dengan jumlah yang lebih kecil dari alel, sejumlah kromosom dapat direpresentasikan. *Crossover* operasi mungkin dalam pengkodean biner adalah 1-point *crossover*, N-point *crossover*, *crossover* yang Seragam dan *Arithmetic crossover*. Operator Mutasi yang mungkin adalah flip. Dalam mutasi flip, bit perubahan dari 0 ke 1 dan 1 menjadi 0 berdasarkan mutasi kromosom yang dihasilkan. Hal ini umumnya digunakan dalam masalah *Knapsack* mana pengkodean biner digunakan untuk menunjukkan adanya item mengatakan 1 untuk menunjukkan keberadaan item dan 0 tidak adanya item.

- Enkoding oktal, dalam pengkodean ini kromosom direpresentasikan dengan menggunakan bilangan oktal(0-7). Gambar 2.3 menunjukan enkoding oktal

Chromosome1	06254524
Chromosome2	63726425

Gambar 2.3. Pengkodean Oktal

- Enkoding heksadesimal, dalam pengkodean ini kromosom direpresentasikan dengan menggunakan bilangan heksa(0-9, A-F). Gambar 2.4. menunjukan enkoding heksadesimal

Chromosome1	97AE
Chromosome2	A2C6

Gambar 2.4. Pengkodean Heksa

- Enkoding permutasi, digunakan dalam masalah pengorderan. Dalam hal ini, setiap kromosom merupakan posisi dalam urutan misalnya dalam *traveling salesman problem*, string angka mewakili urutan kota yang dikunjungi oleh salesman. Kadang-kadang koreksi harus dilakukan setelah operasi genetik selesai. Gambar 2.5. menunjukkan enkoding permutasi.

Chromosome1	1 5 2 3 5 2 6 4 6 9 8
Chromosome2	8 6 3 6 3 9 6 3 1 5 8

Gambar 2.5. Pengkodean Permutasi

Permutasi encoding hanya berguna untuk masalah yang memiliki urutan tertentu. Untuk jenis seperti masalah ini beberapa jenis *crossover* dan mutasi koreksi harus dilakukan untuk meninggalkan kromosom konsisten (yaitu, memiliki urutan nyata di dalamnya). Operator *Crossover* yang dapat dilakukan pada enkoding permutasi ini adalah *Partially Mapped Crossover* (PMX), *Cycle Crossover* (OCX) dan *Order*

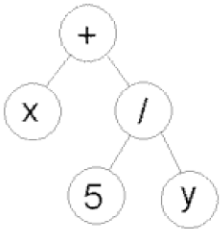
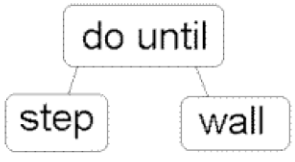
Crossover (OX). Inversi adalah operator mutasi yang paling umum digunakan dan diterapkan pada kromosom order ini. Operator ini merubah lokasi dari karakter.

- Enkoding Nilai, dalam enkoding nilai, setiap kromosom direpresentasikan sebagai string dari beberapa nilai. Nilai bisa berupa integer, bilangan real, karakter atau objek. Dalam kasus nilai integer, operator *crossover* yang digunakan adalah sama dengan yang diterapkan pada pengkodean biner. Nilai dapat berupa apapun yang berhubungan dengan masalah, nomor form, bilangan real atau karakter untuk beberapa objek yang rumit. Gambar 2.6. menunjukkan enkoding nilai.

Chromosome1	1.23, 2.12, 3.14, 0.34, 4.62
Chromosome2	ABDJEIFJDHDDLDFLFEGT

Gambar 2.6. Pengkodean nilai

- Enkoding pohon, digunakan untuk *evolving programs* atau ekspresi untuk pemrograman genetik. Dalam pengkodean pohon setiap kromosom adalah pohon dari beberapa obyek, seperti fungsi atau perintah dalam bahasa pemrograman. Misalnya representasi dari sebuah ekspresi dalam bentuk pohon biner seperti yang ditunjukkan pada gambar 2.7.

Chromosome 1	Chromosome 2
	
(+ x (/ 5 y))	(do until step wall)

Gambar 2.7. Pengkodean pohon

Enkoding pohon baik untuk *evolving programs*. LISP berguna dalam pengkodean ini karena membantu dalam membangun

pohon untuk parsing dan karenanya *crossover* dan mutasi dapat dilakukan dengan mudah. Kromosom adalah fungsi diwakili di dalam pohon.

- Prosedur inisialisasi

  Membangkitkan populasi awal adalah membangkitkan sejumlah individu secara acak atau melalui prosedur tertentu. Ukuran populasi tergantung pada masalah yang akan dipecahkan dan jenis operator genetika yang akan diimplementasikan. Setelah ukuran populasi ditentukan, kemudian harus dilakukan inisialisasi terhadap kromosom yang terdapat pada populasi tersebut. Inisialisasi kromosom dilakukan secara acak, namun demikian harus tetap memperhatikan domain solusi dan kendala permasalahan yang ada.

  Teknik dalam membangkitkan populasi awal ini ada beberapa macam, diantaranya adalah sebagai berikut:

  1) Random *Generator*

     Inti dari cara ini adalah memberikan pembangkitan bilangan random untuk nilai setiap gen sesuai dengan representasi kromosom yang digunakan. Gen hantinya berisi pembulatan dari bilangan random yang dibangkitkan sebanyak Nipop (jumlah populasi) x Nbits (jumlah gen dalam tiap kromosom)

  2) Pendekatan tertentu (memasukan nilai tertentu kedalam gen)

     Cara ini adalah dengan memasukan nilai tertentu kedalam gen dari populasi awal yang dibentuk.

  3) Permutasi gen

     Salah satu cara dari pembangkitan populasi awal dengan permutasi gen adalah penggunaan permutasi Josephus dalam permasalahan kombinatorial seperti *travelling salesmen problem* (TSP).

- Proses evaluasi nilai fitness

  Evaluasi fitness merupakan dasar untuk proses seleksi. Ada tiga langkah dalam proses mengevaluasi nilai fitness kromosom, yaitu:

1. Mengganti genotip kromosom menjadi fenotip kromosom, ini berarti mengganti binary strings menjadi *real value*
2. Mengevaluasi fungsi objektif
3. Mengganti nilai dari fungsi objektif menjadi nilai *fitness*. Agar nilai fitness selalu bernilai positif, maka nilai fitness dari setiap kromosom sama dengan memaksimumkan objektif dikurangi objektif yang telah dievaluasi untuk setiap kromosom dalam populasi.

Suatu individu dievaluasi berdasarkan suatu fungsi tertentu sebagai ukuran performansinya. Di dalam evolusi alam, individu yang bernilai *fitness* tinggi yang akan bertahan hidup, sedangkan individu yang bernilai *fitness* rendah akan mati.

- Proses seleksi

Pemilihan dua buah kromosom yang dijadikan induk atau sebagai orang tua dilakukan secara proporsional sesuai dengan dengan nilai fitness-nya. Masing-masing individu dalam suatu wadah seleksi akan menerima probabilitas reproduksi yang tergantung dari nilai objektif dirinya sendiri terhadap nilai objektif dari semua individu dalam wadah seleksi tersebut. Nilai fitness inilah yang nantinya akan digunakan pada tahap seleksi berikutnya.

Terdapat beberapa metode seleksi orang tua, antara lain sebagai berikut:

1. *Rank-based fitness assignment*

   Pada *Rank-based fitness*, populasi diurutkan menurut nilai objektifnya. Nilai fitness dari tiap-tiap individu hanya tergantung pada posisi individu tersebut dalam urutan, dan tidak dipengaruhi oleh nilai objektifnya.

2. *Roulette wheel selection*

   Metode seleksi roda roulette ini merupakan metode yang paling sederhana serta paling banyak digunakan, dan sering juga dikenal dengan nama *stochastic sampling with replacement*. Pada metode ini, individu-individu dipetakan dalam suatu segmen garis secara

beraturan sedemikian hingga tiap-tiap segmen individu memiliki ukuran yang sama dengan dengan ukuran fitnessnya. Sebuah bilangan random akan dibangkitkan dan individu yang memiliki segmen dalam kawasan bilangan random tersebut akan diseleksi. Proses ini diulang hingga diperoleh sejumlah individu yang diharapkan.

Skema dengan seleksi roda roulette ini adalah berdasarkan *fitness scale* (skala fitness). Terpilihnya suatu kromosom dalam populasi untuk dapat berkembang biak sebanding dengan *fitness*-nya. *Tradeoff* antara eksplorasi dan eksploitasi terjadi jika terdapat satu atau sekelompok kecil kromosom yang mempunyai fitness yang baik, yaitu mengaksplorasi bagian-bagian baru dalam ruang pencarian atau terus mengeksploitasi informasi yang telah diperoleh. Kecenderungan kromosom yang baik untuk terpelihara terus dapat membawa ke hasil optimum lokal atau konvergensi dini (*premature convergence*) ke suatu hasil yang bukan optimum global. Namun demikian, jika semua kromosom dalam populasi mempunyai fitness yang hampir sama maka seleksi ini akan menjadi seleksi yang bersifat acak.

3. *Stochastic universal sampling*

   Stochastic universal sampling memiliki nilai bias nol dan penyebaran yang minimum. Pada metode ini, individu-individu dipetakan dalam suatu segmen garis secara berurutan sedemikian hingga tiap-tiap segmen individu memiliki ukuran yang sama dengan ukuran *fitness*-nya seperti halnya pada seleksi roda *roulette*, dan diberikan sejumlah pointer sebanyak individu yang diseleksi di garis tersebut. Andaikan N adalah jumlah individu, dan posisi pointer pertama diberikan secara acak pada range [1, 1/N].

4. Seleksi lokal (*Local selection*)

   Pada seleksi lokal setiap individu yang berada di dalam constraint tertentu disebut dengan nama lingkungan lokal. Interaksi antar

individu hanya dilakukan dalam wilayah tersebut. Lingkungan tersebut ditetapkan sebagai struktur dimana populasi tersebut terdistribusi. Lingkungan tersebut juga dipandang sebagai sekelompok pasangan-pasangan yang potensial. Langkah pertama yang harus dilakukan adalah menyeleksi separuh pertama dari populasi yang berpasangan secara random, kemudian lingkungan baru tersebut diberikan pada setiap individu yang terseleksi. Jarak antara individu dengan struktur tersebut akan sangat menentukan ukuran lingkungan. Individu yang terdapat dalam lingkungan dengan ukuran yang lebih kecil, akan lebih terisolasi dibandingkan dengan individu yang terletak pada lingkungan dengan ukuran yang lebih besar.

5. Seleksi dengan pemotongan (*Truncation selection*)

   Seleksi dengan pemotongan ini lebih berkesan sebagai seleksi buatan dan biasanya digunakan oleh populasi yang jumlahnya sangat besar. Pada metode ini, individu-individu yang terbaik saja yang akan diseleksi sebagai induk. Parameter yang digunakan dalam metode ini adalah suatu nilai ambang trunc yang mengindikasikan ukuran populasi yang akan diseleksi sebagai induk yang berkisar antara 50% -10%. Individu-individu yang ada di bawah nilai ambang ini tidak akan menghasilkan keturunan.

6. Seleksi dengan turnamen (*Tournament selection*)

   Pada metode seleksi dengan turnamen ini akan ditetapkan suatu nilai tour untuk individu-individu yang dipilh secara acak (random) dari suatu populasi. Individu-indiidu yang terbaik dalam kelompok ini akan diseleksi sebagai induk. Parameter yang digunakan pada metode ini adalah ukuran tour yang bernilai antara 2 sampai N (jumlah individu dalam suatu populasi).

   Dari berbagai jenis seleksi tersebut, Umumnya jenis seleksi pada roda roulette paling sering digunakan, terkadang juga metode rangking dan turnamen. Yang perlu diperhatikan dalam seleksi

adalah prinsip elitism, yang dilakukan dalam sekali seleksi untuk update generasi, biasanya digunakan *steady-state update*. Jadi tujuan utama dari elitism ini adalah untuk menjaga agar individu-individu yang bernilai *fitness* tertinggi tidak hilang selama proses evolusi, maka perlu dibuat kopiannya.

- Proses Rekombinasi

Algoritma genetika merupakan proses pencarian yang heuristic dan acak sehingga penekanan pemilihan operator yang digunakan sangat menentukan keberhasilan algoritma genetika dalam menemukan solusi optimum suatu masalah yang diberikan. Hal yang harus diperhatikan adalah menghindari terjadinya konvergensi prematur, dimana dicapai solusi optimum yang belum waktunya, dalam arti bahwa solusi yang diperoleh adalah hasil optimum lokal.

Terdapat dua operator genetika untuk melakukan rekombinasi, yaitu:

1. Rekombinasi bernilai real

Terdapat beberapa metode dalam rekombinasi bernilai real, yaitu:

(i) Rekombinasi diskrit

Rekombinasi diskrit akan menukar nilai variabel antar kromosom induk. Misalkan ada 2 individu dengan 3 variabel, yaitu:

Induk 1 : 12 25 5

Induk 2 : 123 4 34

Untuk tiap-tiap variabel induk yang menyumbangkan variabelnya ke anak yang dipilih secara random dengan probabilitas yang sama

sampel 1 : 2 2 1

sampel 2 : 1 2 1

Setelah rekombinasi, kromosom-kromosom baru yang terbentuk yaitu:

Anak 1 : 123 4 5

Anak 2 : 12 4 5

Rekombinasi diskrit dapat digunakan untuk sembarang variabel (biner, real, atau simbol).

(ii) Rekombinasi intermediate (menengah)

Rekombinasi intermediate hanya dapat digunakan untuk variabel real (dan variabel yang bukan biner).

Anak dihasilkan menurut aturan sebagai berikut :

Anak = induk 1 + alpha (induk 2 –induk 1)

Dengan alpha adalah faktor skala yang dipilih secara random pada interval [-d, 1+d], biasanya d=0,25. Tiap-tiap variabel pada anak merupakan hasil kombinasi variabel-variabel menurut aturan di atas dengan nilai alpha dipilih ulang untuk tiap variabel. Misalkan ada 2 individu dengan 3 variabel, yaitu:

Induk 1 : 12 25 5

Induk 2 : 123 4 34

Misalkan nilai alpha yang terpilih :

sampel 1 : 0,5 1,1 -0,1

sampel 2 : 0,1 0,8 0,5

Setelah rekombinasi, kromosom-kromosom baru yang terbentuk adalah:

Anak 1 : 67,5 1,9 2,1

Anak 2 : 23,1 8,2 19,5

(iii) Rekombinasi Garis

Pada dasarnya rekombinasi garis ini hampir sama dengan rekombinasi menengah, hanya saja nilai alpha untuk semua variabel adalah sama. Misalkan ada 2 kromosom dengan 3 variabel:

induk1 : 12 25 5

induk2 : 123 4 34

untuk tiap-tiap variabel induk yang menyumbangkan variabelnya ke anak dipilih secara random dengan probabilitas yang sama

sample 1: 0,5

sample 2: 0,1

setelah rekombinasi kromosom-kromosom baru yang terbentuk adalah:

anak1: 67,5 14,5 19,5

anak2: 23,1 22,9 7,9

2. Rekombinasi bernilai biner atau penyilangan (*Crossover*)

Crossover melibatkan dua induk untuk membentuk kromosom baru. Pindah silang menghasilkan titik baru dalam ruang pencarian untuk siap diuji. Proses *crossover* dilakukan pada setiap individu dengan probabilitas *crossover* (Pc) yang ditentukan secara acak dalam rentang (0,1).

Terdapat beberapa metode *crossover*, yaitu:

1. Penyilangan satu titik (*single point Crossover*)

Pada penyilangan satu titik, posisi penyilangan k (k=1,2,…,N-1) dengan panjang kromosom (N) diseleksi secara random. Variabel-variabel ditukar antar kromosom pada titik tersebut untuk menghasilkan anak. Misalkan ada 2 kromosom dengan panjang 12:

Induk 1: 0 1 1 1 0 | 0 1 0 1 1 1 0

Induk 2: 1 1 0 1 0 | 0 0 0 1 1 0 1

Posisi menyilang yang terpilih acak: misalkan setelah bit ke-5. Setelah dilakukan penyilangan, diperoleh kromosom-kromosom

baru:

Anak 1: 0 1 1 1 0 | 0 0 0 1 1 0 1

Anak 2: 1 1 0 1 0 | 0 1 0 1 1 1 0

2. Penyilangan dua titik (*two-point Crossover*)

Penyilangan ini menentukan dua titik secara acak sebagai batas untuk menukar 2 kromosom induk yang berada diantaranya untuk menghasilkan 2 individu yang baru. Misalkan ada 2 kromosom dengan panjang kromosom 10

Induk 1: 110 | 000 | 1100

Induk 2: 100 | 100 | 1011

Posisi menyilang yang terpilih acak: misalkan setelah bit ke-3 dan ke-6, maka setelah dilakukan penyilangan diperoleh kromosom baru:

Anak 1: 110 | 100 | 1100

Anak 2: 100 | 000 | 1011

3. Penyilangan banyak titik (*multi-point Crossover*)

Pada penyilangan ini, jumlah titik posisi penyilangan, (k=1,2,…,N-1,i=1,2,…,m) dengan panjang kromosom (N) diseleksi secara random dan tidak diperbolehkan ada posisi yang sama, serta diurutkan naik. Variabel-variabel ditukar antar kromosom pada titik tersebut untuk menghasilkan anak. Misalkan ada 2 kromosom dengan panjang 12:

Induk 1: 011100101110

Induk 2: 110100001101

Posisi penyilangan yang terpilih adalah setelah bit ke- 2, 6, dan 10. Setelah penyilangan, diperoleh kromosom-kromosom baru:

anak 1: 01 | 0100 | 1011 | 01

anak 2: 11 | 1100 | 0010 | 10

4. Penyilangan seragam (*uniform Crossover*)

Pada penyilangan seragam, setiap lokasi memiliki potensi sebagai tempat penyilangan. Sebuah mask penyilangan dibuat sepanjang panjang kromosom secara random yang menunjukan bit-bit dalam mask yang mana induk akan mensuplai anak dengan bit-bit yang ada. Induk mana yang akan menyumbangkan bit ke anak yang dipilih secara random dengan probabilitas yang sama. Misalkan ada 2 kromosom dengan panjang 12:

Induk 1 : 011100101110

Induk 2 : 110100001101

Mask bit :

Sampel 1 : 100111001101

Sampel 2 : 011000110010

Setelah penyilangan, diperoleh kromosom-kromosom baru :

anak 1 : 010100001100

anak 2 : 111100101111

5. *Partially Matched Crossover* (PMX)

Dalam *Partially Matched Crossover* dua titik *crossover* dipilih secara acak dari kromosom parentis untuk menghasilkan keturunan. Dua titik *crossover* memberikan pilihan pencocokan yang digunakan untuk mempengaruhi lintas melalui posisi dengan operasi pertukaran posisi. Sebagai contoh:

Induk 1 : (1 2 3 | 4 5 6 7 | 8 9)

Induk 2 : (4 5 3 | 1 8 7 6 | 9 2)

Anak 1 : (x x x | 1 8 7 6 | x x)

Anak 2 : (x x x | 4 5 6 7 | x x)

Dari sini diperoleh hasil pemetaan :

1-4, 8-5, 7-6, 6-7.

Kemudian copy sisa gen di induk-1 ke anak-1 dengan menggunakan pemetaan yang sudah ada.

Anak 1 : (1-4 2 3 | 1 8 7 6 | 8-5 9)

Anak 2 : (4 2 3 | 1 8 7 6 | 5 9)

Lakukan hal yang sama untuk anak-2

Anak 1 : (4-1 5-8 3 | 4 5 6 7 | 9 2)

Anak 2 : (1 8 3 | 4 5 6 7 | 9 2)

6. *Ordered Crossover* (OX)

Dalam *Ordered Crossover* operator memerintahkan dua titik potong yang dipilih secara acak dari kromosom parentis. Disini untuk menghasilkan keturunan O1 gen antara titik-titik perpotongan digantikan oleh gen di induk kedua. Sebagai contoh:

Induk 1 = (1 2 3 | 4 5 6 7 | 8 9)

Induk 2 = (4 5 2 | 1 8 7 6 | 9 3)

Anak 1 = (x xx| 1 8 7 6 | x x)

Anak 2 = (x x x | 4 5 6 7 | x x)

Urutkan gen dari Induk 2 (dimulai dari titik potong ke dua)

9 -3 -4 -5 -2 -1 -8 -7 -6

Hilangkan gen yang sudah ada di dalam anak,

9 -3 -2 -1 -8

Hasil urutan gen diatas dimasukan ke dalam anak (dimulai dari titik potong kedua)

Anak 1 = (2 1 8 | 4 5 6 7 | 9 3)

Anak 2 = (3 4 5 | 1 8 7 6 | 9 2)

7. *Cyclic Crossover* (CX)

Operator *Cyclic Crossover* melakukan rekombinasi di bawah kendala bahwa setiap gen berasal dari orang tua atau lainnya. Sebagai contoh:

Induk 1 = (1 2 3 4 5 6 7 8 9)

Induk 2 = (4 1 2 8 7 6 9 3 5)

Anak 1 gen pertama berasal dari gen pertama Induk 1

Anak 1 = (1 x x x x x x x x)

Gen berikutnya adalah gen dibawahnya gen 1 dari induk 1, yaitu gen 4, begitulah selanjutnya,

Anak 1 = (1 x x 4 x x x x x)

Anak 1 = (1 x x 4 x xx8 x)

Anak 1 = (1 x 3 4 x xx8 x)

Anak 1 = (1 2 3 4 x xx8 x)

Pemilihan selanjutnya dari gen 2 seharusnya memilih gen 1 yang sudah ada dalam urutan, sehingga sisa gen diisi dari induk 2. Proses yang sama dilakukan untuk anak 2

Induk 1 = (1 2 3 4 5 6 7 8 9)

Induk 2 = (4 1 2 8 7 6 9 3 5)

Anak 1 = (1 2 3 4 7 6 9 8 5)

Anak 2 = (4 x x x x x x x x)

...

Anak 2 = (4 1 2 8 5 6 7 3 9)

- Proses Mutasi

Mutasi merupakan proses untuk mengubah nilai dari satu atau beberapa gen dalam suatu kromosom. Operasi *crossover* yang dilakukan pada kromosom dengan tujuan untuk memperoleh kromosom-kromosom baru sebagai kandidat solusi pada generasi mendatang dengan fitness yang lebih baik, dan lama-kelamaan menuju solusi optimum yang diinginkan. Akan tetapi, untuk mencapai hal ini, penekanan selektif juga memegang peranan yang penting. Jika dalam proses pemilihan kromosom-kromosom cenderung terus pada kromosom yang memiliki fitness yang tinggi saja, konvergensi prematur akan sangat mudah terjadi.

Proses Mutasi

Terdapat beberapa jenis mutasi, yaitu:

1. Mutasi dalam pengkodean Biner

   Mutasi pada pengkodean biner merupakan operasi yang sangat sederhana. Proses yang dilakukan adalah menginversi nilai bit pada posisi tertentu yang dipilih secara acak (atau menggunakan skema tertentu) pada kromosom, yang disebut dengan inverse bit.

2. Mutasi dalam pengkodean Permutasi

   Proses mutasi yang dilakukan dalam pengkodean biner dengan mengubah langsung bit-bit pada kromosom tidak dapat dilakukan pada pengkodean permutasi karena konsistensi urutan permutasi harus diperhatikan. Salah satu cara yang dapat dilakukan adalah dengan memilih dua posisi (locus) dari kromosom dan kemudian nilainya saling dipertukarkan.

3. Mutasi dalam pengkodean nilai

   Pada pengkodean nilai hampir sama dengan yang dilakukan pada pengkodean biner, tetapi yang dilakukan bukan menginversikan nilai bit, serta penerapannya tergantung pada jenis nilai yang akan

digunakan. Sebagai contoh, untuk nilai riil proses mutasi dapat dilakukan seperti yang dilakukan pada pengkodean permutasi, dengan saling mempertukarkan nilai dua gen pada kromosom. Namun demikian, cara ini tidak menjamin adanya perbedaan pada populasi sehingga semua kromosom dapat dengan mudah mempunyai nilai yang sama, dan justru mempercepat terjadinya konvergensi prematur. Cara lain yang lebih baik adalah dengan memilih sembarang posisi gen pada kromosom. Nilai yang ada tersebut kemudian ditambahkan atau dikurangkan dengan suatu nilai kecil tertentu yang diambil secara acak. Cara ini juga berlaku untuk pengkodean dengan bilangan bulat (cara mutasi lain yang relevan juga dapat digunakan).

4. Mutasi dalam pengkodean pohon

   Dalam metode ini dapat dilakukan dengan cara mengubah operator (+, -, *, /) atau nilai yang terkandung pada suatu vertex pohon yang dipilih, atau dengan memilih dua vertex pohon dan saling mempertukarkan operator atau lainnya

- *Elitism*

  Proses seleksi dilakukan secara random sehingga tidak ada jaminan bahwa suatu indvidu yang bernilai *fitness* tertinggi akan selalu terpilih. Walaupun individu bernilai fitness tertinggi terpilih, mungkin saja individu tersebut akan rusak (nilai *fitness*nya menurun) karena proses pindah silang. Oleh karena itu, untuk menjaga agar individu bernilai *fitness* tertinggi tersebut tidak hilang selama evolusi, maka perlu dibuat satu atau beberapa kopinya. Prosedur ini dikenal sebagai elitisme.

2.2 Penelitian Terkait

2.2.1 *Formulation of Genetic Algorithm to Generate Good Quality Course Timetable*

Dimuat dalam *International Journal of Innovation, Management and Technology*, Vol. 1, No. 3, terbit agustus 2010, ISSN: 2010-0248, ditulis oleh Ashish Jain, Dr. Suresh Jain dan Dr. P.K. Chande

Tujuan dari penelitian ini adalah membuat sebuah formulasi algoritma genetika untuk membuat jadwal kuliah dengan kualitas baik. Untuk operator genetikanya sendiri metode seleksi yang digunakan adalah roulette wheel dan metode *crossover* yang digunakan adalah one site *crossover*. Penelitian ini menggunakan data dari jurusan *M.B.A Truba College of Engineering & Technology, Devi ahilya University*, tahun ajar 2009-2010, dengan 9 dosen yang tersedia.

Berdasarkan hasil yang didapat, berbagai operator genetik yang dilakukan seperti seleksi, mutasi dan *crossover* dapat memperbaiki hasil dalam berbagai aspek. Penelitian ini menunjukkan bahwa algoritma genetika adalah metode yang kuat untuk memecahkan masalah penjadwalan, namun penelitian ini belum mencangkup permasalahan penjadwalan dalam *real-word* karena batasan yang digunakan terlalu sedikit.

2.2.2 *University Timetabling based on Hard Constraints using Genetic Algorithm*

Merupakan jurnal dalam *International Journal of Computer Application (0975-8887)* Vol. 42 No. 15, Maret 2012 ditulis oleh Sanjay R. Sutar dan Rajan S. Bichkar.

Penelitian ini menyajikan sebuah pendekatan pemecahan masalah penjadwalan matakuliah dalam 8 semester dan 3 jurusan dengan berbasiskan batasan kaku yang digambarkan sebagai masalah *NP hard*. Pendekatan algoritma genetika ini digunakan untuk memperkirakan hasil terbaik dalam *dataset realtime* dari jadwal *Dr. Babasaheb Ambedkar Technological University, Maharashtra, India*.

Dari hasil yang didapatkan telah diketahui bahwa optimalisasi 100% dari sumber daya yang tersedia tidak *feasible* dalam kasus apapun. Sebagai contoh penelitian ini tidak dapat mengoptimalkan dosen untuk melakukan sesi dalam jumlah maksimum dalam satu minggu *timeslots*.

2.2.3 *Genetic Algorithm Approach to Automate University Timetable*

Merupakan jurnal dalam *International Journal of Technical Research (IJTR)* Vol 1, Issue 1, Maret-April 2012 ditulis oleh Kuldeep Kumar, Sikander, Ramesh Sharma, dan Kaushal Mehta dari *Dept. of Computer Science, CDLU, Sirsa National Informatics Centre, Mini Secttt, Fatehabad Dept. of Computer Sci.*

Penelitian ini menyajikan investigasi penggunaan algoritma genetika untuk menyelesaikan masalah penjadwalan universitas. Penjadwalan dalam penelitian ini memiliki beberapa batasan yang dibagi menjadi dua kategori yaitu batasan kaku (*hard constraints*) dan batasan lunak (*soft constraints*). Proses *crossover* pada penelitian ini menggunakan single point *crossover*. Sedangkan proses mutasi dilakukan setelah proses *crossover* dilakukan dan sebelum anakan dilepaskan ke populasi.

Pendekatan algoritma genetika ini diimplementasikan menggunakan *GALIB 247* pada prosesor *pentium core 2 duo* dengan *ubuntu* 9.10 sebagai sistem operasinya. Dengan menggunakan dataset yang berdasarkan data asli dari *GJUS & T, Hisar*. Setiap jadwal dievaluasi dengan menguji bebepara kali pada setiap batasan-batasan. Kemudian jadwal dengan jumlah minimum pelanggaran batasan berevolusi.

Hasil dari penelitian ini mendapatkan bahwa pada generasi ke 250 solusi jadwal yang didapatkan masih melanggar beberapa batasan kaku dan batasan lunak, namun jumlah pelanggaran batasan kaku dan batasa lunak semakin berkurang sejalan dengan bertambahnya jumlah generasi.

2.2.4 *Timetabling Construction Problem (TCP)*

Merupakan jurnal dalam ITSMART Vol 1, No 1, Juni 2012 ditulis oleh Palgunadi dari Fakultas MIPA UNS.

Penelitian ini mengusulkan algoritma polinimial baru (Algoritma Palgunadi) untuk menyelesaikan masalah penjadwalan secara umum. Penjadwalan disini

merupakan sebuah tabel dari perkuliahan diampu oleh dosen yang disusun sesuai dengan ruangan-waktu dimana perkuliahan diadakan, penyusunan ruang-waktu ini tentunya harus memenuhi sejumlah kebutuhan dan batasan. Penelitian ini menyimpulkan bahwa TCP dapat diselesaikan dalam waktu polinimial. Namun pada kenyataannya algoritma ini belum diimplementasikan.

2.3 Rencana Penelitian

Berdasarkan tinjauan pustaka yang ada, algoritma gentika sukses diterapkan pada permasalahan penjadwalan, sedangkan algoritma Palgunadi belum diterapkan seperti halnya algoritma genetika. Algoritma Palgunadi yang bekerja pada pencarian satu-satu memiliki kelemahan tidak terjaminnya hasil yang 100% optimal, sedangkan Algoritma Genetika memiliki kemungkinan terdapat hasil yang 100% optimal namun karena ruang cari yang sangat luas proses algortima genetika menjadi sangat lama. Kebaharuan dari penelitian ini adalah pengkombinasian algoritma genetika dengan algoritma palgunadi dengan memanfaatkan pencarian yang eksploratif dari algoritma genetika dikombinasikan dengan Algoritma Palgunadi untuk mengoptimalkan kromosom dengan memastikan tidak ada batasan kaku dilanggar pada setiap proses algoritma genetika. Perbandingan beberapa penelitian diatas dirangkum dalam Tabel 2.2.

Tabel 2.2 Perbandingan Penelitian

No	Metode	Kasus	Hasil
1	Algoritma Genetika (Jain, 2010)	*College of Engineering & Technology, Devi ahilya University*	Algoritma genetika efektif untuk menjadwalkan. Namun batasan yang diambil masih sedikit dan jauh dari kondisi *real-world*
2	Algoritma Genetika (Sutar, 2012)	*Dr. Babasaheb Ambedkar Technological University, Maharashtra, India*	Algoritma berhasil menjadwalkan. Optimalisasi 100% dari sumberdaya tidak feasible.

No	Metode	Kasus	Hasil
3	Algoritma Genetika (Kumar, 2012)	*GJUS & T, Hisar*	Pelanggaran batasan berkurang pada setiap generasi, namun dalam generasi ke 250 masih terdapat pelanggaran batasan kaku dan batasan lunak.
4	Algoritma Palgunadi (Palgunadi, 2012)	(belum diimplementasi)	(belum diimplementasi)