

**IMPLEMENTASI ALGORITMA GENETIKA UNTUK
OTOMATISASI PENJADWALAN SEKOLAH
MENENGAH PERTAMA**

SKRIPSI

**Oleh:
Muhammad Fadhil Al Amal
NIM. 15650035**



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

**IMPLEMENTASI ALGORITMA GENETIKA UNTUK
OTOMATISASI PENJADWALAN SEKOLAH
MENENGAH PERTAMA**

SKRIPSI

**Diajukan kepada:
Fakultas Sains dan Teknologi
Universitas Islam Negeri (UIN) Maulana Malik Ibrahim Malang
Untuk Memenuhi Salah Satu Persyaratan Dalam
Memperoleh Gelar Sarjana Komputer (S.Kom)**

**Oleh :
MUHAMMAD FADHIL AL AMAL
NIM. 15650035**

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

LEMBAR PERSETUJUAN**IMPLEMENTASI ALGORITMA GENETIKA UNTUK OTOMATISASI
PENJADWALAN SEKOLAH MENENGAH PERTAMA****SKRIPSI**

Oleh:
MUHAMMAD FADHIL AL AMAL
NIM. 15650035

Telah Diperiksa dan Disetujui untuk Diuji

Tanggal : 2020

Dosen Pembimbing I

Dosen Pembimbing II

Fatchurrochman, M.Kom
NIP. 19700731 200501 1 002

Ainatul Mardhiyah, M.Cs
NIDT. 19860330 20160801 2 075

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

Dr. Cahyo Crysdian

NIP. 19740424 200901 1 008

LEMBAR PENGESAHAN
IMPLEMENTASI ALGORITMA GENETIKA UNTUK OTOMATISASI
PENJADWALAN SEKOLAH MENENGAH PERTAMA

SKRIPSI

Oleh:
MUHAMMAD FADHIL AL AMAL
NIM. 15650035

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

Susunan Dewan Penguji	Tanda tangan
1. Penguji Utama :	()
2. Ketua Penguji :	()
3. Sekretaris Penguji :	()
4. Anggota Penguji :	()

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

Dr. Cahyo Crysdian
NIP. 19740424 200901 1 008

HALAMAN PERSEMBAHAN

Alhamdulillah puji syukur kehadiran Allah SWT yang telah mengaruniakan rahmat serta berkatnya sehingga penulis dapat mempersembahkan karya sederhana ini kepada orang-orang yang sangat berarti.

Kepada kedua orang tua yang telah memberikan kepercayaan, percayalah ananda akan melakukan yang terbaik. Tanpa doa dan amin ayah dan mamak, ananda bukanlah siapa-siapa. Teruntuk orang tua kami di Malang Abi Imam Muslimin dan Ibu Chusnul Chaidaroh, kami percaya sambung doa yang tidak pernah terputus dari abi dan ibuk selalu meringankan langkah kami.

Teruntuk teman terbaik saya Farrah Nurmalia, Berlian Gita dan Bahrul Ulum yang ikut berkontribusi atas selesainya persembahan ini. Teruntuk keluarga besar Anshofa yang selalu menyemangati saya dengan berbagai cara masing-masing, Hana Wilda, Rizal Musthofa, Iqbal Najib, Ariafal, juga tidak lupa Mbak Siti Qomariah. Kalian sungguh luar biasa. Teruntuk teman seperjuangan seperantauan, Putri Ikrimah dan Sofie Achmad. Teruntuk teman seperjuangan sebibingan yang memaksa menyelesaikan persembahan ini segera mungkin, Pusaka, Najib dan Reza. Teruntuk teman-teman seperkopian, teman-teman seperjuangan dan yang tidak sempat kusebutkan namanya satu persatu. Tanpa inspirasi, dorongan, bantuan dan dukungan yang telah kalian berikan kepada saya, saya mungkin bukan apa-apa saat ini.

Dan untuk pak Fatchurrochman, M.Kom selaku pembimbing I Dengan penuh kesabaran, Bapak selalu membimbing saya agar menyelesaikan tanggung jawab saya dengan baik. Terima kasih telah memberikan banyak ilmu baru yang

belum pernah didapat dalam bangku perkuliahan. Teruntuk ibu Ainatul Mardhiyah, M.Cs selaku pembimbing II yang telah menunjukan seberapa tidak telitinya saya. Teruntuk bapak Fajar Rohman Hariri M.Kom dan bapak Agung Teguh Wibowo Almais, M.T selaku penelaah penelitian ini sedari awal hingga akhir, tanpa masukan dan koreksi bapak-bapak persembahkan ini tidak akan sebaik ini.

Dan untuk beberapa nama yang tidak bisa saya sebutkan namanya satu persatu yang telah memberikan dukungan, motivasi dan doa sehingga penelitian ini dapat terselesaikan.



PERNYATAAN KEASLIAN TULISAN

Saya yang bertanda tangan di bawah ini:

Nama : Muhammad Fadhil Al Amal
 NIM : 15650035
 Fakultas/Jurusan : Sains dan Teknologi/Teknik Informatika
 Judul Skripsi : Implementasi Algoritma Genetika Untuk Otomatisasi Penjadwalan Sekolah Menengah Pertama

Menyatakan dengan sebenarnya bahwa Skripsi yang saya tulis ini benar-benar merupakan hasil karya sendiri, bukan merupakan pengambilalihan data, tulisan atau pikiran orang lain yang saya akui sebagai hasil tulisan atau pikiran saya sendiri, kecuali dengan mencantumkan sumber cuplikan pada daftar pustaka.

Apabila di kemudian hari terbukti atau dapat dibuktikan Skripsi ini hasil jiplakan, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Malang, 24 Desember 2020
 Yang membuat pernyataan,



Muhammad Fadhil Al Amal
 NIM. 15650035

MOTTO

BISMILLAH.



KATA PENGANTAR

Assalamu'alaikum Wr. Wb.

Alhamdulillah, Segala puji syukur kehadirat Allah SWT atas berkah, rahmat dan karunia-Nya yang senantiasa dilimpahkan kepada penulis, sehingga bisa merampungkan skripsi dengan judul “Implementasi Algoritma Genetika Untuk Otomatisasi Penjadwalan Sekolah Menengah Pertama” sebagai syarat untuk menyelesaikan program sarjana (S1) pada jurusan Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang. Sholawat serta salam telah tercurah kepada junjungan kita Nabi Muhammad SAW.

Dalam penyusunan skripsi ini banyak hambatan serta rintangan yang penulis hadapi namun pada akhirnya dapat melaluinya berkat adanya bimbingan dan bantuan dari berbagai pihak baik secara materil, moril maupun spiritual. Untuk itu pada kesempatan ini penulis menyampaikan ucapan terimakasih kepada :

1. Prof. Dr. Abdul Haris, M.Ag selaku Rektor Universitas Islam Negeri Maulana Malik Ibrahim Malang.
2. 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 selaku dosen pembimbing I yang telah meluangkan waktu untuk membimbing, memotivasi, mengarahkan dan memberi masukan kepada penulis dalam pengerjaan skripsi ini hingga akhir.

5. Ibu Ainatul Mardhiyah, M.Cs selaku dosen pembimbing II yang senantiasa memberi masukan dan nasihat serta petunjuk dalam penyusunan skripsi ini.
6. Segenap dosen Teknik Informatika yang telah memberikan bimbingan keilmuan selama masa studi.
7. Teman-teman seperjuangan Teknik Informatika angkatan 2015 (Interface).

Berbagai kekurangan dan kesalahan mungkin pembaca temukan dalam penulisan skripsi ini, untuk itu penulis menerima segala kritik dan saran yang membangun dari pembaca sekalian. Semoga apa yang menjadi kekurangan bisa disempurnakan oleh peneliti selanjutnya dan semoga karya tulis ini bisa bermanfaat dan menginspirasi bagi kita semua. Amin.

Wassalamu'alaikum Wr. Wb.

Malang, 24 Desember 2020

Penulis

DAFTAR ISI

LEMBAR PERSETUJUAN	iii
LEMBAR PENGESAHAN	iv
HALAMAN PERSEMBAHAN	v
PERNYATAAN KEASLIAN TULISAN	vii
MOTTO	viii
KATA PENGANTAR.....	ix
DAFTAR ISI.....	x
DAFTAR GAMBAR.....	xiii
DAFTAR TABEL	xv
ABSTRAK	xvi
ABSTRACT	xvii
مستخلص البحث.....	xviii
BAB I PENDAHULUAN.....	xix
1.1 Latar Belakang	xix
1.2 Pernyataan Masalah.....	3
1.3 Tujuan Penelitian.....	3
1.4 Manfaat Penelitian.....	3
1.5 Batasan Masalah.....	3
1.6 Sistematika Penulisan.....	4
BAB II LANDASAN TEORI	6
2.1 Penjadwalan.....	6
2.2 Sekolah Menengah Pertama	8
2.3 Penjadwalan Sekolah.....	9
2.4 Algoritma Genetika	10
2.3.1 Istilah dalam Algoritma Genetika	11

2.3.2	Parameter Algoritma Genetika.....	11
2.3.3	Struktur Algoritma Genetika.....	13
BAB III PERANCANGAN DAN IMPLEMENTASI		20
3.1	Perancangan Sistem.....	20
3.2	Representasi Kromosom.....	21
3.3	Pembentukan Gen Awal	23
3.4	Evaluasi Kromosom	26
3.5	Perhitungan <i>Fitness</i>	37
3.6	Seleksi	39
3.7	<i>Crossover</i>	41
3.8	Mutasi.....	43
3.9	Skema Elitisme.....	45
3.10	Skenario Pengujian.....	45
BAB IV UJI COBA DAN PEMBAHASAN		48
4.1	Data Uji Coba.....	48
4.2	Pengujian Fungsionalitas Aplikasi	51
4.3	Pengujian Hasil Generate Aplikasi.....	57
4.4	Pengujian Variabel Algoritma Genetika	60
4.5	Perbandingan Hasil Uji Coba Variabel Algoritma Genetika	68
4.6	Pengujian Beberapa Kelas.....	69
4.7	Perbandingan Dengan Penelitian Sebelumnya.....	79
4.8	Kajian Integrasi Islam dan Sains	81
BAB V KESIMPULAN DAN SARAN		84
5.1	Kesimpulan.....	84
5.2	Saran	84
DAFTAR PUSTAKA		85



DAFTAR GAMBAR

Gambar 3. 1 Prosedur Kerja Penelitian.....	20
Gambar 3. 2 Alur Algoritma Genetika.....	21
Gambar 3. 3 Implementasi Pembangkit Gen	25
Gambar 3. 4 Implementasi Perhitungan Penalti Hard Constraint Tabrakan Jam Mengajar	28
Gambar 3. 5 Implementasi Perhitungan Pinalti Guru Terjadwal Pada Hari Libur	31
Gambar 3. 6 Implementasi Perhitungan Pinalti Guru Terjadwal pada Kelas yang Sama.....	33
Gambar 3. 7 Implementasi Perhitungan Pinalti Pelajaran	36
Gambar 3. 8 Implementasi Perhitungan Fitness	39
Gambar 3. 9 Contoh Roulette Wheel	40
Gambar 3. 10 Implementasi Proses Roulette Wheel.....	40
Gambar 3. 11 Implementasi Proses Crossover	43
Gambar 3. 12 Implementasi Proses Mutasi.....	44
Gambar 4. 1 Tampilan Halaman Master Guru.....	51
Gambar 4. 2 Tampilan Halaman Master Pelajaran	52
Gambar 4. 3 Tampilan Halaman Master Kelas.....	53
Gambar 4. 4 Tampilan Halaman Master Kelas - Pelajaran.....	54
Gambar 4. 5 Tampilan Halaman Master Guru - Pelajaran.....	55
Gambar 4. 6 Tampilan Halaman Master Guru - Pelajaran.....	56
Gambar 4. 7 Tampilan Halaman Jadwal	57
Gambar 4. 8 Grafik Fitness Per Generasi	58
Gambar 4. 9 Grafik Pinalti Pergenerasi	58
Gambar 4. 10 Pergerakan Penalti Pada Setiap Skenario.....	60
Gambar 4. 11 Pergerakan Fitness pada Setiap Skenario.....	62
Gambar 4. 12 Pergerakan Waktu Komputasi pada Setiap Skenario.....	62
Gambar 4. 13 Pergerakan Penggunaan Memori pada Setiap Skenario.....	63
Gambar 4. 14 Perbandingan Fitness, Waktu Komputasi dan Penggunaan Memori	64
Gambar 4. 15 Pergerakan Nilai Fitness Berdasarkan Populasi.....	66
Gambar 4. 16 Pergerakan Waktu Komputasi Berdasarkan Populasi.....	67

Gambar 4. 17 Pergerakan Penggunaan Memori Berdasarkan Populasi..... 67



DAFTAR TABEL

Tabel 2.1 Istilah dalam algoritma genetika	11
Tabel 3.1 Bentuk Representasi Kromosom.....	22
Tabel 3.2 Generasi pertama dengan lima individu.....	26
Tabel 3.3 Perhitungan pinalti tabrakan jam mengajar guru	29
Tabel 3.4 Data ketersediaan guru berdasarkan hari	30
Tabel 3.5 Perhitungan penalti ketersediaan guru di hari tertentu.....	32
Tabel 3.6 Perhitungan pinalti guru yang mengajar kelas yang sama dalam sehari.....	34
Tabel 3.7 Perhitungan pinalti pelajaran yang terjadwal di hari yang sama	37
Tabel 3.8 Proses Crossover Individu 1 dan Individu 2	42
Tabel 3.9 Proses Mutasi	44
Tabel 3.10 Skenario uji coba crossover rate dan mutasi rate seragam.....	46
Tabel 3.11 Skenario pengujian dengan crossover rate dan mutasi rate tidak sama	46
Tabel 3.12 Skenario uji coba jumlah populasi	47
Tabel 4.1 Pelajaran dan jam pelajaran	49
Tabel 4.2 Guru, Pelajaran dan Jumlah Jam Mengajar	49
Tabel 4.3 Hasil penyusunan jadwal	59
Tabel 4.4 Perhitungan Skor untuk Menentukan Skenario Terbaik	65
Tabel 4.5 Perhitungan Skor untuk Menentukan Skenario Terbaik Pengujian Kedua	65
Tabel 4.6 Skor Setiap Skenario Uji Coba Populasi.....	68
Tabel 4.7 Perbandingan Hasil Algoritma Genetika	68
Tabel 4.8 Hasil Percobaan Pertama Dua Kelas.....	70
Tabel 4.9 Skor Setiap Skenario Tahap Pertama Dua Kelas	70
Tabel 4.10 Hasil Percobaan Tahap Kedua Dua Kelas	71
Tabel 4.11 Skor Setiap Skenario Tahap Kedua Dua Kelas.....	72
Tabel 4.12 Hasil Percobaan Pertama Tiga Kelas	73
Tabel 4.13 Skor Setiap Skenario Tahap Pertama Tiga Kelas	74
Tabel 4.14 Hasil Percobaan Kedua Tiga Kelas.....	74
Tabel 4.15 Skor Setiap Skenario Tahap Pertama Tiga Kelas	75
Tabel 4.16 Hasil Percobaan Pertama Empat Kelas.....	76

Tabel 4.17 Skor Setiap Skenario Tahap Pertama Empat Kelas	77
Tabel 4.18 Hasil Percobaan Tahap Kedua Empat Kelas.....	78
Tabel 4.19 Skor Setiap Skenario Tahap Kedua Empat Kelas.....	79



ABSTRAK

Amal, Muhammad Fadhil Al. 2020. *Implementasi Algoritma Genetika Untuk Otomatisasi Penjadwalan Sekolah Menengah Pertama*. Skripsi. Jurusan Teknik Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Maulana Malik Ibrahim Malang. Pembimbing : (I) Fathurrochman, M.Kom (II) Ainatul Mardhiyah, M.Cs

Kata Kunci : Algoritma Genetika, Penjadwalan Pelajaran Sekolah, Otomatisasi

Kegiatan menyusun jadwal pelajaran merupakan kegiatan rutin yang dilakukan oleh bagian kurikulum sekolah menengah pertama. Menyusun jadwal pelajaran membutuhkan resource yang banyak, baik waktu maupun tenaga. Penelitian ini menerapkan algoritma genetika untuk melakukan otomatisasi penyusunan jadwal pelajaran untuk sekolah menengah pertama.

Penelitian ini berfokus untuk menentukan parameter algoritma genetika berupa crossover rate, mutation rate dan jumlah populasi yang dapat memberikan hasil terbaik berdasarkan nilai fitness, lama komputasi dan penggunaan memori. Pengujian dilakukan dengan 4 skenario, yaitu: penyusunan jadwal untuk 2 kelas, 3 kelas, 4 kelas dan 35 kelas. Hasil keseluruhan pengujian menunjukkan bahwa parameter terbaik berbeda untuk setiap skenario penyusunan jadwal.

ABSTRACT

Amal, Muhammad Fadhil Al. 2020. *Implementation of Genetic Algorithms for Junior High School Scheduling Automation*. Thesis. Department of Informatics, Faculty of Sains and Technology, Maulana Malik Ibrahim State Islamic University of Malang. Supervisor: (I) Fathurrochman, M.Kom. (II) Ainatul Mardhiyah, M.Cs

Keywords: Genetics algorithm, Scheduling subject in the school, Automation

Arrangement the schedule of subjects in the school is a periodic agenda that is done by Academic and Curriculum Affairs in the Junior High School. That activity needs many resources, both in term of time and energy. This research is applying Genetics Algorithm as the automation of arrangement schedule of the subjects for Junior High School.

This research is focused on determining Genetics Algorithm parameters in the form of crossover rate, mutation rate and the number of population which gives the best result depend on the value of fitness, the time of computation occurred and the using of memory. The testing is carried out in 4 scenarios, that is: arrangement of the schedule for two-classes, three-classes, four-classes and thirty five-classes. The completed result of the research showed that the best parameter is different for every scenario of the arrangement of the schedule.

مستخلص البحث

الأمل ، محمد فاضل. ٢٠٢٠. تنفيذ الخوارزميات الجينية لأتمتة جدولة المدارس الثانوية الإعدادية . البحث العلمي. كلية العلوم و التكنولوجيا . قسم الفن المعلوماتية. جامعة الإسلامية الحكومية مولانا مالك إبراهيم مالانج.

المشرف : (١) فتح الرحمان، الماجستير (٢) عينات المرضية، الماجستير

الكلمات المفتاحية : الخوارزمية الوراثية، جدولة الدروس المدرسية، الأتمتة

نشاط تجميع جدول الدروس هو نشاط روتيني يقوم به قسم مناهج المدرسة الثانوية. يحتاج تجميع جدول الدروس عديدة من المصادر، إما من الوقت أو طاقة كانت. تطبق هذه الدراسة خوارزمية وراثية لأتمتة تجميع جدول الدروس لمدرسة الثانوية.

تركز هذه الدراسة على تحديد معلمات الخوارزمية الوراثية في شكل معدل العبور و معدل الطفرات وعدد السكان التي تمكن أن تعطي أفضل النتائج بناءً على قيم اللياقة ووقت الحساب واستخدام الذاكرة. تم إجراء الاختبار بأربع سيناريوهات ، وهي: جدولة لفصلين، 3 فصول، 4 فصول و 35 فصلاً. تدلّ نتائج الاختبار الإجمالية على أن أفضل المعلمات تختلف لكل اختلاف سيناريو الجدولة.

BAB I PENDAHULUAN

1.1 Latar Belakang

Sekolah menengah pertama yang disingkat dengan SMP merupakan jenjang pendidikan dasar pada pendidikan formal di Indonesia setelah lulus sekolah dasar (atau sederajat). Sekolah menengah pertama ditempuh dalam waktu 3 tahun, dimulai dengan kelas 7 hingga kelas 9. Sekolah menengah pertama termasuk dalam program pemerintah wajib belajar 9 tahun. Sekolah menengah pertama diselenggarakan oleh pemerintah maupun swasta. Sejak diberlakukannya otonomi daerah pada tahun 2001, pengelolaan sekolah menengah pertama negeri di Indonesia menjadi tanggung jawab pemerintah daerah kabupaten/kota.

Jadwal pelajaran merupakan salah satu hal terpenting untuk menjamin kelancaran proses belajar mengajar di sekolah. Pembuatan jadwal pelajaran merupakan tanggung jawab rutin bagian kurikulum dimana hingga saat ini masih dilakukan secara manual. Mata pelajaran, jumlah ruang kelas dan guru pengampu merupakan elemen terpenting dalam penyusunan jadwal pelajaran, namun juga menjadi permasalahan utama dalam proses penjadwalan.

Sistem penjadwalan konvensional membutuhkan tingkat ketelitian tinggi dan waktu yang relatif banyak. Seiring dengan bertambahnya ruang kelas, guru dan mata pelajaran maka semakin besar sumber daya yang dibutuhkan untuk menyusun sebuah jadwal. Demikian pula dengan kemungkinan terjadinya kesalahan. Permasalahan yang umum ditemui adalah terjadinya bentrokan jam mengajar seorang guru, atau mata pelajaran yang berulang di hari yang sama dalam satu kelas. Berdasarkan permasalahan inilah, sistem otomatisasi penjadwalan mata pelajaran

sangat dibutuhkan. Penelitian ini menerapkan algoritma genetika untuk melakukan otomatisasi penjadwalan mata pelajaran sekolah, serta menemukan solusi algoritma genetika yang paling optimal dan efektif.

Algoritma genetika adalah algoritma komputasi yang meniru mekanisme pada proses evolusi yang diungkapkan Darwin. Proses evaluasi dilakukan pada sekumpulan kandidat solusi (kromosom) dengan mengikuti prinsip seleksi alam (Syarif, 2014). Algoritma genetika telah teruji dapat menyelesaikan berbagai permasalahan, diantaranya persoalan Travelling Salesman Problem (Varshika Dwivedi dkk, 2012), Scheduling (Budhi, 2015), Shortest Path Problem (Chang Wook Ahn, & Ramakrishna, R. S. (2002) dan lain sebagainya.

Penelitian ini diharapkan dapat membantu menyelesaikan masalah penjadwalan, sehingga dapat menekan kebutuhan sumber daya, waktu dan kesalahan yang mungkin terjadi. Selain itu, penelitian ini juga bertujuan untuk menemukan parameter optimal untuk algoritma genetika sehingga dapat memaksimalkan kinerja algoritma tersebut. Penggunaan sumber daya dan waktu yang efektif sesuai dengan Al-Quran surat Al-Ashr ayat 1 hingga 3 yang menjelaskan betapa pentingnya waktu bagi orang-orang beriman.

وَالْعَصْرِ إِنَّ الْإِنْسَانَ لَفِي خُسْرٍ إِلَّا الَّذِينَ آمَنُوا وَعَمِلُوا الصَّالِحَاتِ وَتَوَّصَوْا بِالْحَقِّ وَتَوَّصَوْا بِالصَّبْرِ

Artinya :

1. Demi masa, 2. sungguh, manusia berada dalam kerugian, 3. kecuali orang-orang yang beriman dan mengerjakan kebajikan serta saling menasihati untuk kebenaran dan saling menasihati untuk kesabaran.

1.2 Pernyataan Masalah

Berdasarkan latar belakang masalah yang telah dipaparkan maka rumusan masalah yang diangkat dalam penelitian ini adalah bagaimana pengaruh kombinasi variabel *crossover rate* dan *mutasi rate* terhadap kinerja algoritma genetika dalam menyelesaikan permasalahan penjadwalan sekolah ?

1.3 Tujuan Penelitian

Adapun tujuan yang diharapkan dari penelitian ini adalah mengetahui pengaruh kombinasi variabel *crossover rate* dan *mutasi rate* terhadap kinerja algoritma genetika ketika diterapkan pada permasalahan penjadwalan sekolah.

1.4 Manfaat Penelitian

Penelitian ini diharapkan dapat memberikan manfaat secara teoritis maupun praktis, baik bagi peneliti maupun pembaca. Manfaat teoritis yang diharapkan adalah algoritma genetika dapat diterapkan pada studi kasus lain atau dapat dikembangkan lagi dikemudian hari. Manfaat praktis yang diharapkan adalah aplikasi penjadwalan mata pelajaran yang menerapkan algoritma genetika dapat dimanfaatkan oleh sekolah untuk membantu proses pembuatan jadwal pelajaran

1.5 Batasan Masalah

Adapun batasan masalah pada penelitian ini adalah:

1. Variabel algoritma genetika yang diuji adalah *crossover rate*, *mutasi rate* dan jumlah populasi.
2. Parameter yang digunakan untuk menentukan komposisi variabel terbaik adalah nilai *fitness*, waktu komputasi dan penggunaan memori.

1.6 Sistematika Penulisan

Laporan penelitian ini terdiri dari lima bab, dimana isi dari setiap bab terdiri dari:

BAB I : PENDAHULUAN

Berisi tentang latar belakang dari masalah yang akan diteliti, tujuan dan manfaat penelitian, batasan masalah pada penelitian, metodologi penelitian, serta sistematika penulisan laporan penelitian.

BAB II: STUDI PUSTAKA

Bab ini berisi penjelasan mengenai penelitian yang telah dilakukan ataupun teori dasar dan data-data yang terkait dengan penjadwalan maupun algoritma genetika yang digunakan dalam penelitian ini.

BAB III : METODE PENELITIAN

Bab ini berisi metode penelitian yang menjelaskan bagaimana penelitian ini dijalankan. Meliputi hasil analisa dan rincian langkah yang digunakan dalam penyusunan jadwal mata pelajaran otomatis dengan menerapkan algoritma genetika.

BAB IV : UJI COBA DAN PEMBAHASAN

Bab ini berisi uji coba dari aplikasi yang telah dibuat dan dilakukan pembahasan mengenai data yang dihasil aplikasi penyusun jadwal mata pelajaran otomatis dengan menerapkan algoritma genetika.

BAB V : KESIMPULAN DAN SARAN

Pada bab ini berisi kesimpulan dari penelitian yang telah dilakukan, saran dan kritik dari penelitian agar dapat dikembangkan pada penelitian selanjutnya.



BAB II LANDASAN TEORI

2.1 Penjadwalan

Penjadwalan merupakan proses pengorganisasian, pemilihan, dan penentuan waktu penggunaan sumber daya yang ada untuk menghasilkan *output* seperti yang diharapkan dalam waktu yang diharapkan juga. Penjadwalan merupakan sebuah proses pengambilan keputusan yang sering dilakukan di industri manufaktur maupun bidang pelayanan atau jasa (Pinedo, 2012). Persoalan penjadwalan berkaitan dengan pengalokasian sumber daya (*resource*) ke dalam tugas-tugas atau fungsi-fungsi tertentu.

Menurut Pinedo (2012), penjadwalan didefinisikan sebagai proses pengalokasian sumber daya (*resource*) untuk mengerjakan sekumpulan tugas dalam jangka waktu tertentu dengan dua arti penting sebagai berikut:

- a. Penjadwalan merupakan suatu fungsi pengambilan keputusan untuk membuat atau menentukan jadwal
- b. Penjadwalan merupakan suatu teori yang berisi sekumpulan prinsip dasar, model teknik, dan kesimpulan logis dalam proses pengambilan keputusan yang memberikan pengertian dalam fungsi penjadwalan.

Masalah penjadwalan biasanya dipecahkan secara manual yang membutuhkan banyak orang dan waktu. Meskipun telah menggunakan sumber daya yang besar, solusi yang diperoleh juga tidak selalu memuaskan. Seringkali terjadi ketidaksesuaian pada jadwal yang dihasilkan, sehingga ditemukan beberapa kegagalan dalam proses berjalannya jadwal.

Sebagian besar masalah penjadwalan dalam berbagai literatur dibedakan berdasarkan jenis institusi yang membutuhkan penjadwalan. Pillay, N. (2014) mengklasifikasikan masalah penjadwalan menjadi tiga klasifikasi utama:

1. Penjadwalan sekolah. Berupa penjadwalan mingguan untuk semua kelas dalam sebuah sekolah. *Constraint* utama dalam penjadwalan sekolah adalah seorang guru yang mengajar pada dua kelas di waktu yang sama.
2. Penjadwalan kuliah. Berupa penjadwalan mingguan untuk sebuah perkuliahan. *Constraint* utama dalam penjadwalan kuliah adalah memanfaatkan ruang yang tersedia sedemikian rupa untuk memaksimalkan perkuliahan.
3. Penjadwalan ujian. Berupa penjadwalan untuk satu set ujian. *Constraint* utama dalam penjadwalan ujian adalah menghindari tumpang tindih ujian yang diikuti mahasiswa, dan sebaran ujian untuk siswa sebanyak mungkin.

Selama beberapa dekade, perhatian yang cukup besar dicurahkan peneliti dalam masalah penjadwalan. Banyak makalah terkait penjadwalan otomatis telah dipublikasikan (Schaerf, 1999). Aulia (2012) berhasil menerapkan algoritma *harmony search* dalam permasalahan *flow shop* untuk *minimize makespan*. Arifuddin (2016) berhasil mengoptimalkan penjadwalan proyek dengan penyeimbangan biaya dengan mengkombinasikan CPM dan algoritma genetika. Lesmana (2016) berhasil menerapkan metode *branch and bound* dalam penjadwalan produksi untuk meminimalkan waktu produksi.

2.2 Sekolah Menengah Pertama

Atmodiwiro (2000) mendefinisikan sekolah sebagai suatu sistem interaksi sosial suatu organisasi keseluruhan yang terdiri dari interaksi pribadi terkait bersama dalam suatu hubungan organik. Sedangkan berdasarkan undang-undang no 2 tahun 1989, sekolah adalah satuan pendidikan yang berjenjang dan berkesinambungan untuk menyelenggarakan kegiatan belajar mengajar. Sekolah lebih lanjut diatur berdasarkan peraturan pemerintah (PP) nomor 19 tahun 2005 tentang standar nasional pendidikan. Dengan PP 19/2005 itu, seluruh sekolah di Indonesia diarahkan menyelenggarakan pendidikan yang memenuhi standar nasional. Termasuk dalam standar pendidikan nasional adalah pembagian beban mengajar guru. Menteri pendidikan dan budaya Muhadjir Effendy menetapkan peraturan Menteri nomor 23 tahun 2017 tentang hari sekolah mengatur sekolah 8 jam sehari selama 5 hari sepekan dan resmi diterapkan pada tahun ajaran 2017/2018.

Permendikbud Nomor 6 Tahun 2019 tentang Pedoman Organisasi dan Tata Kerja Satuan Pendidikan Dasar dan Menengah menjelaskan bahwa Sekolah Menengah Pertama yang selanjutnya disingkat SMP, adalah salah satu bentuk Satuan Pendidikan formal yang menyelenggarakan pendidikan umum pada jenjang pendidikan dasar sebagai lanjutan dari SD, MI, atau bentuk lain yang sederajat atau lanjutan dari hasil belajar yang diakui sama atau setara SD atau MI.

Lulusan sekolah menengah pertama dapat melanjutkan pendidikan ke sekolah menengah atas atau sekolah menengah kejuruan (atau sederajat). Umumnya pelajar sekolah menengah pertama berusia 13 hingga 15 tahun. Sekolah menengah pertama termasuk dalam program pemerintah wajib belajar 9 tahun

untuk anak berusia 7 hingga 15 tahun. Program wajib belajar 9 tahun terdiri dari sekolah dasar (atau sederajat) yang ditempuh selama 6 tahun dan sekolah menengah pertama (atau sederajat) yang ditempuh dalam waktu 3 tahun.

Penyelenggaraan sekolah menengah pertama dapat dilakukan oleh pemerintah atau swasta. Sejak diberlakukannya otonomi daerah tahun 2001, pengelolaan sekolah menengah pertama negeri di Indonesia menjadi tanggung jawab pemerintah daerah kabupaten/kota, yang sebelumnya berada di bawah Kementerian Pendidikan Nasional. Dengan pemindahan tanggung jawab pengelolaan ini, Kementerian Pendidikan Nasional hanya berperan sebagai regulator dalam bidang standar pendidikan nasional. Secara struktural, sekolah menengah pertama negeri merupakan unit pelaksana teknis dinas pendidikan kabupaten/kota.

2.3 Penjadwalan Sekolah

Penelitian pada masalah penjadwalan sekolah tidak berkembang secepat daripada penjadwalan kuliah dan penjadwalan ujian. Hal ini bisa terjadi karena penelitian yang dilakukan secara terpisah dan hanya untuk sekolah tertentu serta kurangnya berbagai masalah yang dapat diakses publik. Selain itu juga belum teruji pada berbagai masalah penjadwalan sekolah guna memastikan seberapa baik metode yang digunakan dapat digeneralisasi (Pillay, 2013).

Valouxis dan Housos (2003) menggunakan *constraint programming* (CP) yang dikombinasikan dengan pencarian lokal untuk memecahkan masalah penjadwalan sekolah untuk sekolah menengah Yunani. *Constraint programming* digunakan untuk menemukan jadwal yang layak. Kualitas jadwal kemudian

ditingkatkan menggunakan pencarian lokal sampai perbaikan lebih lanjut tidak mungkin.

Beligiannis, dkk. (2008) menggunakan algoritma genetika untuk menyelesaikan masalah penjadwalan sekolah. Setiap elemen populasi adalah matriks dengan baris sesuai jumlah kelas dan kolom sesuai dengan jumlah periode. Setiap sel dalam matriks menyimpan kode guru yang akan mengajar kelas pada periode tertentu. Studi awal menunjukkan bahwa *crossover* tidak efektif dan memakan waktu dan karenanya tidak digunakan. Operator mutasi periode menukar guru antara dua periode jadwal dalam satu kelas. Periode yang dipilih untuk bertukar ditentukan secara acak. Kromosom terbaik dari setiap generasi disalin ke generasi berikutnya. Algoritma ini berhasil menghasilkan solusi untuk masalah penjadwalan sekolah menengah Yunani.

Hidayatulloh (2015) berhasil menerapkan metode *graph coloring* dalam sistem pembelajaran. Walaupun merupakan metode yang sederhana, *graph coloring* dapat memberikan hasil yang baik untuk penjadwalan pelajaran dengan rata-rata 93 persen memenuhi *constraints* yang ditetapkan.

2.4 Algoritma Genetika

Algoritma genetika diperkenalkan oleh Holland pada tahun 1975. Sesuai dengan namanya, algoritma genetika meniru mekanisme evolusi yang dikemukakan oleh Darwin. Pada aplikasinya, algoritma genetika biasanya digunakan untuk memperoleh solusi optimal ataupun solusi pendekatan dari solusi yang memiliki kemungkinan yang sangat banyak.

2.3.1 Istilah dalam Algoritma Genetika

Berikut merupakan berbagai istilah yang sering digunakan dalam algoritma genetika (Syarif, 2014):

Tabel 02.1 Istilah dalam algoritma genetika

Kromosom	Kandidat solusi dari persoalan yang akan diselesaikan
Populasi	Sekelompok kandidat solusi yang digunakan pada algoritma genetika
Gen	Komponen dari kromosom (kromosom biasanya terdiri dari sekelompok gen)
<i>Parent</i>	Kromosom yang dipilih sebagai induk untuk proses reproduksi
<i>Offspring</i>	Kromosom baru yang diperoleh setelah proses reproduksi
<i>Fitness</i>	Suatu nilai yang mewakili kualitas dari suatu kromosom
<i>Crossover</i>	Proses reproduksi yang dilakukan dengan perkawinan silang antara dua kromosom
Mutasi	Proses reproduksi dengan memodifikasi gen yang ada dalam kromosom

2.3.2 Parameter Algoritma Genetika

Penentuan nilai parameter berperan penting terhadap efektifitas algoritma genetika. Apabila parameter terlalu besar maka proses algoritma genetika akan memakan waktu lama, demikian sebaliknya jika parameter terlalu kecil maka kemungkinan hasil yang diperoleh bukanlah merupakan nilai optimal yang diharapkan. Mengingat pentingnya parameter ini, beberapa peneliti telah mencoba melakukan penelitian untuk menentukan pengaruh parameter terhadap efektifitas algoritma genetika (Gen dan Cheng, 2000). Adapun parameter yang perlu ditentukan adalah *Probabilitas Crossover*, *Probabilitas Mutation*, *Population Size*, dan *Maksimum Generation*.

2.3.2.1 Probabilitas Crossover

Parameter probabilitas *crossover* memiliki nilai antara 0-1. Nilai ini menggambarkan seberapa sering *crossover* akan dilakukan. Mudahnya, jika nilai probabilitas *crossover* adalah 0, maka tidak dilakukan *crossover*. Sebaliknya, jika probabilitas *crossover* bernilai 1, maka semua kromosom berpeluang melakukan *crossover* meskipun tidak harus semua.

2.3.2.2 Probabilitas Mutasi (*Probabilitas Mutation*)

Sama halnya dengan probabilitas *crossover*, probabilitas mutasi juga memiliki nilai antara 0-1. Nilai ini menunjukkan seberapa sering proses mutasi terhadap kromosom akan dilakukan. Jika nilai parameter ini 0, berarti tidak dilakukan proses mutasi kromosom. Sebaliknya jika parameter ini bernilai 1, maka semua kromosom mempunyai kesempatan melakukan proses mutasi.

2.3.2.3 Ukuran Populasi (*Population Size*)

Nilai parameter ukuran populasi menunjukkan jumlah kromosom pada populasi (setiap generasi). Apabila ukuran populasi terlalu kecil, maka akan semakin sedikit kromosom yang melakukan *crossover* dan mutasi, sehingga akan mengurangi kualitas solusi yang diperoleh. Demikian sebaliknya, jika ukuran populasi terlampaui besar maka proses algoritma genetika akan sangat lama.

2.3.2.4 Maksimum Generasi (*Maksimum Generation*)

Nilai parameter maksimum generasi menunjukkan jumlah maksimum generasi dalam algoritma genetika. Dengan kata lain, parameter ini menjadi kriteria pemberhentian dari proses algoritma genetika.

2.3.3 Struktur Algoritma Genetika

2.3.2.5 Representasi Kromosom

Representasi kromosom adalah langkah awal dalam algoritma genetika. Langkah ini memegang peranan penting dalam proses selanjutnya dalam algoritma genetika sehingga berpengaruh besar terhadap efektifitas dan efisiensi algoritma genetika. Proses representasi biasa juga disebut proses *encoding*, yaitu menyatakan karakteristik dari persoalan dalam bentuk sekumpulan *string* baik bilangan maupun alphabet. Representasi yang baik haruslah sesuai dengan permasalahan yang akan diselesaikan dan dapat mewakili semua parameter dan solusi yang mungkin.

Secara umum, pengelompokan metode representasi dapat dibedakan berdasarkan berbagai kriteria sebagai berikut:

1. Berdasarkan simbol yang digunakan (*binary string*, *real* dan *integer*)
2. Berdasarkan struktur dari kromosom (kromosom dengan satu dimensi dan kromosom dengan multi-dimensi)
3. Berdasarkan panjang/ukuran dari kromosom (kromosom dengan panjang tetap dan kromosom dengan panjang berubah)
4. Berdasarkan isi/konten dari kromosom (kromosom yang hanya berisi solusi dan kromosom dengan solusi sekaligus parameter)

Adapun beberapa metode representasi yang sering digunakan sebagai berikut:

1. Representasi Nilai (*Value encoding*). Representasi gen dengan langsung berisikan nilai atau karakter calon penyelesaian atas suatu

permasalahan. Metode ini sering digunakan untuk optimasi fungsi dengan beberapa variabel, *Word Matching Problem* dan sebagainya.

2. Representasi Biner (*Binary Encoding*). Representasi yang mengisi setiap gen dengan bilangan biner. Metode representasi yang sering digunakan untuk optimasi fungsi-fungsi matematika.
3. Representasi Permutasi (*Permutation Encoding*). Representasi dengan setiap gen memiliki nilai yang berbeda. Metode yang sering digunakan untuk persoalan yang berkaitan dengan prioritas atau mementingkan urutan gen, seperti *Travelling Salesman Problem*, *Shortest Path Problem* atau *Scheduling Problem*.
4. Representasi Pohon (*Tree Encoding*). Representasi yang digunakan untuk permasalahan yang solusinya memiliki struktur pohon. Metode yang biasa digunakan untuk persoalan transportasi dan *minimum spanning tree*.
5. Representasi Matrik (*Matrik Encoding*). Representasi dengan bentuk matrik, baik satu dimensi atau lebih. Persoalan logistik juga dapat diselesaikan dengan metode representasi ini.

2.3.3.2 Pembentukan Gen Awal

Generasi awal dari algoritma genetika biasanya dibangkitkan secara acak (*random*) mengikuti metode representasi kromosom yang sesuai dengan permasalahan yang akan diselesaikan. Namun untuk menghindari solusi yang tidak layak (*infeasible solution*), perlu adanya proses uji kelayakan dengan beberapa pembatas (*constraint*) yang ada.

Terdapat dua cara yang sering digunakan untuk menangani kromosom yang tidak layak pada generasi, atau dikenal dengan istilah *rejecting strategi*. Cara pertama adalah dengan membuang kromosom tersebut dan membangkitkan kromosom baru hingga diperoleh kromosom yang layak. Cara kedua adalah dengan membuat suatu prosedur untuk memperbaiki kromosom tersebut sehingga menjadi layak (*repairing prosedur*) (Admi dan Gen, 2003).

2.3.3.3 Evaluasi Kromosom

Evaluasi kromosom bertujuan untuk mendapatkan kromosom yang terbaik untuk generasi selanjutnya sehingga didapatkan penyelesaian masalah. Metode evaluasi kromosom yang umum digunakan adalah dengan menghitung nilai fungsi tujuan sebagai *fitness value*. Metode lain yang juga sering digunakan adalah dengan memberikan nilai penalti (*penalty strategy*).

2.3.3.4 Crossover

Crossover adalah proses persilangan yang bertujuan untuk membentuk kromosom turunan (*offspring*) dengan menggabungkan elemen dari elemen induk terpilih (*parent*). Proses *crossover* bertujuan untuk mendapatkan solusi yang lebih baik. Metode *crossover* berkaitan dengan metode representasi yang digunakan, beberapa metode *crossover* hanya sesuai dengan metode representasi tertentu. Menggunakan metode yang sesuai merupakan sebuah keharusan agar algoritma genetika dapat mendapatkan penyelesaian dari suatu permasalahan. Berikut beberapa metode *crossover* yang umum digunakan:

1. *Crossover Satu Titik (One-point crossover)*
2. *Crossover Dua Titik (Two-point crossover)*
3. *Partial Mapped Crossover (PMX)*
4. *Order Crossover (OX)*
5. *Position-based crossover (PX)*
6. *Order-based Crossover*
7. *Cycle Crossover (CX)*
8. *Weight Mapping Crossover (WMX)*

2.3.3.5 Mutasi (*Mutation*)

Mutasi dilakukan dengan mengubah gen pada suatu kromosom untuk meningkatkan keberagaman kromosom yang ada sehingga tidak terperangkap pada solusi *local optimum*. Berikut beberapa metode mutasi yang umum digunakan:

1. Metode pembalikan (*Inversion mutation*), dilakukan dengan mengambil suatu *substring* di antara dua titik yang ditentukan secara acak, kemudian dilakukan proses pembalikan (*invers*) gen pada *substring* tersebut.
2. Metode penyisipan (*Insertion mutation*), dilakukan dengan memilih satu gen secara acak pada *parent*, kemudian disisipkan pada posisi yang juga dipilih secara acak.
3. Metode pemindahan (*Displacement mutation*), dilakukan dengan memilih *substring* pada *parent* dengan dua titik yang ditentukan secara acak, kemudian *substring* tersebut dipindahkan pada titik yang juga ditentukan secara acak.

4. Metode penukaran (*Swap mutation*), dilakukan dengan memilih dua gen pada kromosom, kemudian kedua gen tersebut saling dipertukarkan.
5. Metode pergantian (*Flip mutation*), dilakukan hanya pada metode representasi biner dengan mengganti nilai biner pada gen terpilih, baik satu atau banyak gen.

2.3.3.6 Seleksi

Memilih kromosom yang akan tetap dipertahankan di generasi selanjutnya merupakan salah satu proses yang penting dalam algoritma genetika. Sesuai dengan teori evolusi Darwin, hanya kromosom terbaik yang dipilih ke generasi berikutnya. Secara umum, metode seleksi yang digunakan dalam algoritma genetika dapat dikelompokkan menjadi dua: metode proporsional (*proportional selection methods*) dan *ordinal-based selection methods*.

Metode proporsional memilih kromosom berdasarkan nilai *fitness relative* terhadap nilai fitness kromosom pada populasi, yang termasuk dalam kelompok ini diantaranya *roulette wheel selection*, *stochastic universal selection*, dan sebagainya. Adapun *ordinal-base selection methods* memilih kromosom berdasarkan ranking suatu kromosom pada populasi, yang termasuk dalam kelompok ini diantaranya *elitist selection*, *tournament selection*, *rank selection* dan lain sebagainya.

Berikut sekilas ulasan tentang beberapa metode seleksi yang sering digunakan.

1. *Roulette Wheel Selection*

Metode ini memberikan peluang bagi setiap kromosom untuk terpilih ke generasi selanjutnya sesuai dengan nilai *fitness* kromosom tersebut. Kromosom dengan nilai *fitness* yang baik tentu akan mempunyai peluang lebih besar untuk terpilih ke generasi berikutnya, demikian sebaliknya. Salah satu kelemahan metode ini terjadi ketika perbedaan nilai *fitness* yang sangat besar diantara kromosom. Maka salah satu kromosom akan memiliki peluang yang sangat besar untuk terpilih ke generasi selanjutnya, sedangkan kromosom lainnya memiliki peluang yang sangat kecil. Hal ini berakibat pada berkurangnya keragaman kromosom pada suatu populasi yang pada akhirnya menghasilkan solusi yang kurang baik.

2. *Rank Selection*

Metode ini digunakan untuk menghindari kendala pada metode seleksi *roulette-wheel*. Metode ini dilakukan dengan memberikan nilai ranking pada masing-masing kromosom. Kromosom dengan nilai *fitness* terburuk diberikan nilai satu, demikian seterusnya hingga kromosom dengan nilai *fitness* terbaik diberi nilai n (jumlah kromosom) sehingga masing-masing kromosom mempunyai peluang yang proporsional untuk terpilih. Namun kendala pada metode ini adalah ada peluang yang lebih besar kromosom dengan nilai *fitness* terbaik tidak terpilih ke generasi selanjutnya.

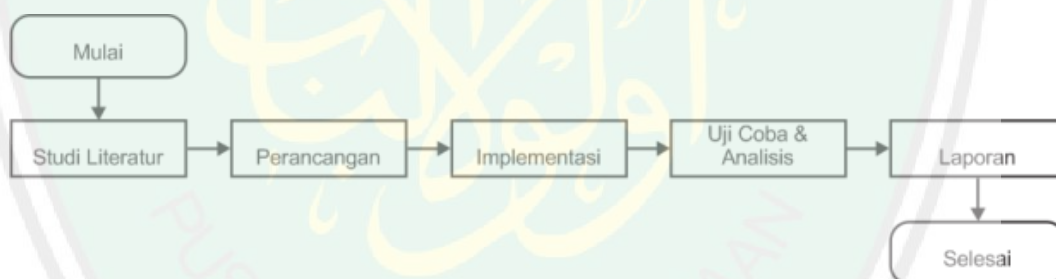
3. *Elitist*

Metode ini digunakan untuk menghindari kromosom dengan nilai *fitness* terbaik tidak terpilih ke generasi selanjutnya. Metode *elitist* dilakukan dengan mengambil kromosom dengan nilai *fitness* terbaik ke generasi berikutnya. Pemilihan kromosom lain dapat dilakukan dengan metode *Roulette Wheel* atau metode *Rank*.



BAB III PERANCANGAN DAN IMPLEMENTASI

Perancangan dan implementasi menjelaskan bagaimana penelitian ini dirancang dan rincian yang digunakan dalam pembuatan aplikasi penyusunan jadwal mata pelajaran dengan menerapkan algoritma genetika. Tahapan pertama dari penelitian ini adalah melakukan persiapan awal berupa studi literatur guna mendalami materi dan penelitian sebelumnya tentang penjadwalan dan algoritma genetika. Langkah kedua adalah melakukan perancangan algoritma genetika yang akan diterapkan dalam sistem penjadwalan. Tahap selanjutnya adalah implementasi (*coding*) algoritma genetika yang telah dirancang pada tahap sebelumnya ke sistem penjadwalan. Langkah berikutnya adalah uji coba sistem sekaligus analisis hasil implementasi. Dilanjutkan tahapan terakhir yaitu pembuatan laporan.

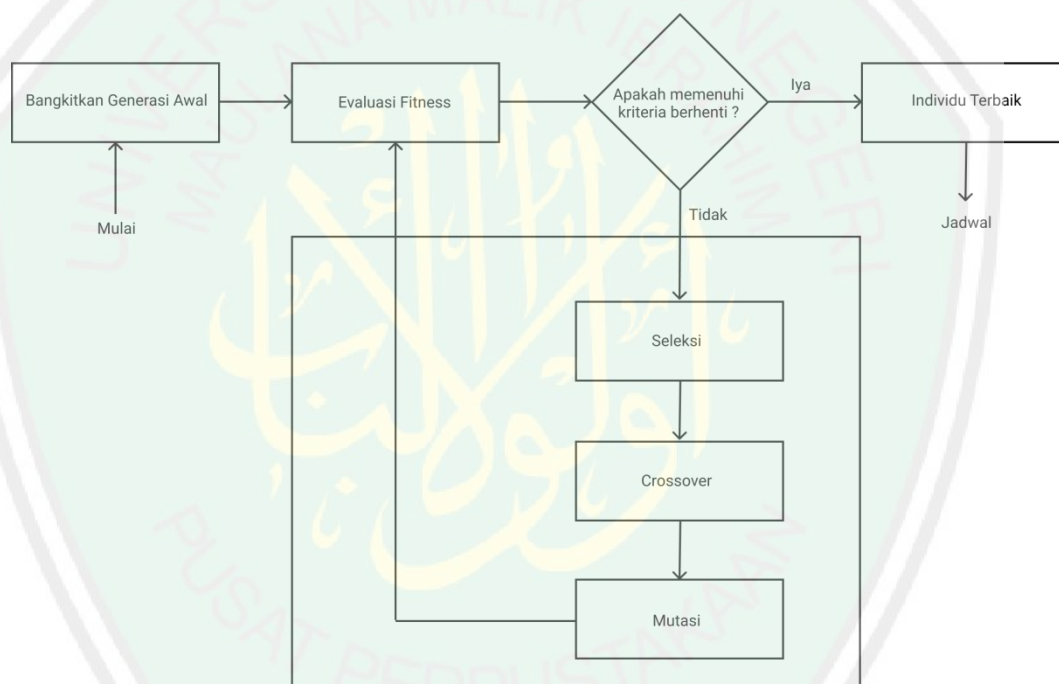


Gambar 3. 1 Prosedur Kerja Penelitian

3.1 Perancangan Sistem

Rancangan alur algoritma genetika diuraikan pada Gambar 3.2. Algoritma genetika dimulai dengan menentukan representasi kromosom. Tahapan awal ini memiliki peranan penting dalam algoritma genetika karena berpengaruh dalam setiap proses algoritma ini. Langkah kedua adalah membangkitkan generasi awal. Kemudian dilanjutkan dengan mengevaluasi nilai *fitness* dari setiap individu dalam satu generasi. Setelah mendapatkan nilai *fitness*, langkah selanjutnya adalah

memeriksa apakah nilai *fitness* sudah memenuhi persyaratan berhenti atau telah mencapai generasi maksimal. Jika belum, maka dilanjutkan dengan proses pembentukan individu baru untuk generasi selanjutnya dengan tahapan seleksi, *crossover* dan mutasi secara berurutan. Setelah terbentuknya generasi baru, kembali dilakukan proses perhitungan nilai *fitness*. Jika nilai *fitness* telah terpenuhi atau telah mencapai generasi maksimal maka individu terbaik pada generasi tersebut adalah solusi yang dihasilkan. Hasil dari keseluruhan proses tersebut adalah sebuah jadwal pelajaran sekolah.



Gambar 3. 2 Alur Algoritma Genetika

3.2 Representasi Kromosom

Representasi kromosom adalah salah satu faktor signifikan yang menentukan efektifitas dan efisiensi algoritma genetika. Kromosom harus memuat seluruh informasi yang diperlukan untuk pembentukan jadwal sekolah. Kromosom yang baik seharusnya memiliki struktur yang sederhana dan mudah untuk diproses untuk reproduksi.

Seluruh informasi yang dibutuhkan dimuat dalam kromosom secara jelas untuk mempermudah dan mempercepat proses perhitungan. Kebutuhan *hard constraint* ada pada guru, sedangkan *soft constraint* membutuhkan mata pelajaran dan jumlah jam pelajaran dalam sehari. Sehingga sebuah kromosom harus memuat kode guru dan mata pelajaran. Sedangkan jam pelajaran tersirat secara intrinsik pada posisi *index array*.

Penulis mengusulkan representasi kromosom dengan bentuk *array* dua dimensi $m \times n$, dimana m adalah jumlah kelas dan n adalah jumlah slot waktu per kelas dalam satu minggu. Jika jadwal mingguan dibagi menjadi 6 hari, dan setiap hari ada 4 pelajaran dalam satu kelas, jumlah sel dari deretan *array* sama dengan 24. Dalam setiap sel disimpan kode guru, mata pelajaran serta jumlah jam pelajaran. Posisi setiap sel menunjukkan secara implisit ke kelas dan waktu tertentu. Contoh representasi kromosom yang dimaksud dapat dilihat pada Tabel 3.1.

Tabel 03.1 Bentuk Representasi Kromosom

	Waktu 1	Waktu 2	Waktu 3	...	Waktu n
Kelas 1	Guru A	Guru A	Guru C	...	Guru B
Kelas 2	Guru C	Guru D	Guru B	...	Guru C
Kelas 3	Guru G	Guru E	Guru D	...	Guru C
...	...	...	...	...	...
Kelas m	Guru E	Guru F	Guru A	...	Guru B

Pengelompokan metode representasi diatas termasuk dalam metode representasi matriks dua dimensi. Sedangkan secara umum, pengelompokannya sebagai berikut:

1. Berdasarkan simbol yang digunakan tergolong dalam representasi *integer*, karena memuat kode guru dalam bentuk bilangan *integer*.

2. Berdasarkan struktur dari kromosom tergolong dalam representasi multi dimensi, karena berbentuk array dua dimensi berukuran $m \times n$.
3. Berdasarkan panjang dan ukuran dari kromosom tergolong dalam representasi ukuran tidak tetap, karena ukuran kromosom mengikuti jumlah slot waktu dan kelas.
4. Berdasarkan isi dari kromosom tergolong dalam representasi hanya dengan solusi, karena tidak memuat parameter.

Representasi kromosom dirancang agar dapat memuat seluruh informasi yang dibutuhkan berupa kelas, slot waktu dan guru. Representasi kromosom sebagaimana dijelaskan sebelumnya juga dirancang untuk menghindari kesalahan penempatan guru pada gelas yang tidak seharusnya dan menjaga agar jam mengajar setiap guru sesuai dengan yang telah ditetapkan.

3.3 Pembentukan Gen Awal

Gen awal berperan penting untuk proses pencarian solusi dalam algoritma genetika. Gen awal yang terlalu *random* dapat menyebabkan proses menjadi lebih panjang. Sehingga dibutuhkan sebuah prosedur untuk membangkitkan kromosom yang layak. Pembentukan kromosom berdasarkan waktu mengajar seorang guru.

Sebagai contoh, jika guru tertentu memiliki jam mengajar di kelas A, maka guru tersebut akan diberikan slot mengajar di kelas A. Sedangkan jika guru tersebut tidak mengajar di suatu kelas, atau jam mengajarnya telah terisi semua, maka akan dipilihkan guru lain. Sehingga ketetapan awal mengenai jumlah jam mengajar di suatu kelas terpenuhi. Prosedur tersebut dapat dilihat pada *pseudocode* berikut:

```

For each kromosom do
  For each kelas do
    For each slot waktu do
      1. Pilih guru secara acak
      2. If guru terpilih masih memiliki jam mengajar, then
        Masukan guru terpilih ke kelas dan slot waktu
        Kurangi jam mengajar guru
      Else
        Ulangi langkah 1
      End If
    End For
  End For
End For

```

Pada *pseudocode* diatas dapat dipahami proses mendapatkan kromosom, diawali dengan sistem akan memastikan bahwa subyek atau guru memiliki slot pada kelas yang diinginkan atau tidak. Apabila tidak memiliki slot kelas maka sistem akan mencari guru untuk mengisi slot tersebut. Jika telah mendapatkan guru sesuai dengan slot maka guru tersebut secara otomatis akan dimasukkan pada kelas dan slot waktu dan jam mengajar guru tersebut secara otomatis akan berkurang. Jika sistem gagal maka, sistem akan melakukan pengulangan pada tahap pencarian guru secara acak hingga sesuai dengan slot yang dibutuhkan. Penerapan *pseudocode* tersebut dalam bahasa pemrograman PHP dapat dilihat pada Gambar 3.3.

```

application > controllers > Welcome.php
116 function generate_gen($kelas_pelajaran, $guru_pelajaran) {
117     $kolom = array_column($guru_pelajaran, 'id_pelajaran');
118     foreach ($kelas_pelajaran as $k => $kelas) { // Proses Menyusun Guru yang mengajar dalam setiap kelas
119         foreach ($kelas['kelas_pelajaran'] as $k2 => $val) { // Untuk setiap pelajaran dalam kelas
120             // Cari guru yang mengajar pelajaran
121             $cari = array_search($val->id_pelajaran, $kolom);
122             $data[$k2]['pelajaran'] = $val->pelajaran;
123             $data[$k2]['id_pelajaran'] = $val->id_pelajaran;
124             $data[$k2]['jam_pelajaran'] = $val->jam_pelajaran;
125             if (!empty($guru_pelajaran[$cari]['guru_pelajaran'][0]) AND
126                 $guru_pelajaran[$cari]['guru_pelajaran'][0]->jam_mengajar >= $val->jam_pelajaran)
127             {
128                 $data[$k2]['guru'] = $guru_pelajaran[$cari]['guru_pelajaran'][0]->guru;
129                 $data[$k2]['id_guru'] = $guru_pelajaran[$cari]['guru_pelajaran'][0]->id_guru;
130                 $guru_pelajaran[$cari]['guru_pelajaran'][0]->jam_mengajar
131                 = $guru_pelajaran[$cari]['guru_pelajaran'][0]->jam_mengajar - $val->jam_pelajaran;
132                 if ($guru_pelajaran[$cari]['guru_pelajaran'][0]->jam_mengajar == 0) {
133                     array_shift($guru_pelajaran[$cari]['guru_pelajaran']);
134                 }
135             }
136         }
137     }
138     $kelas_guru[$k]['id_kelas'] = $kelas['id_kelas'];
139     $kelas_guru[$k]['kelas'] = $kelas['kelas'];
140     $kelas_guru[$k]['guru'] = $data;
141 }
142 foreach ($kelas_guru as $k => $value) { // Proses membagi jam pelajaran menjadi maksimal 2 jam pelajaran
143     $pelajaran = [];
144     foreach ($value['guru'] as $k2 => $val) {
145         while ($val['jam_pelajaran'] > 0) {
146             $data = $val;
147             if ($val['jam_pelajaran'] > 2) {
148                 $data['jam_pelajaran'] = 2;
149                 $val['jam_pelajaran'] -= 2;
150             } else {
151                 $data['jam_pelajaran'] = $val['jam_pelajaran'];
152                 $val['jam_pelajaran'] = 0;
153             }
154             array_push($pelajaran, $data);
155         }
156         shuffle($pelajaran);
157         $kelas_guru[$k]['guru'] = $pelajaran;
158     }
159     return $kelas_guru;
160 }

```

Gambar 3. 3 Implementasi Pembangkit Gen

Contoh individu pada generasi pertama dengan 5 kelas dan 22 slot waktu dapat dilihat pada Tabel 3.2. Individu pada tahap ini merupakan sebuah bentuk penjadwalan yang belum baik. Individu ini akan terus berkembang dalam algoritma genetika sehingga dapat menjadi sebuah penjadwalan yang baik di akhir proses.

Tabel 03.2 Generasi pertama dengan lima individu

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
A	D4	B2	B2	B2	B2	B2	B2	E5	E5	C3	C3	A1	C3	D4	D4	A1	A1	E5	A1	A1	E5	E5
B	A1	A1	D4	B2	B2	D4	D4	A1	E5	C3	C3	B2	B2	A1	A1	C3	E5	E5	B2	B2	E5	E5
C	B2	B2	C3	B2	B2	D4	B2	B2	A1	E5	E5	A1	A1	C3	C3	E5	D4	D4	A1	A1	E5	E5
D	E5	E5	E5	E5	E5	B2	B2	C3	A1	A1	B2	B2	C3	C3	A1	B2	B2	A1	A1	D4	D4	D4
E	C3	B2	B2	D4	D4	C3	C3	E5	E5	A1	A1	B2	B2	D4	A1	A1	A1	E5	E5	B2	B2	E5
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
A	C3	D4	D4	E5	E5	E5	E5	C3	C3	A1	A1	A1	B2	B2	B2	B2	E5	D4	A1	A1	B2	B2
B	E5	A1	A1	A1	C3	C3	C3	B2	B2	B2	B2	B2	B2	D4	E5	E5	E5	E5	A1	A1	D4	D4
C	C3	D4	B2	B2	A1	A1	E5	E5	E5	E5	A1	A1	C3	C3	B2	B2	E5	B2	B2	D4	D4	A1
D	D4	D4	A1	A1	A1	B2	B2	E5	E5	E5	E5	B2	B2	A1	A1	C3	D4	B2	B2	E5	C3	C3
E	B2	B2	A1	A1	C3	A1	A1	A1	E5	E5	E5	B2	B2	D4	D4	B2	B2	D4	C3	C3	E5	E5
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
A	C3	D4	D4	E5	E5	E5	E5	C3	C3	A1	A1	A1	B2	B2	B2	B2	E5	D4	A1	A1	B2	B2
B	C3	D4	B2	B2	A1	A1	E5	E5	E5	E5	A1	A1	C3	C3	B2	B2	E5	B2	B2	D4	D4	A1
C	C3	D4	D4	E5	E5	E5	E5	C3	C3	A1	A1	A1	B2	B2	B2	B2	E5	D4	A1	A1	B2	B2
D	B2	B2	C3	B2	B2	D4	B2	B2	A1	E5	E5	A1	A1	C3	C3	E5	D4	D4	A1	A1	E5	E5
E	C3	D4	B2	B2	A1	A1	E5	E5	E5	E5	A1	A1	C3	C3	B2	B2	E5	B2	B2	D4	D4	A1
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
A	E5	A1	A1	A1	C3	C3	C3	B2	B2	B2	B2	B2	B2	D4	E5	E5	E5	E5	A1	A1	D4	D4
B	D4	D4	A1	A1	A1	A1	B2	B2	C3	C3	E5	E5	E5	E5	D4	B2	B2	A1	B2	B2	E5	C3
C	B2	B2	A1	A1	D4	D4	B2	B2	A1	B2	B2	A1	A1	C3	C3	E5	D4	E5	E5	E5	E5	C3
D	D4	D4	A1	A1	A1	A1	B2	B2	C3	C3	E5	E5	E5	E5	D4	B2	B2	A1	B2	B2	E5	C3
E	C3	B2	B2	D4	D4	C3	C3	E5	E5	A1	A1	B2	B2	D4	A1	A1	A1	E5	E5	B2	B2	E5
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
A	A1	A1	D4	B2	B2	D4	D4	A1	E5	C3	C3	B2	B2	A1	A1	C3	E5	E5	B2	B2	E5	E5
B	D4	D4	A1	A1	A1	B2	B2	E5	E5	E5	E5	B2	B2	A1	A1	C3	D4	B2	B2	E5	C3	C3
C	B2	B2	A1	A1	C3	A1	A1	A1	E5	E5	E5	B2	B2	D4	D4	B2	B2	D4	C3	C3	E5	E5
D	D4	D4	A1	A1	A1	B2	B2	E5	E5	E5	E5	B2	B2	A1	A1	C3	D4	B2	B2	E5	C3	C3
E	B2	B2	A1	A1	D4	D4	B2	B2	A1	B2	B2	A1	A1	C3	C3	E5	D4	E5	E5	E5	E5	C3

3.4 Evaluasi Kromosom

Evaluasi kromosom dilakukan dengan memberikan nilai penalti untuk setiap *constraints* yang dilanggar. Perbandingan nilai pinalti antara *hard constraint* dan *soft constraint* adalah 1 banding 10. Adapun perbedaan poin penalti adalah untuk menjaga agar mendahulukan *hard constraints* tanpa meninggalkan *soft constraints*.

Hard constraint merupakan batasan yang ditetapkan dimana jika batasan ini tidak terpenuhi, maka jadwal tidak dapat dilaksanakan. Adapun pada penelitian ini ada dua *hard constraint* yaitu:

1. Seorang guru mengajar di kelas yang berbeda dalam satu waktu
2. Guru ditugaskan mengajar pada hari liburnya

Soft constraints adalah batasan yang ditetapkan dimana jika batasan ini tidak terpenuhi, maka jadwal masih dapat berjalan. Adapun pada penelitian ini terdapat dua *soft constraint* yaitu:

1. Seorang guru mengajar di kelas yang sama lebih dari satu pelajaran dalam satu hari.
2. Pelajaran yang sama terjadwal dalam satu hari pada suatu kelas lebih dari tiga jam pelajaran.

Perhitungan pinalti untuk *hard constraints* pertama berupa tabrakan jadwal mengajar seorang guru dilakukan dengan memeriksa tidak ada tumpang tindih setiap guru pada setiap slot. Tahapan perhitungan bentrokan waktu mengajar dapat diselesaikan dengan prosedur sebagaimana *pseudocode* berikut:

```

For each individu do
  For each kelas do
    For each slot waktu do
      For each slot waktu do
        If kelas dan slot waktu berbeda then
          If guru sama then
            Tambahkan penalti
          End If
        End If
      End If
    End For
  End For
End For

```

Pada *pseudocode* diatas, dapat diamati proses penentuan penalti dengan cara satu individu guru dipastikan memiliki slot jadwal pada suatu kelas. Selanjutnya sistem akan membandingkan slot guru tersebut terdapat di slot kelas lain pada waktu yang sama atau tidak. Selanjutnya, jika guru tersebut terdapat slot di kelas berbeda pada waktu yang sama maka secara otomatis guru atau individu tersebut mendapatkan penalti. Namun, apabila tidak ditemukan jadwal pada kelas atau waktu yang berbeda maka guru atau individu tersebut tidak mendapatkan penalti. Berikut penerapan pada sistem dapat diamati pada Gambar 3.4.

```
// Hitung Tabrak Jadwal Guru antar kelas dalam satu gen
function tabrak($gen)
{
    $x = 0;
    for ($i=0; $i<count($gen)-1; $i++) {
        for ($j=$i; $j<count($gen); $j++) {
            if($i != $j){
                foreach ($gen[$i]['guru'] as $key => $value) {
                    if (!isset($gen[$i]['guru'][$key]['tabrak'])) {
                        $gen[$i]['guru'][$key]['tabrak_detail'] = '';
                        $gen[$i]['guru'][$key]['tabrak'] = 0;
                    }

                    if (!isset($gen[$j]['guru'][$key]['tabrak'])) {
                        $gen[$j]['guru'][$key]['tabrak_detail'] = '';
                        $gen[$j]['guru'][$key]['tabrak'] = 0;
                    }

                    if ($gen[$j]['guru'][$key]['id_guru'] == $value['id_guru']) {
                        $x++;
                        $gen[$i]['guru'][$key]['tabrak']++;
                        $gen[$i]['guru'][$key]['tabrak_detail'] .= '('. $gen[$j]['kelas']. ')';
                        $gen[$j]['guru'][$key]['tabrak']++;
                        $gen[$j]['guru'][$key]['tabrak_detail'] .= '('. $gen[$i]['kelas']. ')';
                    }
                }
            }
        }
    }
    $gen[0]['total_err_tabrak'] = $x;
    return $gen;
}
```

Gambar 3. 4 Implementasi Perhitungan Penalti Hard Constraint Tabrakan Jam Mengajar

Contoh hasil perhitungan dengan prosedur diatas pada salah satu individu generasi pertama dapat dilihat pada Tabel 3.3. Sel yang berwarna merah

menunjukkan sel dimana terjadi tabrakan jadwal mengajar seorang guru. Relatif terdapat banyak tabrakan yang terjadi pada setiap individu pada generasi pertama. Terdapat 66 tabrakan pada individu 1, 62 tabrakan pada individu 2, 95 tabrakan pada individu 3, sedangkan pada individu 4 terdapat 80 tabrakan dan pada individu 5 terdapat 79 tabrakan.

Tabel 03.3 Perhitungan pinalti tabrakan jam mengajar guru

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
A	D4	B2	B2	B2	B2	B2	B2	E5	E5	C3	C3	A1	C3	D4	D4	A1	A1	E5	A1	A1	E5	E5
B	A1	A1	D4	B2	B2	D4	D4	A1	E5	C3	C3	B2	B2	A1	A1	C3	E5	E5	B2	B2	E5	E5
C	B2	B2	C3	B2	B2	D4	B2	B2	A1	E5	E5	A1	A1	C3	C3	E5	D4	D4	A1	A1	E5	E5
D	E5	E5	E5	E5	E5	B2	B2	C3	A1	A1	B2	B2	C3	C3	A1	B2	B2	A1	A1	D4	D4	D4
E	C3	B2	B2	D4	D4	C3	C3	E5	E5	A1	A1	B2	B2	D4	A1	A1	A1	E5	E5	B2	B2	E5
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
A	C3	D4	D4	E5	E5	E5	E5	C3	C3	A1	A1	A1	B2	B2	B2	B2	E5	D4	A1	A1	B2	B2
B	E5	A1	A1	A1	C3	C3	C3	B2	B2	B2	B2	B2	B2	D4	E5	E5	E5	E5	A1	A1	D4	D4
C	C3	D4	B2	B2	A1	A1	E5	E5	E5	E5	A1	A1	C3	C3	B2	B2	E5	B2	B2	D4	D4	A1
D	D4	D4	A1	A1	A1	B2	B2	E5	E5	E5	E5	B2	B2	A1	A1	C3	D4	B2	B2	E5	C3	C3
E	B2	B2	A1	A1	C3	A1	A1	A1	E5	E5	E5	B2	B2	D4	D4	B2	B2	D4	C3	C3	E5	E5
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
A	C3	D4	D4	E5	E5	E5	E5	C3	C3	A1	A1	A1	B2	B2	B2	B2	E5	D4	A1	A1	B2	B2
B	C3	D4	B2	B2	A1	A1	E5	E5	E5	E5	A1	A1	C3	C3	B2	B2	E5	B2	B2	D4	D4	A1
C	C3	D4	D4	E5	E5	E5	E5	C3	C3	A1	A1	A1	B2	B2	B2	B2	E5	D4	A1	A1	B2	B2
D	B2	B2	C3	B2	B2	D4	B2	B2	A1	E5	E5	A1	A1	C3	C3	E5	D4	D4	A1	A1	E5	E5
E	C3	D4	B2	B2	A1	A1	E5	E5	E5	E5	A1	A1	C3	C3	B2	B2	E5	B2	B2	D4	D4	A1
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
A	E5	A1	A1	A1	C3	C3	C3	B2	B2	B2	B2	B2	B2	D4	E5	E5	E5	E5	A1	A1	D4	D4
B	D4	D4	A1	A1	A1	A1	B2	B2	C3	C3	E5	E5	E5	E5	D4	B2	B2	A1	B2	B2	E5	C3
C	B2	B2	A1	A1	D4	D4	B2	B2	A1	B2	B2	A1	A1	C3	C3	E5	D4	E5	E5	E5	E5	C3
D	D4	D4	A1	A1	A1	A1	B2	B2	C3	C3	E5	E5	E5	E5	D4	B2	B2	A1	B2	B2	E5	C3
E	C3	B2	B2	D4	D4	C3	C3	E5	E5	A1	A1	B2	B2	D4	A1	A1	A1	E5	E5	B2	B2	E5
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
A	A1	A1	D4	B2	B2	D4	D4	A1	E5	C3	C3	B2	B2	A1	A1	C3	E5	E5	B2	B2	E5	E5
B	D4	D4	A1	A1	A1	B2	B2	E5	E5	E5	E5	B2	B2	A1	A1	C3	D4	B2	B2	E5	C3	C3
C	B2	B2	A1	A1	C3	A1	A1	A1	E5	E5	E5	B2	B2	D4	D4	B2	B2	D4	C3	C3	E5	E5
D	D4	D4	A1	A1	A1	B2	B2	E5	E5	E5	E5	B2	B2	A1	A1	C3	D4	B2	B2	E5	C3	C3
E	B2	B2	A1	A1	D4	D4	B2	B2	A1	B2	B2	A1	A1	C3	C3	E5	D4	E5	E5	E5	E5	C3

Sebelum menghitung *hard constraints* kedua yaitu guru ditugaskan pada hari libur, perlu adanya data hari libur untuk masing-masing guru. Adapun

contoh data hari libur setiap guru dapat dilihat pada Tabel 3.4. Data libur guru berbentuk *array* dua dimensi dengan baris sejumlah guru dan kolom sejumlah hari aktif.

Tabel 03.4 Data ketersediaan guru berdasarkan hari

	1	2	3	4	5
A					
B					
C			1		
D				1	
E					

Kemudian perhitungan *hard constraint* dapat dilanjutkan dengan membandingkan ketersediaan setiap guru pada slot waktu yang telah terjadwalkan. Perhitungan pinalti *hard constraint* ini dapat diselesaikan dengan sebuah prosedur sebagaimana terlihat pada *pseudocode* berikut:

```

For each individu do
  For each kelas do
    For each slot waktu do
      If guru terjadwal sedang libur then
        Tambahkan penalti
      End If
    End For
  End For
End For

```

Pada *pseudocode* diatas, dapat diamati proses penentuan penalti dengan cara satu individu guru dipastikan memiliki slot jadwal pada suatu kelas. Selanjutnya sistem akan membandingkan slot guru tersebut bertepatan pada hari libur atau tidak. Selanjutnya, jika slot guru tersebut bertepatan pada jadwal libur maka akan ditambahkan penalti. Namun, apabila tidak ditemukan jadwal libur pada waktu yang sama maka tidak akan ditambahkan penalti. Berikut penerapan pada sistem dapat diamati pada Gambar 3.5.

```

You, 18 days ago • Initial Commit
// Hitung Guru yang terjadwal ketika libur -> input: gen2jam
function libur($gen)
{
    $soff = $this->model->get_libur_guru();
    $x = 0;
    foreach ($gen as $g => $kelas) {
        foreach ($soff as $key => $value) {
            if ($value['senin'] == 1) {
                $kolom = array_column($kelas['guru']['senin'], 'id_guru');
                $count = array_count_values($kolom);
                if(isset($count[$value['id_guru']]))
                    $x += $count[$value['id_guru']];
            }
            if ($value['selasa'] == 1) {
                $kolom = array_column($kelas['guru']['selasa'], 'id_guru');
                $count = array_count_values($kolom);
                if(isset($count[$value['id_guru']]))
                    $x += $count[$value['id_guru']];
            }
            if ($value['rabu'] == 1) {
                $kolom = array_column($kelas['guru']['rabu'], 'id_guru');
                $count = array_count_values($kolom);
                if(isset($count[$value['id_guru']]))
                    $x += $count[$value['id_guru']];
            }
            if ($value['kamis'] == 1) {
                $kolom = array_column($kelas['guru']['kamis'], 'id_guru');
                $count = array_count_values($kolom);
                if(isset($count[$value['id_guru']]))
                    $x += $count[$value['id_guru']];
            }
            if ($value['jumat'] == 1) {
                $kolom = array_column($kelas['guru']['jumat'], 'id_guru');
                $count = array_count_values($kolom);
                if(isset($count[$value['id_guru']]))
                    $x += $count[$value['id_guru']];
            }
        }
    }
    $gen[0]['total_err_libur'] = $x;
    return $gen;
}

```

Gambar 3. 5 Implementasi Perhitungan Pinalti Guru Terjadwal Pada Hari Libur

Hasil perhitungan *hard constraint* kedua sebagaimana pada Tabel 3.5 menunjukkan total 9 kali seorang guru mengajar ketika hari liburnya pada individu 1. Terdapat 6 pinalti untuk individu 2, 9 penalti untuk individu 3, sedangkan individu 4 menerima 4 pinalti dan 6 pinalti untuk individu 5. Sama halnya dengan contoh perhitungan sebelumnya, sel berwarna merah menunjukkan sel yang berisi seorangan guru mengajar ketika hari liburnya.

Tabel 03.5 Perhitungan penalti ketersediaan guru di hari tertentu

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
A	D4	B2	B2	B2	B2	B2	B2	E5	E5	C3	C3	A1	C3	D4	D4	A1	A1	E5	A1	A1	E5	E5
B	A1	A1	D4	B2	B2	D4	D4	A1	E5	C3	C3	B2	B2	A1	A1	C3	E5	E5	B2	B2	E5	E5
C	B2	B2	C3	B2	B2	D4	B2	B2	A1	E5	E5	A1	A1	C3	C3	E5	D4	D4	A1	A1	E5	E5
D	E5	E5	E5	E5	E5	B2	B2	C3	A1	A1	B2	B2	C3	C3	A1	B2	B2	A1	A1	D4	D4	D4
E	C3	B2	B2	D4	D4	C3	C3	E5	E5	A1	A1	B2	B2	D4	A1	A1	A1	E5	E5	B2	B2	E5
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
A	C3	D4	D4	E5	E5	E5	E5	C3	C3	A1	A1	A1	B2	B2	B2	B2	E5	D4	A1	A1	B2	B2
B	E5	A1	A1	A1	C3	C3	C3	B2	B2	B2	B2	B2	B2	D4	E5	E5	E5	E5	A1	A1	D4	D4
C	C3	D4	B2	B2	A1	A1	E5	E5	E5	E5	A1	A1	C3	C3	B2	B2	E5	B2	B2	D4	D4	A1
D	D4	D4	A1	A1	A1	B2	B2	E5	E5	E5	E5	B2	B2	A1	A1	C3	D4	B2	B2	E5	C3	C3
E	B2	B2	A1	A1	C3	A1	A1	A1	E5	E5	E5	B2	B2	D4	D4	B2	B2	D4	C3	C3	E5	E5
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
A	C3	D4	D4	E5	E5	E5	E5	C3	C3	A1	A1	A1	B2	B2	B2	B2	E5	D4	A1	A1	B2	B2
B	C3	D4	B2	B2	A1	A1	E5	E5	E5	E5	A1	A1	C3	C3	B2	B2	E5	B2	B2	D4	D4	A1
C	C3	D4	D4	E5	E5	E5	E5	C3	C3	A1	A1	A1	B2	B2	B2	B2	E5	D4	A1	A1	B2	B2
D	B2	B2	C3	B2	B2	D4	B2	B2	A1	E5	E5	A1	A1	C3	C3	E5	D4	D4	A1	A1	E5	E5
E	C3	D4	B2	B2	A1	A1	E5	E5	E5	E5	A1	A1	C3	C3	B2	B2	E5	B2	B2	D4	D4	A1
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
A	E5	A1	A1	A1	C3	C3	C3	B2	B2	B2	B2	B2	B2	D4	E5	E5	E5	E5	A1	A1	D4	D4
B	D4	D4	A1	A1	A1	A1	B2	B2	C3	C3	E5	E5	E5	E5	D4	B2	B2	A1	B2	B2	E5	C3
C	B2	B2	A1	A1	D4	D4	B2	B2	A1	B2	B2	A1	A1	C3	C3	E5	D4	E5	E5	E5	E5	C3
D	D4	D4	A1	A1	A1	A1	B2	B2	C3	C3	E5	E5	E5	E5	D4	B2	B2	A1	B2	B2	E5	C3
E	C3	B2	B2	D4	D4	C3	C3	E5	E5	A1	A1	B2	B2	D4	A1	A1	A1	E5	E5	B2	B2	E5
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
A	A1	A1	D4	B2	B2	D4	D4	A1	E5	C3	C3	B2	B2	A1	A1	C3	E5	E5	B2	B2	E5	E5
B	D4	D4	A1	A1	A1	B2	B2	E5	E5	E5	E5	B2	B2	A1	A1	C3	D4	B2	B2	E5	C3	C3
C	B2	B2	A1	A1	C3	A1	A1	A1	E5	E5	E5	B2	B2	D4	D4	B2	B2	D4	C3	C3	E5	E5
D	D4	D4	A1	A1	A1	B2	B2	E5	E5	E5	E5	B2	B2	A1	A1	C3	D4	B2	B2	E5	C3	C3
E	B2	B2	A1	A1	D4	D4	B2	B2	A1	B2	B2	A1	A1	C3	C3	E5	D4	E5	E5	E5	E5	C3

Selanjutnya contoh perhitungan *soft constraint* berupa seorang guru mengajar di kelas yang sama dilakukan dengan membandingkan setiap guru yang mengajar pada satu kelas dalam hari tertentu. Perhitungan pinalti untuk *soft constraint* ini dapat diselesaikan dengan prosedur sebagaimana digambarkan dengan *pseudocode* berikut:

```

For each individu do
  For each kelas do
    For each slot waktu do
      For each slot waktu di hari yang sama do
        If guru terjadwal sama then
          Tambahkan pinalti
        End If
      End for
    End For
  End For
End For

```

Pada *pseudocode* diatas, dapat diamati proses penentuan penalti dengan cara satu individu guru dipastikan memiliki slot jadwal pada suatu kelas. Selanjutnya sistem akan membandingkan slot guru tersebut dengan slot waktu pada hari yang sama. Selanjutnya, jika guru tersebut terdapat slot jadwal yang sama maka guru atau individu tersebut akan secara otomatis ditambahkan penalti. Berikut penerapan pada sistem dapat diamati pada Gambar 3.6.



```

You, 18 days ago • Initial Commit
// Hitung Guru yang terjadwal dalam satu hari -> input: gen2jam
function guru($gen)
{
    $x = 0;
    foreach ($gen as $key => $value) {
        $x1 = 0;
        foreach ($value['guru'] as $k => $val) {
            $kolom = array_column($val, 'id_guru');
            $count = array_count_values($kolom);
            $gen[$key]['guru'][$k][0]['err_guru_sehari'] = 0;
            foreach ($count as $c => $v) {
                if ($v > 3) {
                    $x += $v;
                    $x1 += $v;
                    $gen[$key]['guru'][$k][0]['err_guru_sehari'] += $v;
                }
            }
        }
        if ($gen[$key]['guru'][$k][0]['err_guru_sehari'] == 0){
            unset($gen[$key]['guru'][$k][0]['err_guru_sehari']);
        }
        $gen[$key]['err_guru_sekelas'] = $x1;
    }
    $gen[0]['total_err_guru'] = $x;
    return $gen;
}

```

Gambar 3. 6 Implementasi Perhitungan Pinalti Guru Terjadwal pada Kelas yang Sama

Hasil perhitungan sebagai contoh dapat diamati pada Tabel 3.6. Sebagaimana sebelumnya, sel berwarna merah menunjukkan bentuk pelanggaran yang terjadi. Terlihat terdapat 13 kesempatan seorang guru mengajar di kelas yang sama dalam sehari pada individu 1, pada individu 2 terdapat 4 pinalti, dan masing masing 12 penalti untuk individu 3 dan 4. Sedangkan individu 5 tidak mendapatkan penalti.

Tabel 03.6 Perhitungan pinalti guru yang mengajar kelas yang sama dalam sehari

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
A	D4	B2	B2	B2	B2	B2	B2	E5	E5	C3	C3	A1	C3	D4	D4	A1	A1	E5	A1	A1	E5	E5
B	A1	A1	D4	B2	B2	D4	D4	A1	E5	C3	C3	B2	B2	A1	A1	C3	E5	E5	B2	B2	E5	E5
C	B2	B2	C3	B2	B2	D4	B2	B2	A1	E5	E5	A1	A1	C3	C3	E5	D4	D4	A1	A1	E5	E5
D	E5	E5	E5	E5	E5	B2	B2	C3	A1	A1	B2	B2	C3	C3	A1	B2	B2	A1	A1	D4	D4	D4
E	C3	B2	B2	D4	D4	C3	C3	E5	E5	A1	A1	B2	B2	D4	A1	A1	A1	E5	E5	B2	B2	E5
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
A	C3	D4	D4	E5	E5	E5	E5	C3	C3	A1	A1	A1	B2	B2	B2	B2	E5	D4	A1	A1	B2	B2
B	E5	A1	A1	A1	C3	C3	C3	B2	B2	B2	B2	B2	B2	D4	E5	E5	E5	E5	A1	A1	D4	D4
C	C3	D4	B2	B2	A1	A1	E5	E5	E5	E5	A1	A1	C3	C3	B2	B2	E5	B2	B2	D4	D4	A1
D	D4	D4	A1	A1	A1	B2	B2	E5	E5	E5	E5	B2	B2	A1	A1	C3	D4	B2	B2	E5	C3	C3
E	B2	B2	A1	A1	C3	A1	A1	A1	E5	E5	E5	B2	B2	D4	D4	B2	B2	D4	C3	C3	E5	E5
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
A	C3	D4	D4	E5	E5	E5	E5	C3	C3	A1	A1	A1	B2	B2	B2	B2	E5	D4	A1	A1	B2	B2
B	C3	D4	B2	B2	A1	A1	E5	E5	E5	E5	A1	A1	C3	C3	B2	B2	E5	B2	B2	D4	D4	A1
C	C3	D4	D4	E5	E5	E5	E5	C3	C3	A1	A1	A1	B2	B2	B2	B2	E5	D4	A1	A1	B2	B2
D	B2	B2	C3	B2	B2	D4	B2	B2	A1	E5	E5	A1	A1	C3	C3	E5	D4	D4	A1	A1	E5	E5
E	C3	D4	B2	B2	A1	A1	E5	E5	E5	E5	A1	A1	C3	C3	B2	B2	E5	B2	B2	D4	D4	A1
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
A	E5	A1	A1	A1	C3	C3	C3	B2	B2	B2	B2	B2	B2	D4	E5	E5	E5	E5	A1	A1	D4	D4
B	D4	D4	A1	A1	A1	A1	B2	B2	C3	C3	E5	E5	E5	E5	D4	B2	B2	A1	B2	B2	E5	C3
C	B2	B2	A1	A1	D4	D4	B2	B2	A1	B2	B2	A1	A1	C3	C3	E5	D4	E5	E5	E5	E5	C3
D	D4	D4	A1	A1	A1	A1	B2	B2	C3	C3	E5	E5	E5	E5	D4	B2	B2	A1	B2	B2	E5	C3
E	C3	B2	B2	D4	D4	C3	C3	E5	E5	A1	A1	B2	B2	D4	A1	A1	A1	E5	E5	B2	B2	E5
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
A	A1	A1	D4	B2	B2	D4	D4	A1	E5	C3	C3	B2	B2	A1	A1	C3	E5	E5	B2	B2	E5	E5
B	D4	D4	A1	A1	A1	B2	B2	E5	E5	E5	E5	B2	B2	A1	A1	C3	D4	B2	B2	E5	C3	C3
C	B2	B2	A1	A1	C3	A1	A1	A1	E5	E5	E5	B2	B2	D4	D4	B2	B2	D4	C3	C3	E5	E5
D	D4	D4	A1	A1	A1	B2	B2	E5	E5	E5	E5	B2	B2	A1	A1	C3	D4	B2	B2	E5	C3	C3
E	B2	B2	A1	A1	D4	D4	B2	B2	A1	B2	B2	A1	A1	C3	C3	E5	D4	E5	E5	E5	E5	C3

Terakhir adalah menghitung penalti untuk *soft constraint* berupa pelajaran yang sama terjadwal dalam satu hari. *Soft constraint* ini dapat dihitung dengan cara hampir sama dengan menghitung *soft constraint* sebelumnya. Yaitu dengan membandingkan pelajaran dalam satu hari. Pseudocode berikut menggambarkan prosedur perhitungan penalti untuk pelajaran sama terjadwal dalam satu hari.

```

For each individu do
  For each kelas do
    For each slot waktu do
      For each slot waktu di hari yang sama do
        If pelajaran lebih dari 3 jam pelajaran then
          Tambahkan pinalti sejumlah pelajaran jam pelajaran yang
          terjadwal
        End If
      End for
    End For
  End For
End For

```

Pada *pseudocode* diatas, dapat diamati proses penentuan penalti dengan cara satu individu guru dipastikan memiliki slot jadwal pada suatu kelas. Selanjutnya pada slot mata pelajaran kelas tersebut sistem memindai adakah 3 jam dengan mata pelajaran yang sama atau tidak. Jika terdapat pelajaran yang sama pinalti akan secara otomatis ditambahkan sejumlah jam pelajaran yang terjadwal. Implementasi *pseudocode* tersebut dapat dilihat pada Gambar 3.7.

```

You, 18 days ago • Initial Commit
// Hitung Pelajaran yang terjadwal dalam satu hari -> input: gen2jam
function pelajaran($gen)
{
    $x = 0;
    foreach ($gen as $key => $value) {
        $x1 = 0;
        foreach ($value['guru'] as $k => $val) {
            $kolom = array_column($val, 'id_pelajaran');
            $count = array_count_values($kolom);
            $gen[$key]['guru'][$k][0]['err_pelajaran_sehari'] = 0;
            foreach ($count as $c => $v) {
                if ($v > 3) {
                    $x += $v;
                    $x1 += $v;
                    $gen[$key]['guru'][$k][0]['err_pelajaran_sehari'] += $v;
                }
            }
            if ($gen[$key]['guru'][$k][0]['err_pelajaran_sehari'] == 0){
                unset($gen[$key]['guru'][$k][0]['err_pelajaran_sehari']);
            }
        }
        $gen[$key]['err_pelajaran_sekelas'] = $x1;
    }
    $gen[0]['total_err_pelajaran'] = $x;
    return $gen;
}

```

Gambar 3. 7 Implementasi Perhitungan Pinalti Pelajaran

Hasil perhitungan pelajaran sama yang terjadwal dihari yang sama dapat diamati pada Tabel 3.7. Terlihat bahwa perhitungan untuk *soft constraint* ini sama dengan sebelumnya, hal ini karena satu guru hanya mengajar satu pelajaran. Perhitungan akan berbeda jika satu guru mengajar beberapa pelajaran. Hasil perhitungan pelajaran yang terjadwal di hari yang sama mulai dari individu 1 hingga individu 5 secara berurutan adalah 13, 4, 12, 12 dan 0.

Tabel 03.7 Perhitungan pinalti pelajaran yang terjadwal di hari yang sama

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
A	D4	B2	B2	B2	B2	B2	B2	E5	E5	C3	C3	A1	C3	D4	D4	A1	A1	E5	A1	A1	E5	E5
B	A1	A1	D4	B2	B2	D4	D4	A1	E5	C3	C3	B2	B2	A1	A1	C3	E5	E5	B2	B2	E5	E5
C	B2	B2	C3	B2	B2	D4	B2	B2	A1	E5	E5	A1	A1	C3	C3	E5	D4	D4	A1	A1	E5	E5
D	E5	E5	E5	E5	E5	B2	B2	C3	A1	A1	B2	B2	C3	C3	A1	B2	B2	A1	A1	D4	D4	D4
E	C3	B2	B2	D4	D4	C3	C3	E5	E5	A1	A1	B2	B2	D4	D4	A1	A1	E5	E5	B2	B2	E5
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
A	C3	D4	D4	E5	E5	E5	E5	C3	C3	A1	A1	A1	B2	B2	B2	B2	E5	D4	A1	A1	B2	B2
B	E5	A1	A1	A1	C3	C3	C3	B2	B2	B2	B2	B2	B2	D4	E5	E5	E5	E5	A1	A1	D4	D4
C	C3	D4	B2	B2	A1	A1	E5	E5	E5	E5	A1	A1	C3	C3	B2	B2	E5	B2	B2	D4	D4	A1
D	D4	D4	A1	A1	A1	B2	B2	E5	E5	E5	E5	B2	B2	A1	A1	C3	D4	B2	B2	E5	C3	C3
E	B2	B2	A1	A1	C3	A1	A1	A1	E5	E5	E5	B2	B2	D4	D4	B2	B2	D4	C3	C3	E5	E5
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
A	C3	D4	D4	E5	E5	E5	E5	C3	C3	A1	A1	A1	B2	B2	B2	B2	E5	D4	A1	A1	B2	B2
B	C3	D4	B2	B2	A1	A1	E5	E5	E5	E5	A1	A1	C3	C3	B2	B2	E5	B2	B2	D4	D4	A1
C	C3	D4	D4	E5	E5	E5	E5	C3	C3	A1	A1	A1	B2	B2	B2	B2	E5	D4	A1	A1	B2	B2
D	B2	B2	C3	B2	B2	D4	B2	B2	A1	E5	E5	A1	A1	C3	C3	E5	D4	D4	A1	A1	E5	E5
E	C3	D4	B2	B2	A1	A1	E5	E5	E5	E5	A1	A1	C3	C3	B2	B2	E5	B2	B2	D4	D4	A1
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
A	E5	A1	A1	A1	C3	C3	C3	B2	B2	B2	B2	B2	B2	D4	E5	E5	E5	E5	A1	A1	D4	D4
B	D4	D4	A1	A1	A1	A1	B2	B2	C3	C3	E5	E5	E5	E5	D4	B2	B2	A1	B2	B2	E5	C3
C	B2	B2	A1	A1	D4	D4	B2	B2	A1	B2	B2	A1	A1	C3	C3	E5	D4	E5	E5	E5	E5	C3
D	D4	D4	A1	A1	A1	A1	B2	B2	C3	C3	E5	E5	E5	E5	D4	B2	B2	A1	B2	B2	E5	C3
E	C3	B2	B2	D4	D4	C3	C3	E5	E5	A1	A1	B2	B2	D4	A1	A1	A1	E5	E5	B2	B2	E5
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
A	A1	A1	D4	B2	B2	D4	D4	A1	E5	C3	C3	B2	B2	A1	A1	C3	E5	E5	B2	B2	E5	E5
B	D4	D4	A1	A1	A1	B2	B2	E5	E5	E5	E5	B2	B2	A1	A1	C3	D4	B2	B2	E5	C3	C3
C	B2	B2	A1	A1	C3	A1	A1	A1	E5	E5	E5	B2	B2	D4	D4	B2	B2	D4	C3	C3	E5	E5
D	D4	D4	A1	A1	A1	B2	B2	E5	E5	E5	E5	B2	B2	A1	A1	C3	D4	B2	B2	E5	C3	C3
E	B2	B2	A1	A1	D4	D4	B2	B2	A1	B2	B2	A1	A1	C3	C3	E5	D4	E5	E5	E5	E5	C3

3.5 Perhitungan *Fitness*

Proses selanjutnya setelah menghitung penalti untuk setiap *constraint* bagi setiap individu adalah menentukan nilai *fitness*. Perhitungan nilai *fitness* dilakukan dengan membagi total penalti individu dengan total penalti seluruh individu dalam satu generasi. Secara matematis perhitungan nilai *fitness* dituliskan dengan rumus (1)(Saputra, 2019).

$$F = \frac{1}{h \times (PT+PL) + s \times (PG+PP)} \dots \dots \dots (1)$$

Keterangan:

F : Nilai *fitness*

h : Bobot *hard constraint*

s : Bobot *soft constraint*

PT : Jumlah penalti tabrakan jam mengajar

PL : Jumlah penalti guru libur

PG : Jumlah penalti guru yang sama terjadwal dalam satu hari

PP : Jumlah penalti pelajaran yang sama terjadwal dalam satu hari

Melanjutkan tahapan selanjutnya, keseluruhan penalti yang diterima individu 1 adalah sebagai berikut: 66 pinalti tabrakan jam pelajaran, 9 pinalti guru libur, 13 penalti guru yang terjadwal dalam satu hari dan 13 pinalti pelajaran yang terjadwal dalam satu hari. Penyebut pada rumus penentuan nilai *fitness* didapat dari penjumlahan nilai pinalti dikalikan dengan bobot *hard constraint* senilai 0.01 dan *soft constraint* senilai 0.001, sehingga didapatkan perhitungan sebagai berikut:

$$F = \frac{1}{1 \times (66 + 9) + 0.1 \times (13 + 13)}$$

$$F = \frac{1}{77,6}$$

$$F = 0,0128865$$

Dengan melakukan perhitungan yang sama, didapatkan nilai *fitness* untuk individu lainnya secara berurutan dari individu 2 hingga individu 5 sebagai berikut: 0,014534884, 0,009398496, 0,011574074 dan 0,011764706. Adapun implementasi perhitungan *fitness* pada pemrograman PHP dapat diamati pada Gambar 3.8.

```

$TF = 0;
// Hitung nilai fitness
$fitness = 0;
foreach ($arr as $key => $value) {
    $arr[$key][0]['fitnes'] = 1 / (1 +
    ($value[0]['total_err_tabrak'] * 0.01+
    $value[0]['total_err_libur'] * 0.01 +
    $value[0]['total_err_pelajaran'] * 0.001 +
    $value[0]['total_err_guru'] * 0.001 ));
    $TF += $arr[$key][0]['fitnes'];
    if ($arr[$key][0]['fitnes'] > $fitness) {
        $best = $key;
    }
}

```

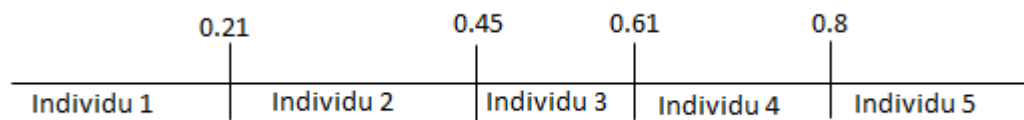
Gambar 3. 8 Implementasi Perhitungan Fitness

3.6 Seleksi

Proses seleksi adalah proses memilih beberapa kromosom guna menciptakan populasi tengah atau *mating pool* untuk reproduksi. Adapun metode yang digunakan pada penelitian ini adalah metode seleksi *roulette wheel* dan *elitism*. Berikut langkah seleksi *roulette wheel*:

1. Memetakan setiap individu ke segmen-segmen tertentu hingga membentuk suatu garis demikian rupa sehingga setiap individu memiliki ukuran yang sama dengan nilai *fitnessnya*.
2. Ulangi langkah (a) dan (b) hingga jumlah individu untuk populasi kawin yang diinginkan diperoleh
 - a. Hasilkan angka acak dari nol hingga satu
 - b. Pilih individu yang segmennya mencakup angka acak

Individu 2 adalah individu terbaik, sehingga memiliki interval terbesar dibandingkan dengan individu 3 dengan interval terkecil (Gambar 3.9). Sebagai contoh, diasumsikan bahwa 5 angka acak yang muncul adalah 0,59, 0,22, 0,17, 0,5 dan 0.043. Sehingga individu yang terpilih ke proses reproduksi adalah Individu 3, Individu 2, Individu 1, Individu 3 dan Individu 1.



Gambar 3. 9 Contoh Roulette Wheel

Setelah melakukan *roulette wheel* terbentuk populasi baru dengan jumlah individu yang sama dengan populasi sebelumnya. Populasi baru inilah yang akan mengalami proses genetika *crossover* dan mutasi untuk menghasilkan generasi baru. Diharapkan dengan proses seleksi dapat mengeliminasi individu dengan *fitness* kurang baik dan mempertahankan individu dengan *fitness* baik.

Implementasi proses *roulette wheel* dapat diamati pada Gambar 3.10

```
// Fungsi untuk seleksi gen, mengembalikan gen terpilih sejumlah sama dengan jumlah gen awal
// Input bentuk jam
function seleksi($arr)
{
    $TF = 0;
    // Hitung nilai fitness
    $fitness = 0;
    foreach ($arr as $key => $value) { ...
    }

    // Cari gen terbaik
    if ($this->gen_terbaik == []) { ...
    } else { ...
    }

    // Wheel untuk bagian setiap gen
    $wheel = 0;
    foreach ($arr as $key => $value) {
        $arr[$key][0]['probabilitas'] = $arr[$key][0]['fitnes'] / $TF;
        $arr[$key][0]['kode_gen'] = $key;
        $wheel += $arr[$key][0]['probabilitas'];
        $arr[$key][0]['wheel'] = $wheel;
    }
    // Rorretwheel sejumlah gen
    for ($j=0; $j<count($arr); $j++) {
        $r = rand(1, 100)/100;
        $x = 0;
        for ($i=1; $i<count($arr); $i++) {
            if ($r <= $arr[$i][0]['wheel'] AND $r > $arr[$i-1][0]['wheel']) {
                $x = $i;
            }
        }
        $new_arr[$j] = $arr[$x];
    }
    // Elitism
    $new_arr[rand(0, count($new_arr)-1)] = $arr[$best];
    return $new_arr;
}
```

Gambar 3. 10 Implementasi Proses Roulette Wheel

3.7 Crossover

Crossover adalah proses rekombinasi untuk menciptakan individu baru dengan harapan meningkatnya nilai *fitness*. Secara umum *crossover* dilakukan dengan cara memilih secara acak sepasang individu dari generasi tengah, kemudian membentuk individu baru dengan mencampurkan atau menyilangkan gen antara dua individu.

Berikut langkah *crossover* yang digunakan dalam penelitian ini:

1. Ulangi langkah 2 dan 3 sesuai dengan *crossover rate* yang telah ditentukan
2. Pilih secara acak dua individu, sebagai contoh Individu 2 dan Individu 3 (Gambar 3.11)
3. Hasilkan individu baru dari kedua *parent* dengan prosedur berikut:
 - a. Untuk setiap slot waktu, bangkitkan angka acak dari nol hingga satu
 - b. Jika nilai angka acak lebih besar dari 0,5 maka individu baru akan mempertahankan slot waktu dari *parent*, sedangkan jika nilai angka acak kurang dari 0,5 maka individu baru akan menggunakan slot waktu dari *parent* lainnya.

Proses *crossover* di atas tidak menimbulkan masalah terkait ketersediaan guru di suatu kelas. Persilangan hanya terjadi pada slot waktu dan tidak terjadi persilangan antar kelas. Pengaturan sedemikian adalah untuk menjaga agar tidak ada guru yang mengajar pada kelas yang tidak seharusnya. Perhatikan Tabel 3.8 sebagai contoh proses *crossover*.

Tabel 03.8 Proses Crossover Individu 1 dan Individu 2

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
A	C3	D4	D4	E5	E5	E5	E5	C3	C3	A1	A1	A1	B2	B2	B2	B2	E5	D4	A1	A1	B2	B2
B	E5	A1	A1	A1	C3	C3	C3	B2	B2	B2	B2	B2	B2	D4	E5	E5	E5	E5	A1	A1	D4	D4
C	C3	D4	B2	B2	A1	A1	E5	E5	E5	E5	A1	A1	C3	C3	B2	B2	E5	B2	B2	D4	D4	A1
D	D4	D4	A1	A1	A1	B2	B2	E5	E5	E5	E5	B2	B2	A1	A1	C3	D4	B2	B2	E5	C3	C3
E	B2	B2	A1	A1	C3	A1	A1	A1	E5	E5	E5	B2	B2	D4	D4	B2	B2	D4	C3	C3	E5	E5
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
A	C3	D4	D4	E5	E5	E5	E5	C3	C3	A1	A1	A1	B2	B2	B2	B2	E5	D4	A1	A1	B2	B2
B	C3	D4	B2	B2	A1	A1	E5	E5	E5	E5	A1	A1	C3	C3	B2	B2	E5	B2	B2	D4	D4	A1
C	C3	D4	D4	E5	E5	E5	E5	C3	C3	A1	A1	A1	B2	B2	B2	B2	E5	D4	A1	A1	B2	B2
D	B2	B2	C3	B2	B2	D4	B2	B2	A1	E5	E5	A1	A1	C3	C3	E5	D4	D4	A1	A1	E5	E5
E	C3	D4	B2	B2	A1	A1	E5	E5	E5	E5	A1	A1	C3	C3	B2	B2	E5	B2	B2	D4	D4	A1
R	0,8	0,7	0,1	0,4	0,3	0,4	0,8	0,8	0,9	1	0,9	1	0,6	0,2	0,3	0,5	0,1	0,2	0,5	0,8	0,8	0,3
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
A	C3	D4	D4	E5	E5	E5	E5	C3	C3	A1	A1	A1	B2	B2	B2	B2	E5	D4	A1	A1	B2	B2
B	E5	A1	B2	B2	A1	A1	C3	B2	B2	B2	B2	B2	B2	C3	B2	B2	E5	B2	B2	D4	D4	A1
C	C3	D4	D4	E5	E5	E5	E5	E5	E5	E5	A1	A1	C3	B2	B2	B2	E5	D4	A1	A1	B2	B2
D	D4	D4	C3	B2	B2	D4	B2	E5	E5	E5	E5	B2	B2	C3	C3	E5	D4	D4	A1	A1	E5	E5
E	B2	B2	B2	B2	A1	A1	A1	A1	E5	E5	E5	B2	B2	C3	B2	B2	E5	B2	B2	D4	D4	A1

Proses *crossover* dimulai dengan membangkitkan nilai *random* untuk setiap slot waktu sebagaimana terlihat pada Gambar 3.13 dibagian bawah Individu 2. Kemudian hasil *crossover* atau *child* akan mempertahankan slot waktu dari Individu 2 dengan nilai *random* lebih besar dari 0,5 dan menggunakan slot waktu dari Individu 3 untuk menggantikan slot waktu dengan nilai *random* lebih kecil dari 0,5. *Child* mempertahankan slot waktu 1, 2, 7, 8, 9, 10, 11, 12, 13, 20 dan 21 dari Individu 2 dan menggunakan slot lainnya dari Individu 3. Slot waktu berwarna biru pada *child* menunjukkan slot waktu dari Individu 3. Implementasi proses *crossover* dapat diamati pada Gambar 3.11.

```

You, 17 days ago • crossover
// Mengembalikan gen baru hasil crossover dua induk, Bentuk Jam
// Dipotong sesuai kelas, satu kelas ikut sekaligus
function crossover($gen1, $gen2)
{
    $count = count($gen1);

    for ($i=0; $i < $count; $i++) {
        $rand = rand(0, 100);
        if ($rand > 50) {
            $new[$i] = $gen1[$i];
        } else {
            $new[$i] = $gen2[$i];
        }
    }
    return $new;
}

```

Gambar 3. 11 Implementasi Proses Crossover

3.8 Mutasi

Mutasi berperan untuk menyegarkan setiap populasi guna menjaga keberagaman solusi yang mungkin. Proses mutasi yang digunakan dalam penelitian ini adalah mutasi slot waktu di dalam kelas. Sebagaimana proses *crossover*, prosedur mutasi juga dirancang agar tidak menimbulkan masalah pada ketersediaan guru. Adapun proses mutasi yang disarankan adalah sebagai berikut:

1. Untuk setiap individu populasi saat ini, lakukan langkah-langkah berikut:
 - a. Untuk setiap kelas dari setiap individu, jalankan prosedur berikut:
 - Hasilkan angka acak antara nol hingga satu
 - Jika angka acak ini kurang dari probabilitas mutasi, maka pilih secara acak dua slot waktu yang berbeda dan tukar.

Proses mutasi dapat diamati pada Tabel 3.9. Untuk setiap kelas dalam individu yang telah dihasilkan dari proses *crossover* diberikan nilai *random* sebagaimana terlihat pada Tabel 3.9 bagian kanan setiap individu. Kemudian untuk

setiap kelas yang memiliki nilai dibawah 0,5, pilih dua guru secara acak untuk saling ditukar. Sel berwarna biru menunjukkan posisi dua guru yang ditukar.

Child pada Tabel 3.9 memiliki 3 kelas yang bermutasi. Pada kelas B, guru A1 pada slot 5 ditukar dengan guru C3 pada slot 14. Demikian halnya pada kelas C slot bermutasi ditukar adalah slot 9 dan slot 18 yang berisikan guru E5 dan D4. Dilanjutkan kelas D, slot bermutasi adalah slot 2 dan slot 12 yang berisikan guru D4 dan B2. Implementasi proses mutasi dapat diamati pada Gambar 3.12.

Tabel 03.9 Proses Mutasi

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	
A	C3	D4	D4	E5	E5	E5	E5	C3	C3	A1	A1	A1	B2	B2	B2	B2	E5	D4	A1	A1	B2	B2	
B	E5	A1	B2	B2	A1	A1	C3	B2	B2	B2	B2	B2	B2	C3	B2	B2	E5	B2	B2	D4	D4	A1	
C	C3	D4	D4	E5	E5	E5	E5	E5	E5	A1	A1	C3	B2	B2	B2	B2	E5	D4	A1	A1	B2	B2	
D	D4	D4	C3	B2	B2	D4	B2	E5	E5	E5	E5	B2	B2	C3	C3	E5	D4	D4	A1	A1	E5	E5	
E	B2	B2	B2	B2	A1	A1	A1	A1	E5	E5	E5	B2	B2	C3	B2	B2	E5	B2	B2	D4	D4	A1	

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	R
A	C3	D4	D4	E5	E5	E5	E5	C3	C3	A1	A1	A1	B2	B2	B2	B2	E5	D4	A1	A1	B2	B2	0,57
B	E5	A1	B2	B2	C3	A1	C3	B2	B2	B2	B2	B2	B2	A1	B2	B2	E5	B2	B2	D4	D4	A1	0,27
C	C3	D4	D4	E5	E5	E5	E5	E5	D4	E5	A1	A1	C3	B2	B2	B2	E5	E5	A1	A1	B2	B2	0,3
D	D4	B2	C3	B2	B2	D4	B2	E5	E5	E5	E5	D4	B2	C3	C3	E5	D4	D4	A1	A1	E5	E5	0,18
E	B2	B2	B2	B2	A1	A1	A1	A1	E5	E5	E5	B2	B2	C3	B2	B2	E5	B2	B2	D4	D4	A1	0,88

```
// Mutasi sekali dari seluruh populasi, bentuk Arr
function mutasi($populasi)
{
    $pop = count($populasi);
    $kelas = count($populasi[0]);
    $pelajaran = count($populasi[0][0]['guru']);
    $tot = $pop * $kelas * $pelajaran;

    $ulang = true;
    while ($ulang) {
        $rand = rand(0, $tot-1);
        $genKe = intval($rand/($pelajaran*$kelas));
        $sisagen = $rand % ($pelajaran*$kelas);
        $kelasKe = intval($sisagen / $pelajaran);
        $pelajaranKe = $sisagen % $pelajaran;

        // You, 16 days ago • mutasi
        if ($populasi[$genKe][$kelasKe]['guru'][$pelajaranKe]['jam_pelajaran'] == 2){
            $ulang = false;
        }
    }

    $sudahTukar = false;
    while (!$sudahTukar){
        $tukar = rand(0, $pelajaran-1);
        if ($populasi[$genKe][$kelasKe]['guru'][$pelajaranKe]['jam_pelajaran']
            == $populasi[$genKe][$kelasKe]['guru'][$tukar]['jam_pelajaran'])
        {
            $temp = $populasi[$genKe][$kelasKe]['guru'][$pelajaranKe];
            $populasi[$genKe][$kelasKe]['guru'][$pelajaranKe] = $populasi[$genKe][$kelasKe]['guru'][$tukar];
            $populasi[$genKe][$kelasKe]['guru'][$tukar] = $temp;
            $sudahTukar = true;
        }
    }

    return $populasi;
}
```

Gambar 3. 12 Implementasi Proses Mutasi

3.9 Skema Elitisme

Guna melestarikan individu terbaik di setiap generasi, digunakan skema *elitism* sederhana. Individu terbaik dari setiap generasi disalin ke generasi selanjutnya untuk memastikan bahwa individu terbaik dalam setiap generasi akan lebih baik dari generasi sebelumnya, atau setidaknya sama baiknya. Selain itu untuk mencegah algoritma dari menghasilkan solusi yang prematur, skema *elitism* yang diajukan hanya diterapkan pada satu kromosom dari setiap generasi. Individu terbaik pada generasi pertama adalah Individu 2 dengan nilai *fitness* 0.2416, sehingga tetap dipertahankan pada generasi selanjutnya. Sedangkan untuk 4 individu lain adalah 4 individu dari hasil *crossover* dan mutasi.

3.10 Skenario Pengujian

Pengujian dilakukan dengan mengubah 3 variabel algoritma genetika, yaitu: *crossover rate*, *mutasi rate* dan jumlah populasi. Adapun skenario pengujian dilakukan dengan melakukan pengujian pada kombinasi *crossover rate* dan *mutasi rate* terlebih dahulu dengan jumlah populasi 10. Kemudian dilanjutkan dengan pengujian pada jumlah populasi dengan menggunakan kombinasi *crossover rate* dan *mutasi rate* terbaik dari hasil pengujian sebelumnya.

Pengujian pertama dilakukan dengan *crossover rate* dan *mutasi rate* seragam dengan selisih 0,1. Terdapat sepuluh skenario pada pengujian pertama ini. Tabel 3.10 menunjukkan 10 skenario pengujian tersebut. Untuk setiap skenario dilakukan 10 percobaan dan akan diambil nilai rata-rata sebagai data pembandingan.

Tabel 03.10 Skenario uji coba crossover rate dan mutasi rate seragam

Skenario	Crossover Rate	Mutasi Rate	Populasi
Skenario 1	0,1	0,1	10
Skenario 2	0,2	0,2	10
Skenario 3	0,3	0,3	10
Skenario 4	0,4	0,4	10
Skenario 5	0,5	0,5	10
Skenario 6	0,6	0,6	10
Skenario 7	0,7	0,7	10
Skenario 8	0,8	0,8	10
Skenario 9	0,9	0,9	10
Skenario 10	1	1	10

Pengujian kedua dilakukan dengan menetapkan *crossover rate* terbaik yang didapatkan pada pengujian sebelumnya dengan mengubah *mutasi rate* pada setiap skenario pengujian. Kemudian dilanjutkan dengan menetapkan *mutasi rate* terbaik yang didapatkan pada pengujian sebelumnya dengan mengubah *crossover rate* pada setiap skenario pengujian. Dengan skema demikian terdapat 20 skenario pengujian sebagaimana dapat diamati pada Tabel 3.11 dengan permisalan *crossover rate* dan *mutasi rate* terbaik dari pengujian sebelumnya adalah 0,5.

Tabel 03.11 Skenario pengujian dengan crossover rate dan mutasi rate tidak sama

Skenario	Crossover Rate	Mutasi Rate	Populasi
Skenario 1	0,5	0,1	10
Skenario 2	0,5	0,2	10
Skenario 3	0,5	0,3	10
Skenario 4	0,5	0,4	10
Skenario 5	0,5	0,5	10
Skenario 6	0,5	0,6	10
Skenario 7	0,5	0,7	10
Skenario 8	0,5	0,8	10
Skenario 9	0,5	0,9	10
Skenario 10	0,5	1	10
Skenario 11	0,1	0,5	10
Skenario 12	0,2	0,5	10
Skenario 13	0,3	0,5	10

Skenario 14	0,4	0,5	10
Skenario 15	0,5	0,5	10
Skenario 16	0,6	0,5	10
Skenario 17	0,7	0,5	10
Skenario 18	0,8	0,5	10
Skenario 19	0,9	0,5	10
Skenario 20	1	0,5	10

Pengujian terakhir dilakukan pada jumlah populasi dengan kombinasi *crossover rate* dan *mutasi rate* terbaik yang didapatkan pada hasil pengujian kedua. Pengujian dilakukan dengan menetapkan jumlah populasi 5, 10, 15, 20, 25, 50 dan 100. Skenario pada pengujian ketiga dapat diamati pada Tabel 3.12 dengan asumsi kombinasi *crossover rate* dan *mutasi rate* terbaik yang didapatkan pada pengujian sebelumnya adalah 0,5 dan 0,2.

Tabel 03.12 Skenario uji coba jumlah populasi

Skenario	<i>Crossover Rate</i>	<i>Mutasi Rate</i>	Populasi
Skenario 1	0,5	0,2	5
Skenario 2	0,5	0,2	10
Skenario 3	0,5	0,2	15
Skenario 4	0,5	0,2	20
Skenario 5	0,5	0,2	25
Skenario 6	0,5	0,2	50
Skenario 7	0,5	0,2	100

BAB IV

UJI COBA DAN PEMBAHASAN

Pada bab uji coba dan pembahasan akan dijelaskan mengenai rangkaian uji coba yang telah dilakukan sesuai dengan skenario yang telah dipaparkan pada bab sebelumnya. Dijelaskan juga pembahasan data yang diperoleh dalam proses uji coba, kemudian ditambahkan perbandingan dengan hasil penelitian-penelitian sebelumnya.

4.1 Data Uji Coba

Data uji coba pada penelitian ini diambil dari SMP N 1 Nisam Antara yang beralamat di Jalan Alue Papeun - Krueng Tuan Kec. Nisam Antara Kab. Aceh Utara Prov. Aceh. Data diperoleh dengan metode dengan wakil kepala sekolah bagian kurikulum. Pihak sekolah memberikan data berupa data pembagian tugas guru tahun ajaran 2019/2020 semester genap dan jadwal pelajaran semester genap tahun ajaran 2019/2020. Kedua data yang diberikan berupa *file* dalam format xls atau format exel. Data pembagian tugas guru berisi nama guru, pelajaran yang diampu dan jumlah jam mengajar. Sedangkan data jadwal pelajaran merupakan jadwal pelajaran yang disusun secara manual oleh bagian kurikulum.

SMP N 1 Nisam Antara memiliki komposisi 10 mata pelajaran, 35 guru, dan 14 kelas. Kelas di SMP N 1 Nisam Antara terdiri dari lima kelas 7, lima kelas 8 dan empat kelas 9. Setiap kelas memiliki total 38 jam pelajaran yang rinciannya dapat diamati pada Tabel 4.1. Jumlah jam pelajaran untuk setiap pelajaran berbeda, ada pelajaran yang memiliki 2 jam pelajaran, 3 jam pelajaran, 4 jam pelajaran, 5 jam

pelajaran, dan 6 jam pelajaran. Dalam pembentukan jadwal, setiap pelajaran dalam satu hari maksimal hanya mengambil porsi 2 jam mata pelajaran pada tiap kelas.

Tabel 04.1 Pelajaran dan jam pelajaran

Pelajaran	Kode	Jumlah Jam Pelajaran
Matematika	MTK	5
Bahasa Indonesia	BI	6
Agama	AGM	3
PKN	PKN	3
Ilmu Pengetahuan Alam	IPA	5
Ilmu Pengetahuan Sosial	IPS	4
Bahasa Inggris	ENG	4
Seni Budaya	SBD	3
Olah Raga	ORG	3
Prakarya	PKR	2

Susunan guru dengan pelajaran yang diampu serta jumlah mengajarnya dapat diamati pada Tabel 4.2. Penyusunan jadwal pelajaran akan dilakukan berdasarkan jumlah jam mengajar setiap guru pada pelajaran yang diampu. Guru tidak terikat dengan kelas, melainkan dengan pelajaran. Sehingga seorang guru dapat mengajar di kelas manapun jika di kelas tersebut ada pelajaran yang diajar.

Tabel 04.2 Guru, Pelajaran dan Jumlah Jam Mengajar

No	Guru	Kode Guru	Pelajaran	Jam Mengajar
1	M. AMIN, S.Pd	AM	PPKN	12
2	RUHANI, A.Md	RU	IPS	16
3	WARDANA, S.Pd	WA	B.INDONESIA	24
4	ISNAWATI, S.Pd	IS	MATEMATIKA	25
5	RAMLAH, S.Pd	RA	PPKN	24
6	HERLINA, S.Pd	HE	B.INGGRIS	12
7	ARRITAWATI, S.Pd	AR	SENI BUDAYA	3
			IPS	16
8	MARDLIAH, S.Pd	MA	IPS	12
			PKN	3

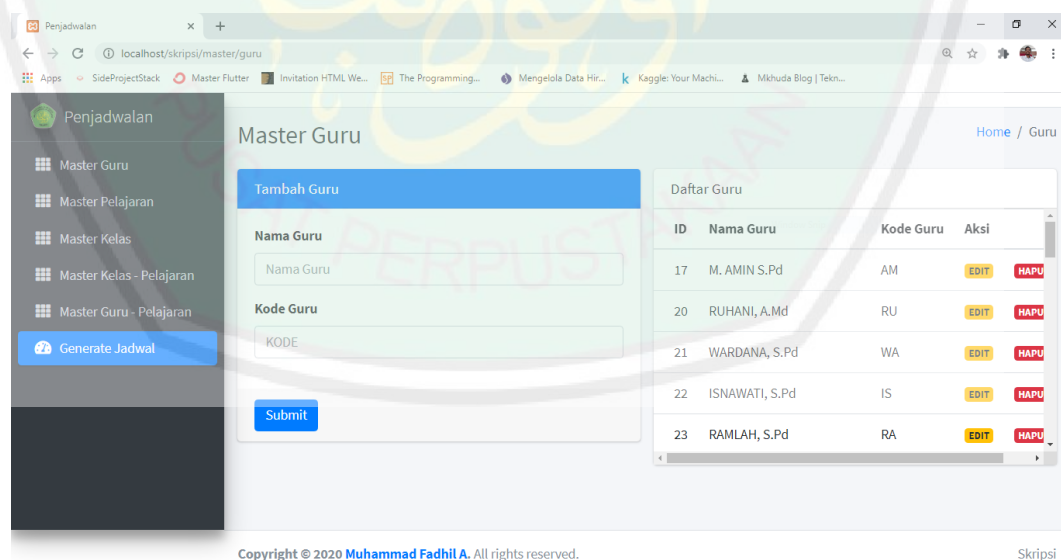
9	ELIZA NORA, S.Pd	EL	IPA	20
			PRAKARYA	4
10	KHAIRANI, S.Ag	KH	AGAMA	21
11	SAFRIYATI, S.Pd	SA	MATEMATIKA	25
12	MUKARRAMAH, S.Pd.I	MK	IPA	10
			PRAKARYA	2
13	SURYANI, S.Pd	SU	B.INGGRIS	12
			SENI BUDAYA	3
14	Drs. MUHAMMAD YUSUF	MU	IPS	12
15	NURAINI, S.Pd	NU	IPA	10
			PRAKARYA	14
16	RADLIAH, S.Pd	RD	IPA	20
			PRAKARYA	4
17	NURJULIATI, S.Pd	NR	B.INGGRIS	16
18	ROHANA, S.Pd	RO	MATEMATIKA	10
19	NURATIAH, S.Pd	NT	IPA	10
			PRAKARYA	4
20	JAMILAH, S.Pd	JA	B.INDONESIA	24
21	LIS SURYANI, S.Pd	LI	B.INGGRIS	16
22	JUM'ATI, S. Ag	JU	AGAMA	21
23	JURIAWATI, A.Md	JR	SENI BUDAYA	18
24	LINDAWATI, S.Pd	LD	MATEMATIKA	10
			B . INDONESIA	6
25	ASNIRAWATI, S.Pd	AS	PJOK	9
26	CUT RAHMI S.Pd	CU	SENI BUDAYA	12
27	MAHLIL, S.Pd	ML	PENJAS	15
28	HALIMAHTUSAKDIAH, S.Pd	HA	B. INDONESIA	12
29	FITRI YANI, S.Pd	FI	B.INDONESIA	6
30	MISRAWATI, S.Pd	MI	PJOK	6
31	ZAMAN HURI	ZH	PENJASKES	12
32	NURHAYATI, S.Pd	NY	B. INDONESIA	6
33	NOERUL ASTRI	NO	SENI BUDAYA	3
34	LIZA KHAIRINA, S.Pd	LZ	SENI BUDAYA	3
			PKN	3
35	SERI ARYATI, S.Pd	SE	B. INDONESIA	6

Sekolah memiliki 6 hari aktif, dari hari Senin hingga Sabtu. Adapun pembagian jam pelajaran pada hari aktif tersebut sebagai berikut: 6 jam pelajaran pada hari Senin ditambah dengan satu jam pelajaran untuk upacara pagi, 7 jam pelajaran pada hari Selasa, Rabu, dan Kamis, kemudian 5 jam pelajaran pada hari Jumat dan terakhir 6 pelajaran pada hari Sabtu.

4.2 Pengujian Fungsionalitas Aplikasi

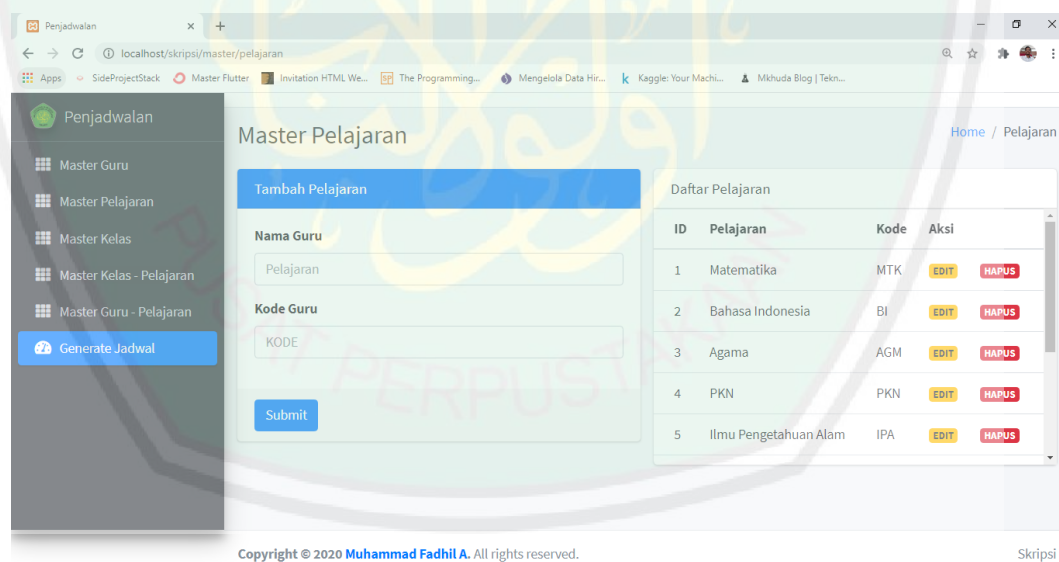
4.2.1 Fungsi Manajemen Data

Fungsi manajemen data berfungsi untuk menambah data, mengedit data dan menghapus data. Data yang dibutuhkan untuk menyusun jadwal adalah data guru, data pelajaran, data kelas, data kelas pelajaran dan data guru kelas pelajaran. Keseluruhan master data memiliki fungsi tambah data baru, menampilkan tabel data, edit data dan hapus data. Navigasi untuk setiap master data terdapat pada bagian *side bar*.



Gambar 4. 1 Tampilan Halaman Master Guru

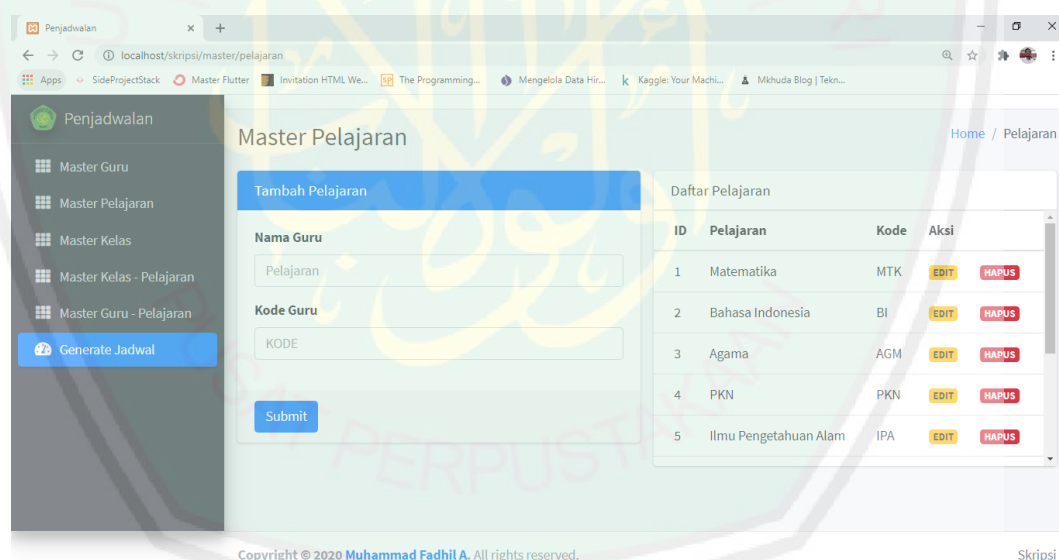
Gambar 4.1 menunjukkan halaman master guru yang memuat semua fungsi yang berkaitan dengan data guru. Pada halaman ini terdapat *form* tambah guru untuk menginputkan guru baru. Nilai yang dibutuhkan untuk menambah seorang guru baru adalah nama guru dan kode guru. Tombol submit berfungsi untuk menambahkan guru baru ke *database*. Terdapat pula tabel daftar guru yang berisi seluruh guru yang telah diinputkan. Tabel daftar guru menampilkan informasi id, nama guru, kode guru serta tombol untuk melakukan aksi edit dan hapus. Tombol edit akan mengarahkan *user* menuju halaman edit guru yang memuat *form* informasi nama guru dan kode guru dengan data yang telah ada sebelumnya sehingga *user* dapat mengubah informasi yang diperlukan. Tombol hapus berfungsi untuk menghapus data guru dari *database*. Guru tidak dapat dihapus dari *database* jika masih mengajar pelajaran tertentu yang tercatat pada master guru pelajaran.



Gambar 4. 2 Tampilan Halaman Master Pelajaran

Gambar 4.2 menampilkan halaman master pelajaran yang memuat semua fungsi yang berkaitan dengan data pelajaran. Pada halaman ini terdapat *form* tambah pelajaran untuk menginputkan pelajaran baru. Nilai yang dibutuhkan untuk

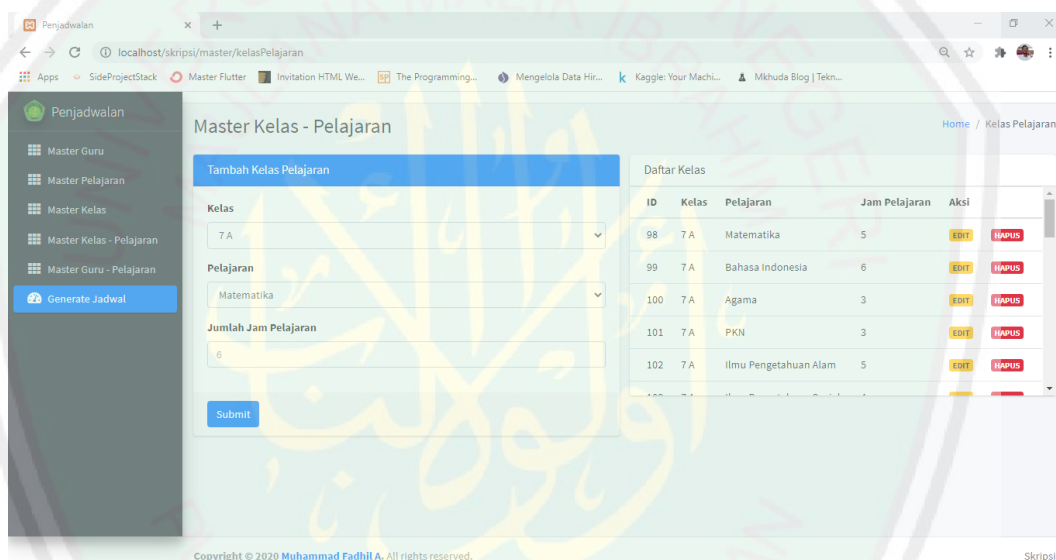
menambah sebuah pelajaran baru adalah nama pelajaran dan kode pelajaran. Tombol submit berfungsi untuk menambahkan pelajaran baru ke *database*. Terdapat pula tabel daftar pelajaran yang berisi seluruh pelajaran yang telah diinputkan. Tabel daftar pelajaran menampilkan informasi id, nama pelajaran, kode pelajaran serta tombol untuk melakukan aksi edit dan hapus. Tombol edit akan mengarahkan *user* menuju halaman edit pelajaran yang memuat *form* informasi nama pelajaran dan kode pelajaran sesuai dengan data yang telah ada sebelumnya sehingga *user* dapat mengubah informasi yang diperlukan. Tombol hapus berfungsi untuk menghapus data pelajaran dari *database*. Pelajaran tidak dapat dihapus dari *database* jika pelajaran tersebut masih tercatat pada master guru pelajaran dan master kelas pelajaran.



Gambar 4. 3 Tampilan Halaman Master Kelas

Gambar 4.3 menampilkan halaman master kelas yang memuat semua fungsi yang berkaitan dengan data kelas. Pada halaman ini terdapat *form* tambah kelas untuk menginputkan kelas baru. Nilai yang dibutuhkan untuk menambah sebuah kelas baru hanya nama kelas. Tombol submit berfungsi untuk menambahkan kelas

baru ke *database*. Terdapat pula tabel daftar kelas yang berisi seluruh kelas yang telah ada di *database*. Tabel daftar kelas menampilkan informasi id, nama kelas, serta tombol untuk melakukan aksi edit dan hapus. Tombol edit akan mengarahkan *user* menuju halaman edit kelas yang memuat *form* informasi nama kelas sesuai dengan data yang telah ada sebelumnya sehingga *user* dapat mengubah informasi yang diperlukan. Tombol hapus berfungsi untuk menghapus data kelas dari *database*. Kelas tidak dapat dihapus dari *database* jika kelas tersebut masih tercatat pada master kelas pelajaran.



Gambar 4. 4 Tampilan Halaman Master Kelas - Pelajaran

Gambar 4.4 menampilkan halaman master kelas-pelajaran yang memuat semua fungsi yang berkaitan dengan relasi antara data kelas dan data pelajaran. Pada halaman ini terdapat *form* tambah relasi kelas dan pelajaran untuk menginputkan relasi baru. Nilai yang dibutuhkan untuk menambah sebuah relasi baru adalah pilihan kelas yang diambil dari data kelas, pilihan pelajaran yang diambil dari data pelajaran, serta jumlah jam pelajaran. Tombol submit berfungsi untuk menambahkan relasi baru ke *database*. Terdapat pula tabel daftar kelas

pelajaran yang berisi seluruh relasi kelas dan pelajaran yang telah ada di *database*. Tabel daftar kelas pelajaran menampilkan informasi id, nama kelas, nama pelajaran, jumlah jam pelajaran, serta tombol untuk melakukan aksi edit dan hapus. Tombol edit akan mengarahkan *user* menuju halaman edit kelas pelajaran yang memuat *form* informasi pilihan nama kelas, pilihan nama pelajaran, dan jumlah jam pelajaran sesuai dengan data yang telah ada sebelumnya sehingga *user* dapat mengubah informasi yang diperlukan. Tombol hapus berfungsi untuk menghapus data kelas pelajaran dari *database*.

ID	Guru	Pelajaran	Jam Mengajar	Aksi
38	M. AMIN S.Pd	PKN	9	EDIT HAPUS
39	RUHANI, A.Md	Ilmu Pengetahuan Sosial	12	EDIT HAPUS

Gambar 4. 5 Tampilan Halaman Master Guru - Pelajaran

Gambar 4.5 menampilkan halaman master guru-pelajaran yang memuat semua fungsi yang berkaitan dengan relasi antara data guru dan data pelajaran. Pada halaman ini terdapat *form* tambah relasi guru dan pelajaran untuk menginputkan relasi baru. Nilai yang dibutuhkan untuk menambah sebuah relasi baru adalah pilihan nama guru yang diambil dari data guru, pilihan pelajaran yang diambil dari data pelajaran, serta jumlah jam mengajar. Tombol submit berfungsi untuk menambahkan relasi baru ke *database*. Terdapat pula tabel daftar guru pelajaran

yang berisi seluruh relasi guru dan pelajaran yang telah ada di *database*. Tabel daftar guru pelajaran menampilkan informasi id, nama guru, nama pelajaran, jumlah jam mengajar, serta tombol untuk melakukan aksi edit dan hapus. Tombol edit akan mengarahkan *user* menuju halaman edit guru pelajaran yang memuat *form* informasi pilihan nama guru, pilihan nama pelajaran, dan jumlah jam mengajar sesuai dengan data yang telah ada sebelumnya sehingga *user* dapat mengubah informasi yang diperlukan. Tombol hapus berfungsi untuk menghapus data guru pelajaran dari *database*.

4.2.1 Fungsi *Generate Jadwal*

ID	Banyak Gen	Maksimal Generasi	Crossover Rate	Mutasi Rate	Tabrak	Guru	Pelajaran	Fitness	Skenario	Lama Komputasi	Penggunaan M
67	10	1000	0.1	0.1	3	0	0	0.9709	final-1-1-	15.55356	5622408
68	10	1000	0.1	0.1	2	8	8	0.9802	final-1-1-	15.41068	5652176
69	10	1000	0.1	0.1	2	8	8	0.9802	final-1-1-	16.16432	5631160
70	10	1000	0.2	0.2	1	8	8	0.9899	final-2-2-	17.97855	5573576
71	10	1000	0.2	0.2	1	13	13	0.9898	final-2-2-	18.27173	5598456

Gambar 4. 6 Tampilan Halaman Master Guru - Pelajaran

Fungsi *generate* jadwal merupakan fungsi utama dalam aplikasi dan penelitian ini. Fungsi *generate* jadwal berfungsi untuk menyusun sebuah jadwal sesuai dengan data yang ada pada master data. Fungsi *generate* jadwal terdapat pada

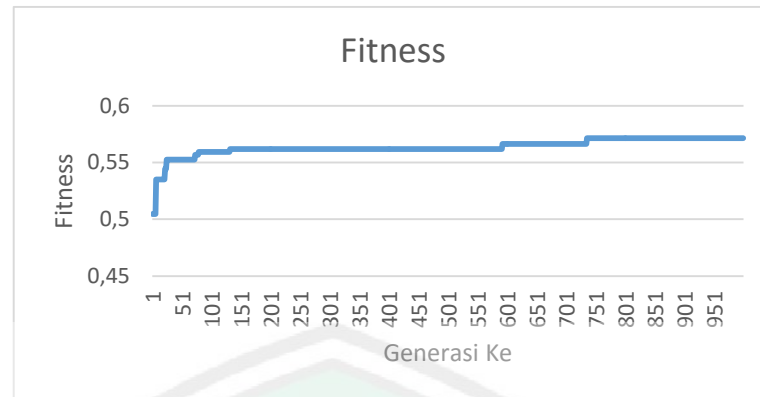
halaman *generate* jadwal. Sebagaimana terlihat pada Gambar 4.6, terdapat beberapa inputan yang diperlukan fungsi *generate* jadwal. *Form* berisi parameter yang dibutuhkan algoritma genetika, seperti banyak gen, maksimal generasi, *crossover rate* dan *mutasi rate*. Inputan skenario berfungsi untuk memudahkan untuk menemukan hasil *generate* jadwal yang diinginkan. Terdapat pula tabel hasil *generate* yang menampilkan info parameter algoritma genetika yang digunakan serta tombol untuk melihat jadwal yang dihasilkan sebagaimana terlihat pada Gambar 4.7.

	Senin	Senin	Senin	Senin	Senin	Senin	Selasa	Selasa	Selasa	Selasa	Selasa	Selasa	Selasa	Rabu	Rabu	Rabu	Rabu	Rabu	Rabu	Rabu	Rabu	Kamis	Kamis	Kamis	Kamis	Kamis	Kamis
Kelas	1	2	3	4	5	6	1	2	3	4	5	6	7	1	2	3	4	5	6	7	1	2	3	4	5	6	7
7 A	WA	WA	AR	KH	KH	WA	WA	IS	HE	HE	AM	AM	IS	IS	SA	EL	EL	AR	AR	HE	HE	AS	SA	SA	AS	AS	AM
7 B	AS	AS	WA	WA	HE	HE	KH	KH	IS	SA	SA	AR	WA	WA	IS	IS	AM	AM	RU	RU	RU	RU	IS	IS	SA	SA	EL
7 C	EL	EL	SA	SA	RU	RU	AM	SA	KH	AM	AM	HE	HE	WA	WA	SA	SA	AR	IS	IS	WA	WA	HE	HE	AR	AR	AS

Gambar 4. 7 Tampilan Halaman Jadwal

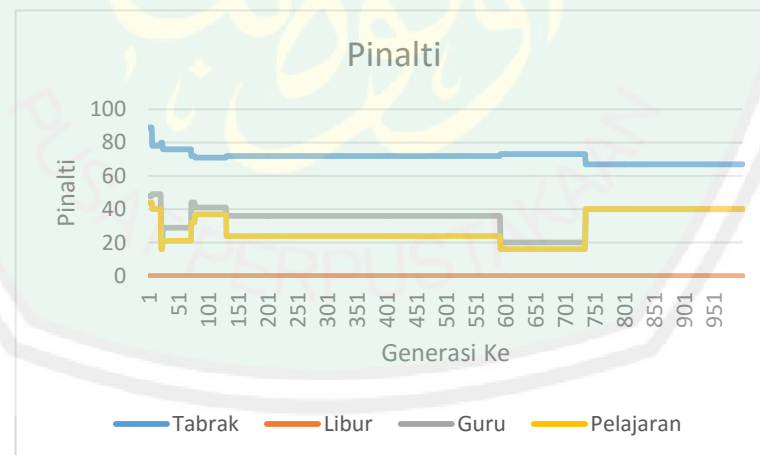
4.3 Pengujian Hasil Generate Aplikasi

Pengujian awal dilakukan dengan pengaturan parameter algoritma genetika sebagai berikut: banyak populasi 50, maksimal generasi 100, *crossover rate* 0.5 dan *mutasi rate* 0.5. Dengan pengaturan parameter demikian didapatkan jadwal yang memiliki *fitness* 0.5714 dengan lama komputasi selama 160,3844 detik atau sekitar 2 menit 40 detik. Pergerakan *fitness* terbaik pada setiap generasi dapat diamati pada Gambar 4.8.



Gambar 4. 8 Grafik Fitness Per Generasi

Jadwal yang dihasilkan memiliki 67 pinalti *hard constraint* berupa tabrakan jam mengajar seorang guru, 40 pinalti *soft constraint* guru yang terjadwal di kelas yang sama pada satu hari dan 40 penalti *soft constraint* pelajaran yang sama terjadwal lebih dari dua jam pelajaran pada satu hari. Sedangkan untuk pinalti *hard constraint* guru yang terjadwal pada hari liburnya tidak ada karena sekolah tidak mengizinkan guru untuk libur pada hari aktif. Pergerakan penalti dapat diamati pada Gambar 4.9. Hasil penyusunan jadwal dapat dilihat pada Tabel 4.1.



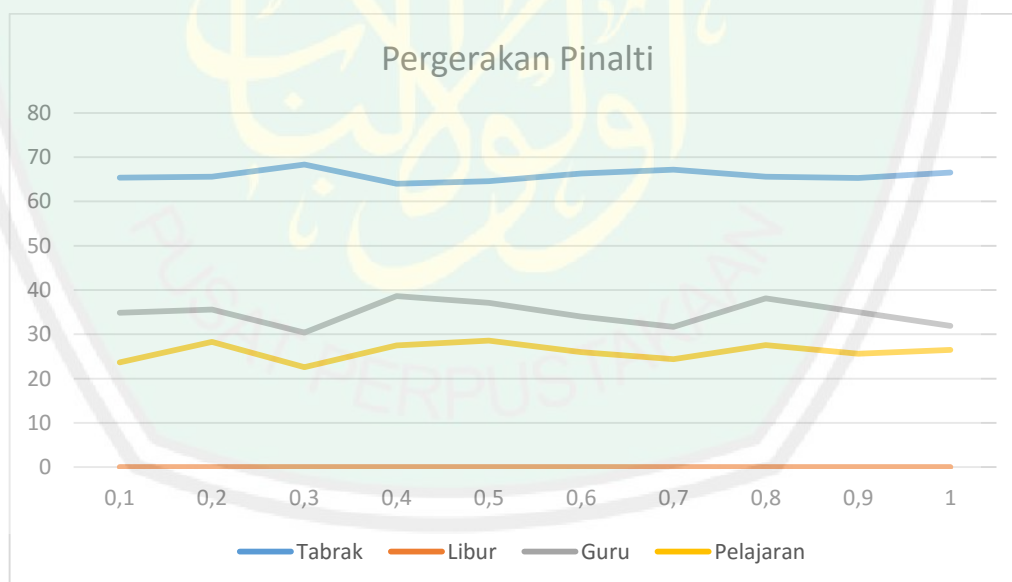
Gambar 4. 9 Grafik Pinalti Pergenerasi

Tabel 04.3 Hasil penyusunan jadwal

7 A	IPA (EL)	ORG (AS)	ORG (AS)	BI (WA)	BI (WA)	IPA (EL)	IPA (EL)	ENG (HE)	ENG (HE)	PKR (EL)	PKR (EL)	IPS (RU)	IPS (RU)	AGM (KH)	AGM (KH)	AGM (KH)	SBD (HE)	SBD (HE)	PKN (AM)	SBD (HE)	MTK (IS)	MTK (IS)	MTK (IS)	MTK (IS)	ENG (HE)	ENG (HE)	ORG (AS)	BI (WA)	BI (WA)	PKN (AM)	PKN (AM)	MTK (IS)	BI (WA)	BI (WA)	IPS (RU)	IPS (RU)	IPA (EL)	IPA (EL)			
7 B	SBD (SU)	SBD (SU)	IPS (RU)	IPS (RU)	MTK (IS)	MTK (IS)	BI (WA)	BI (WA)	PKN (AM)	PKR (EL)	PKR (EL)	AGM (KH)	SBD (SU)	ENG (HE)	ENG (HE)	ORG (AS)	ORG (AS)	BI (WA)	BI (WA)	BI (WA)	IPA (EL)	IPA (EL)	MTK (IS)	MTK (IS)	AGM (KH)	AGM (KH)	PKN (AM)	PKN (AM)	IPS (RU)	IPS (RU)	IPA (EL)	IPA (EL)	ORG (AS)	ORG (AS)	MTK (IS)	ENG (HE)	ENG (HE)	ENG (HE)			
7 C	IPS (RU)	IPS (RU)	ORG (AS)	IPA (EL)	IPA (EL)	PKR (MK)	PKR (MK)	SBD (CU)	SBD (CU)	SBD (CU)	AGM (KH)	AGM (KH)	ENG (HE)	ENG (HE)	IPS (RU)	IPS (RU)	MTK (IS)	MTK (IS)	BI (WA)	BI (WA)	BI (WA)	BI (WA)	PKN (AM)	PKN (AM)	ORG (AS)	ORG (AS)	ENG (HE)	ENG (HE)	AGM (KH)	MTK (IS)	BI (WA)	BI (WA)	IPA (EL)	IPA (EL)	MTK (IS)	MTK (IS)	IPA (EL)	PKN (AM)			
7 D	IPA (EL)	IPA (EL)	BI (WA)	BI (WA)	MTK (IS)	MTK (IS)	IPS (RU)	IPS (RU)	BI (WA)	BI (WA)	PKN (AM)	SBD (CU)	IPA (EL)	IPS (RU)	IPS (RU)	MTK (IS)	ORG (ML)	ENG (SU)	ENG (SU)	ENG (SU)	ENG (SU)	ORG (ML)	ORG (ML)	AGM (KH)	AGM (KH)	BI (WA)	BI (WA)	MTK (IS)	MTK (IS)	PKN (AM)	PKN (AM)	SBD (CU)	SBD (CU)	PKR (NU)	PKR (NU)	IPA (EL)	IPA (EL)	AGM (KH)	AGM (KH)		
7 E	IPS (AR)	IPS (AR)	PKR (NU)	PKR (NU)	ENG (SU)	ENG (SU)	ENG (SU)	ENG (SU)	SBD (CU)	SBD (CU)	IPA (MK)	IPA (MK)	IPA (MK)	IPA (MK)	BI (JA)	BI (JA)	SBD (CU)	MTK (IS)	MTK (IS)	ORG (ML)	ORG (ML)	AGM (KH)	AGM (KH)	IPA (MK)	ORG (ML)	PKN (RA)	PKN (RA)	PKN (RA)	MTK (IS)	MTK (IS)	BI (JA)	BI (JA)	MTK (IS)	BI (JA)	BI (JA)	IPS (AR)	IPS (AR)	IPS (AR)	AGM (KH)		
8 A	BI (JA)	BI (JA)	BI (JA)	BI (JA)	MTK (SA)	MTK (SA)	IPA (MK)	IPA (MK)	ENG (SU)	ENG (SU)	SBD (CU)	SBD (CU)	MTK (SA)	MTK (SA)	IPA (MK)	IPA (MK)	SBD (CU)	ORG (ML)	ORG (ML)	PKN (RA)	IPA (JA)	BI (JA)	BI (JA)	BI (JA)	PKN (RA)	IPS (AR)	IPS (AR)	ORG (ML)	AGM (KH)	AGM (KH)	MTK (SA)	AGM (KH)	ENG (SU)	ENG (SU)	PKR (NU)	PKR (NU)	IPS (AR)	IPS (AR)			
8 B	AGM (KH)	ORG (ML)	ORG (ML)	AGM (KH)	AGM (KH)	PKR (NU)	PKR (NU)	PKN (RA)	IPA (NU)	IPS (AR)	IPS (AR)	MTK (SA)	SBD (NO)	MTK (SA)	MTK (SA)	MTK (SA)	MTK (SA)	BI (JA)	BI (JA)	IPA (NU)	IPA (NU)	PKN (RA)	PKN (RA)	ENG (NR)	ENG (NR)	ENG (NR)	ENG (NR)	BI (JA)	BI (JA)	BI (JA)	BI (JA)	IPS (AR)	IPS (AR)	IPA (NU)	IPA (NU)	ORG (ML)	SBD (NO)	SBD (NO)	SBD (NO)		
8 C	ORG (ML)	PKR (NU)	PKR (NU)	ENG (NR)	ENG (NR)	MTK (SA)	MTK (SA)	ORG (ML)	ORG (ML)	IPA (NU)	IPA (NU)	IPS (AR)	IPS (AR)	ENG (NR)	ENG (NR)	AGM (JU)	SBD (LZ)	SBD (LZ)	PKN (RA)	BI (JA)	BI (JA)	IPA (NU)	IPA (NU)	IPA (NU)	MTK (SA)	MTK (SA)	SBD (LZ)	BI (JA)	BI (JA)	MTK (SA)	PKN (RA)	PKN (RA)	BI (JA)	BI (JA)	AGM (JU)	AGM (JU)	IPS (AR)	IPS (AR)			
8 D	AGM (JU)	AGM (JU)	BI (LD)	BI (LD)	IPA (RD)	IPA (RD)	ORG (MI)	PKN (RA)	PKN (RA)	SBD (JR)	IPS (MA)	IPS (MA)	MTK (SA)	BI (LD)	BI (LD)	BI (LD)	MTK (SA)	MTK (SA)	PKR (NU)	PKR (NU)	ENG (NR)	ENG (NR)	IPA (RD)	IPA (RD)	IPA (RD)	IPA (RD)	IPS (MA)	IPS (MA)	AGM (JU)	SBD (JR)	ENG (NR)	ENG (NR)	ORG (MI)	ORG (MI)	SBD (JR)	SBD (JR)	MTK (SA)	MTK (SA)	PKN (RA)	PKN (RA)	
8 E	MTK (SA)	IPA (RD)	PKN (RA)	PKN (RA)	PKR (NU)	PKR (NU)	ORG (MI)	PKN (RA)	IPS (MA)	IPS (MA)	AGM (JU)	ORG (MI)	ORG (MI)	MTK (SA)	MTK (SA)	MTK (SA)	MTK (SA)	IPA (RD)	IPA (RD)	ENG (NR)	ENG (NR)	IPS (MA)	IPS (MA)	BI (HA)	BI (HA)	AGM (JU)	AGM (JU)	SBD (JR)	SBD (JR)	ENG (NR)	ENG (NR)	SBD (JR)	IPA (RD)	IPA (RD)	BI (HA)	BI (HA)	BI (HA)	BI (HA)			
9 A	IPA (RD)	IPA (RD)	PKN (RA)	MTK (RO)	MTK (RO)	AGM (JU)	MTK (RO)	MTK (RO)	IPA (RD)	IPA (RD)	BI (HA)	BI (HA)	ENG (LI)	ENG (LI)	PKR (RD)	PKR (RD)	PKN (RA)	PKN (RA)	ORG (ZH)	MTK (RO)	IPS (MA)	IPS (MA)	ENG (LI)	ENG (LI)	IPS (MA)	IPS (MA)	BI (HA)	BI (HA)	BI (HA)	BI (HA)	ORG (ZH)	ORG (ZH)	SBD (JR)	SBD (JR)	AGM (JU)	AGM (JU)	MTK (RO)	MTK (RO)	SBD (JR)	SBD (JR)	
9 B	IPA (RD)	IPA (RD)	BI (FI)	BI (FI)	MTK (RO)	MTK (RO)	BI (FI)	BI (FI)	IPS (HA)	IPS (HA)	IPA (RD)	IPA (RD)	ORG (ZH)	ORG (ZH)	SBD (JR)	BI (FI)	BI (FI)	ENG (LI)	ENG (LI)	PKR (RD)	PKR (RD)	ORG (ZH)	MTK (RO)	ENG (LI)	ENG (LI)	IPA (RD)	AGM (JU)	IPS (HA)	IPS (HA)	SBD (JR)	SBD (JR)	AGM (JU)	AGM (JU)	ENG (LI)	ENG (LI)	ENG (LI)	ENG (LI)	MTK (RO)	MTK (RO)	PKN (RA)	PKN (RA)
9 C	PKN (MA)	PKN (MA)	SBD (JR)	SBD (JR)	AGM (JU)	AGM (JU)	IPA (NT)	IPA (NT)	AGM (JU)	MTK (LD)	MTK (LD)	ORG (ZH)	BI (NY)	BI (NY)	PKR (NT)	PKR (NT)	SBD (JR)	IPA (NT)	IPA (NT)	ORG (ZH)	ORG (ZH)	BI (NY)	BI (NY)	PKN (MA)	BI (NY)	BI (NY)	MTK (LD)	MTK (LD)	IPS (ZH)	IPS (ZH)	ENG (LI)	ENG (LI)	ENG (LI)	ENG (LI)	MTK (LD)	MTK (LD)	IPS (ZH)	IPS (ZH)	IPS (ZH)	IPS (ZH)	
9 D	IPS (NY)	IPS (NY)	BI (SE)	BI (SE)	IPA (NT)	IPA (NT)	IPA (NT)	ORG (ZH)	ORG (ZH)	AGM (JU)	AGM (JU)	ENG (LI)	ENG (LI)	IPA (NT)	IPA (NT)	MTK (LD)	MTK (LD)	IPS (NY)	IPS (NY)	BI (SE)	BI (SE)	SBD (JR)	ENG (LI)	ENG (LI)	SBD (JR)	SBD (JR)	AGM (JU)	ORG (ZH)	BI (SE)	BI (SE)	MTK (LD)	PKN (LZ)	PKR (NT)	PKR (NT)	PKN (LZ)	PKN (LZ)	MTK (LD)	MTK (LD)			

4.4 Pengujian Variabel Algoritma Genetika

Pengujian variabel pertama algoritma genetika dilakukan dengan mengubah nilai *crossover rate* dan *mutasi rate* secara serentak, dimulai dari 0,1 hingga 1,0 dengan selisih 0,1. Uji coba kemudian dilakukan dengan mengumpulkan data pinalti, *fitness*, waktu komputasi dan penggunaan memori pada 10 percobaan dan membandingkan rata-rata pada hasil yang didapat. Pada pengujian ini jumlah populasi ditetapkan sejumlah 10 dan dengan 1000 maksimal generasi. Pergerakan rata-rata penalti untuk setiap skenario dapat diamati pada Gambar 4.10. Gambar 4.11 menampilkan pergerakan rata-rata *fitness* untuk setiap skenario. Gambar 4.12 dan gambar 4.13 menunjukkan pergerakan rata-rata waktu komputasi dan rata-rata memori yang digunakan dalam setiap generasi. Waktu komputasi ditampilkan dalam satuan detik dan memori ditampilkan dalam satuan *byte*.



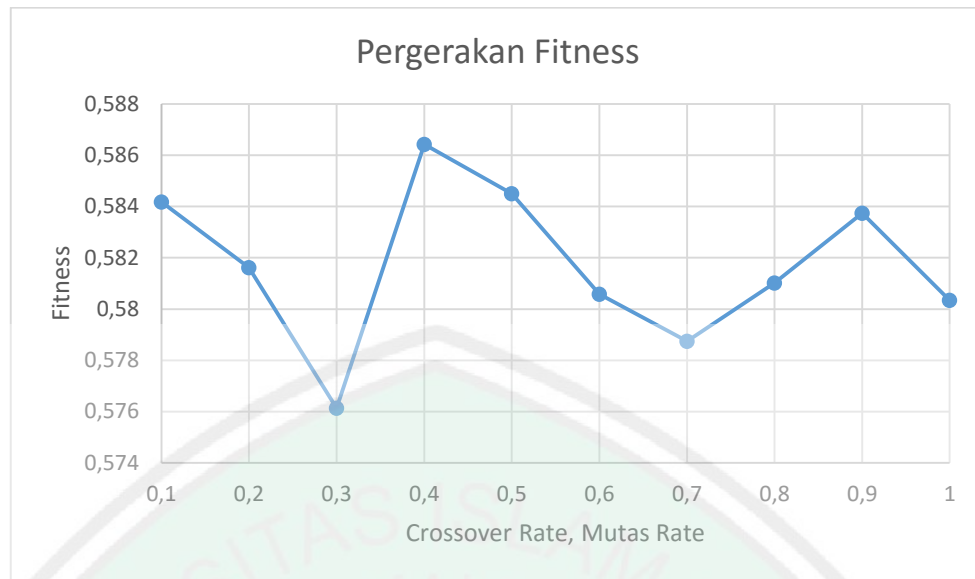
Gambar 4. 10 Pergerakan Penalti Pada Setiap Skenario

Gambar 4.10 menunjukkan pergerakan rata-rata jumlah penalti pada setiap percobaan. Terlihat bahwa pinalti *hard constraint* tabrakan jadwal mengajar seorang guru adalah pinalti terbesar pada setiap percobaan dengan rentang 64

hingga 69 penalti. Penalti tabrakan jadwal mengajar terendah terdapat pada skenario *crossover rate* dan *mutasi rate* 0.4 dengan rata-rata 64 penalti. Sedangkan pinalti terbesar terjadi pada skenario *crossover rate* dan *mutasi rate* 0,3 dengan rata-rata 68,3 pinalti.

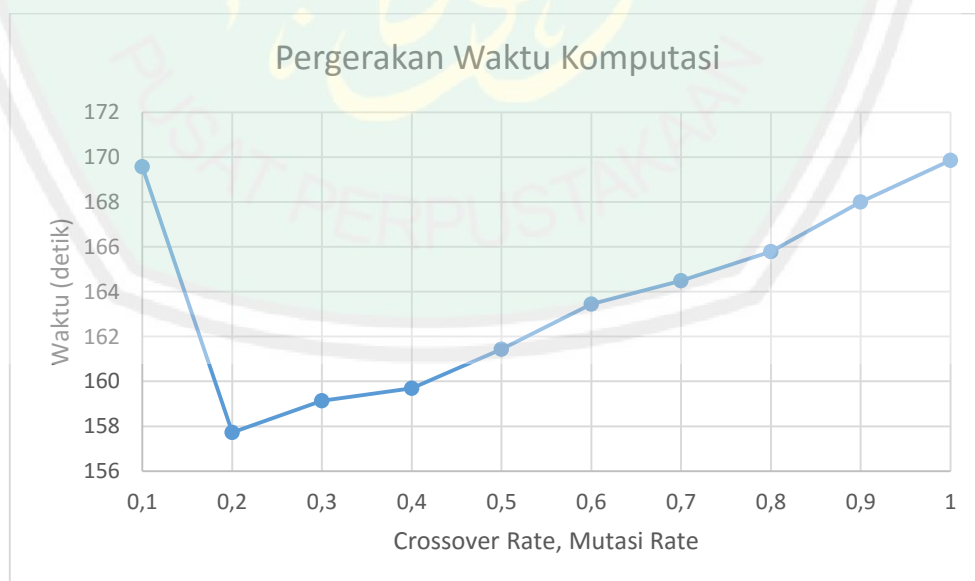
Selanjutnya Gambar 4.10 juga menunjukkan bahwa penalti *soft constraint* seorang guru terjadwal lebih dari satu pelajaran pada kelas yang sama dalam satu hari adalah pinalti kedua terbesar pada setiap percobaan dengan rentang 30 hingga 39 penalti. Penalti *soft constraint* guru mengajar pada kelas yang sama terendah terjadi pada skenario *crossover rate* dan *mutasi rate* 0,3 dengan rata-rata 30,4 pinalti. Sedangkan pinalti terbesar terjadi pada skenario *crossover rate* dan *mutasi rate* 0,4 dengan rata-rata 38,6 pinalti.

Kemudian Gambar 4.10 juga menunjukkan bahwa pinalti *soft constraint* suatu pelajaran terjadwal lebih dari dua jam pelajaran pada kelas yang sama dalam satu hari adalah pinalti terbesar selanjutnya pada setiap percobaan dengan rentang 22 hingga 29 penalti. Penalti *soft constraint* pelajaran terendah terjadi pada skenario *crossover rate* dan *mutasi rate* 0,3 dengan rata-rata 22,6 pinalti. Sedangkan pinalti terbesar terjadi pada skenario *crossover rate* dan *mutasi rate* 0,5 dengan rata-rata 28,6 pinalti.



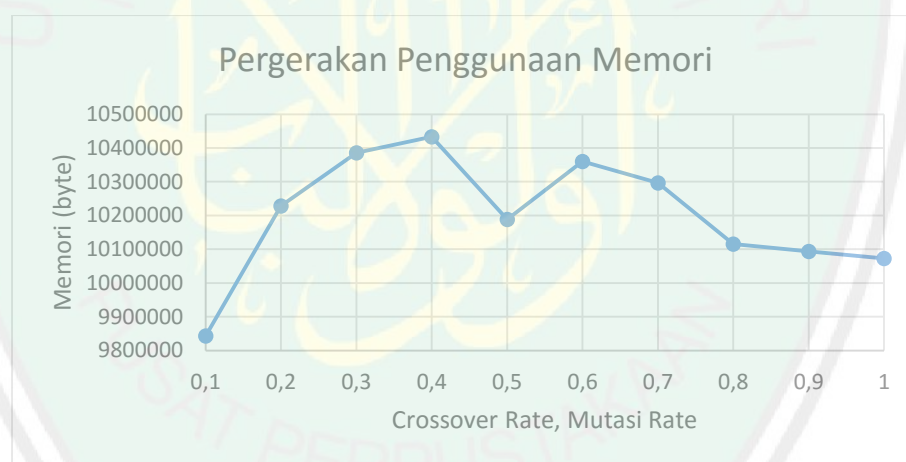
Gambar 4. 11 Pergerakan Fitness pada Setiap Skenario

Fitness pada setiap generasi tidak terlalu berbeda jauh. Rata-rata untuk seluruh skenario berada pada 0,58. Namun pada skenario *crossover rate* dan *mutasi rate* 0,4 menjadi skenario yang memiliki *fitness* terbaik dengan nilai *fitness* 0,58643. Sedangkan skenario *crossover rate* dan *mutasi rate* 0,3 menjadi skenario yang memiliki nilai *fitness* terendah dengan nilai *fitness* 0,57614.



Gambar 4. 12 Pergerakan Waktu Komputasi pada Setiap Skenario

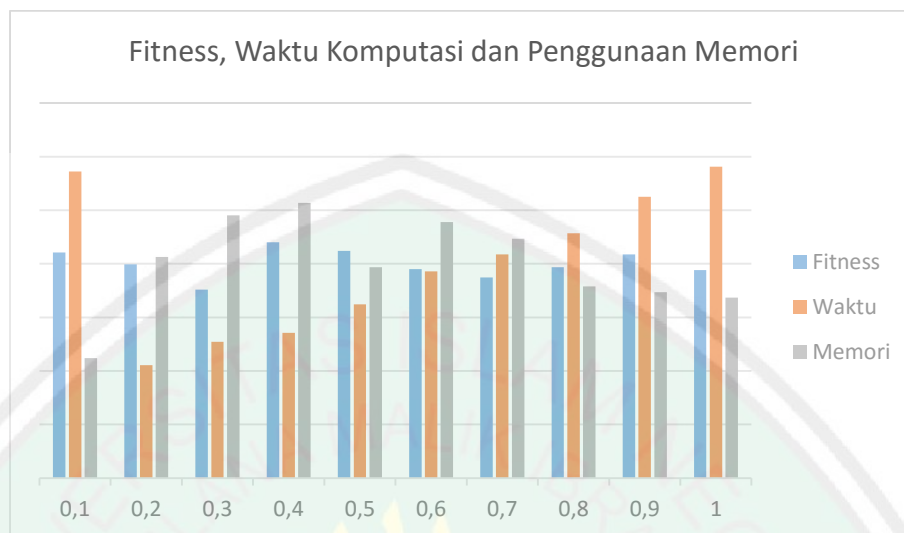
Pergerakan rata-rata waktu komputasi untuk setiap skenario terlihat peningkatan untuk setiap skenario kecuali pada skenario pertama dengan *crossover rate* dan *mutasi rate* 0,1. Terjadi penurunan tajam pada rata-rata waktu komputasi pada skenario pertama dengan skenario berikutnya, dengan waktu komputasi rata-rata 169,564 detik dan 157,717 detik. Kemudian pada skenario berikutnya terjadi kenaikan dengan rata-rata 1,5 detik untuk setiap skenario. Waktu komputasi tercepat diraih oleh skenario *crossover rate* dan *mutasi rate* 0,2 dengan rata-rata waktu komputasi 157,717 detik. Sedangkan waktu komputasi terlama diperoleh skenario *crossover rate* dan *mutasi rate* 1,0 dengan waktu komputasi 169,8499 detik, diikuti oleh skenario pertama dengan perolehan waktu komputasi 169,5647 detik.



Gambar 4. 13 Pergerakan Penggunaan Memori pada Setiap Skenario

Penggunaan memori pada setiap kali penyusunan jadwal berkisar antara 9.800.000 *byte* atau 9,8 Mb hingga 10.500.000 *byte* atau 10,5 Mb dengan rata-rata pada penggunaan memori 10.201.709 *byte* atau 10,2 Mb. Dalam 10 skenario pengujian diketahui bahwa penggunaan memori terendah diraih pada skenario pertama dengan *crossover rate* dan *mutasi rate* 0,1 yang rata-rata menggunakan

9.842.901 *byte*. Sedangkan konsumsi memori terbesar diketahui pada skenario *crossover rate* dan *mutasi rate* 0,4 dengan konsumsi memori 10.433.612 *byte*.



Gambar 4. 14 Perbandingan Fitness, Waktu Komputasi dan Penggunaan Memori

Gambar 4.14 menunjukkan perbandingan nilai *fitness*, waktu komputasi dan penggunaan memori dengan normalisasi. Dari gambar 4.14 saja belum dapat ditarik kesimpulan manakah skenario terbaik dengan *fitness* terbaik, waktu komputasi tercepat dan penggunaan memori paling sedikit. Sehingga penentuan skenario terbaik dilakukan dengan pemberian poin untuk setiap kategori berdasarkan *rank* yang diperoleh setiap skenario pada setiap kategori. Setelah pemberian poin didapati bahwa skenario terbaik diraih oleh skenario *crossover rate* dan *mutasi rate* 0,5 yang memperoleh total skor 22 dengan rincian: 9 poin pada kategori *fitness* terbaik kedua, 7 poin pada kategori waktu tercepat ke 4 dan 6 poin pada kategori penggunaan memori paling sedikit ke 5. Detail perhitungan skor untuk setiap skenario dapat diamati pada Tabel 4.4.

Tabel 04.4 Perhitungan Skor untuk Menentukan Skenario Terbaik

Crossover Rate	Mutasi Rate	Fitness	Waktu	Memori	Total Skor
0.1	0.1	3 (+8)	9 (+2)	1 (+10)	20
0.2	0.2	5 (+6)	1 (+10)	6 (+5)	21
0.3	0.3	10 (+1)	2 (+9)	9 (+2)	12
0.4	0.4	1 (+10)	3 (+8)	10 (+1)	19
0.5	0.5	2 (+9)	4 (+7)	5 (+6)	22
0.6	0.6	7 (+4)	5 (+6)	8 (+3)	13
0.7	0.7	9 (+2)	6 (+5)	7 (+4)	11
0.8	0.8	6 (+5)	7 (+4)	4 (+7)	16
0.9	0.9	4 (+7)	8 (+3)	3 (+8)	18
1	1	8 (+3)	10 (+1)	2 (+9)	13

Kemudian dilanjutkan dengan pengujian kedua dengan mengubah *crossover rate* dan *mutasi rate* secara tidak seragam. Nilai *crossover rate* dan *mutasi rate* terbaik yang didapat dari pengujian sebelumnya adalah 0,5. Pengujian untuk setiap skenario dilakukan sebanyak 10 kali. Rata-rata nilai dari *fitness*, lama komputasi dan penggunaan memori menjadi tolak ukur keoptimalan variabel. Setelah dilakukan pengujian, sama halnya seperti sebelumnya digunakan sistem ranking dan skor untuk menentukan skenario terbaik. Hasil dari pengujian dapat diamati pada Tabel 4.5.

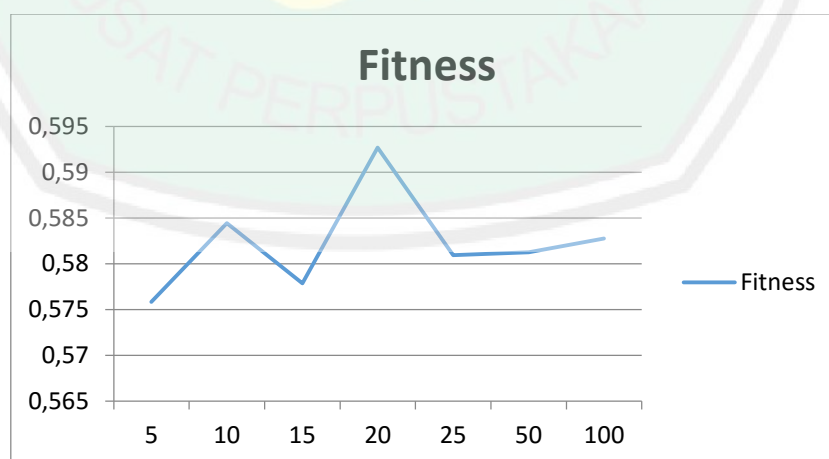
Tabel 04.5 Perhitungan Skor untuk Menentukan Skenario Terbaik Pengujian Kedua

Crossover Rate	Mutasi Rate	Fitness	Lama Komputasi	Penggunaan Memori	Total Skor
0,1	0,5	5 (+15)	5 (+15)	18 (+2)	32
0,2	0,5	15 (+5)	7 (+13)	17 (+3)	21
0,3	0,5	17 (+3)	8 (+12)	16 (+4)	19
0,4	0,5	3 (+17)	13 (+7)	15 (+5)	29
0,6	0,5	13 (+7)	9 (+11)	5 (+15)	33
0,7	0,5	4 (+16)	10 (+10)	4 (+16)	42
0,8	0,5	10 (+10)	11 (+9)	3 (+17)	36
0,9	0,5	11 (+9)	14 (+6)	1 (+19)	34
1	0,5	19 (+1)	6 (+14)	2 (+18)	33
0,5	0,1	18 (+2)	1 (+19)	11 (+9)	30
0,5	0,2	9 (+11)	2 (+18)	6 (+14)	43

0,5	0,3	2 (+18)	3 (+17)	13 (+7)	42
0,5	0,4	6 (+14)	4 (+16)	12 (+8)	38
0,5	0,6	14 (+6)	15 (+5)	9 (+11)	22
0,5	0,7	8 (+12)	16 (+4)	7 (+13)	29
0,5	0,8	1 (+19)	19 (+1)	14 (+6)	26
0,5	0,9	12 (+8)	17 (+3)	10 (+10)	21
0,5	1	16 (+4)	18 (+2)	8 (+12)	18
0,5	0,5	7 (+13)	12 (+8)	19 (+1)	22

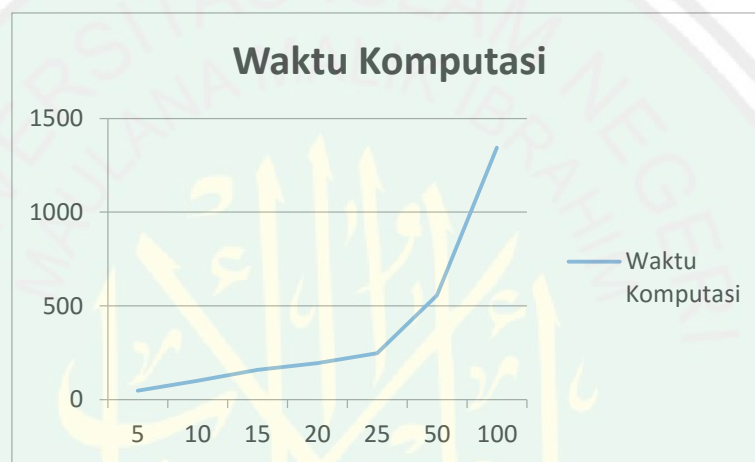
Kemudian dilanjutkan dengan uji coba variabel jumlah populasi dengan menggunakan nilai *crossover rate* 0,5 dan *mutasi rate* 0,2 sebagai kombinasi nilai terbaik yang didapat dari uji coba sebelumnya. Pengujian variabel jumlah populasi dilakukan dengan skenario 5, 10, 15, 20, 25, 50 dan 100 sebagaimana dipaparkan pada Bab III. Pengujian dilakukan sebanyak 10 kali untuk setiap skenario.

Hasil pengujian menunjukkan bahwa nilai *fitness* pada setiap generasi tidak terlalu berbeda jauh. Rata-rata untuk seluruh skenario berada pada 0,58. Namun pada skenario populasi 20 didapatkan *fitness* terbaik dengan nilai *fitness* 0,5926. Sedangkan scenario populasi 5 menjadi skenario yang memiliki nilai *fitness* terendah dengan nilai *fitness* 0,57584. Pergerakan nilai *fitness* untuk uji coba variabel populasi dapat diamati pada Gambar 4.15

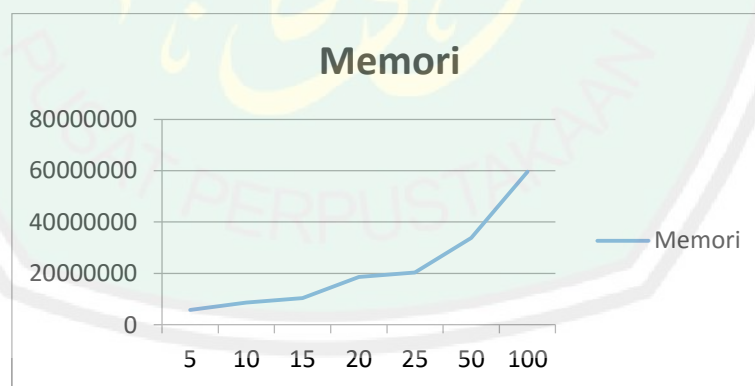


Gambar 4. 15 Pergerakan Nilai Fitness Berdasarkan Populasi

Berbeda dengan nilai *fitness* yang tidak teratur, waktu komputasi dan memori yang digunakan selalu mengalami kenaikan. Rata-rata waktu komputasi yang dibutuhkan untuk satu individu adalah 10 detik, sedangkan rata-rata memori yang dibutuhkan untuk satu individu adalah 820038 *byte*. Pergerakan waktu komputasi berdasarkan populasi dapat diamati pada Gambar 4.16, sedangkan Gambar 4.17 menggambarkan pergerakan penggunaan memori berdasarkan populasi.



Gambar 4. 16 Pergerakan Waktu Komputasi Berdasarkan Populasi



Gambar 4. 17 Pergerakan Penggunaan Memori Berdasarkan Populasi

Penentuan skenario terbaik dilakukan dengan pemberian poin untuk setiap kategori berdasarkan *rank* yang diperoleh setiap skenario pada setiap kategori. Setelah pemberian poin didapati bahwa skenario terbaik diraih oleh jumlah populasi

10 yang memperoleh total skor 18 dengan rincian: 6 poin pada kategori *fitness* terbaik kedua, 6 poin pada kategori waktu tercepat kedua dan 6 poin pada kategori penggunaan memori paling sedikit kedua. Detail perhitungan skor untuk setiap skenario dapat diamati pada Tabel 4.6.

Tabel 04.6 Skor Setiap Skenario Uji Coba Populasi

Crossover Rate	Mutasi Rate	Populasi	Fitness	Waktu	Memori	Skor
0,5	0,2	5	7 (+1)	1 (+7)	1 (+7)	15
0,5	0,2	10	2 (+6)	2 (+6)	2 (+6)	18
0,5	0,2	15	6 (+2)	3 (+5)	3 (+5)	12
0,5	0,2	20	1 (+7)	4 (+4)	4 (+4)	15
0,5	0,2	25	5 (+3)	5 (+3)	5 (+3)	9
0,5	0,2	50	4 (+4)	6 (+2)	6 (+2)	8
0,5	0,2	100	3 (+3)	7 (+1)	7 (+1)	7

4.5 Perbandingan Hasil Uji Coba Variabel Algoritma Genetika

Pengujian variabel dilakukan untuk mendapatkan variabel algoritma genetika yang optimal. Guna membuktikan variabel hasil pengujian lebih optimal berikut perbandingan hasil algoritma ditampilkan pada Tabel 4.7. Terlihat bahwa variabel terbaik hasil uji coba memberikan hasil lebih optimal. Terbukti dengan *fitness* yang lebih baik, sedangkan waktu komputasi lebih cepat dan penggunaan memori lebih sedikit.

Tabel 04.7 Perbandingan Hasil Algoritma Genetika

Variabel		Awal	Akhir
Populasi		10	10
Crossover Rate		0,5	0,5
Mutasi Rate		0,5	0,2
Maksimal Generasi		1.000	1.000
<i>Hard Constraint</i>	Tabrakan	66	59
	Libur	0	0
<i>Soft Constraint</i>	Guru	44	36
	Pelajaran	28	32

Fitness	0,6021	0,6287
Lama Komputasi	146,087	93,558
Penggunaan Memori	6.907.000	6.864.288

4.6 Pengujian Beberapa Kelas

Pengujian sebelumnya membuktikan bahwa algoritma genetika belum dapat menyusun jadwal terbaik untuk 35 kelas sekaligus. Sehingga dilakukan pengujian dengan jumlah kelas yang lebih kecil untuk membuktikan kemampuan algoritma genetika untuk menyusun jadwal. Pengujian banyak kelas dilakukan dengan menyusun jadwal untuk 2 kelas, 3 kelas dan 4 kelas. Maksimal generasi pada pengujian ini adalah 1000, sehingga tidak dapat dijadikan sebagai hasil akhir. Pengujian hanya memberikan gambaran bagaimana kinerja algoritma genetika dalam menyusun jadwal pelajaran.

Skenario pengujian yang dilakukan untuk setiap kelas sama dengan skenario pengujian sebelumnya. Pengujian dilakukan dengan dua tahap, tahapan pertama dengan *crossover rate* dan mutasi *rate* seragam. Hasil terbaik dari tahapan pertama akan digunakan pada tahapan kedua dengan *crossover rate* dan mutasi *rate* tidak seragam. Untuk setiap skenario dilakukan tiga kali percobaan. Perhitungan skenario optimal dilakukan dengan sistem ranking dan poin sama dengan perhitungan sebelumnya.

4.6.1 Pengujian Dua Kelas

Hasil pengujian tahap pertama dengan *crossover rate* dan mutasi *rate* seragam dapat diamati pada Tabel 4.8. Secara umum terlihat bahwa algoritma genetika dapat menyusun jadwal dengan rata-rata *fitness* diatas 0,9 bahkan mencapai *fitness* 1. Lama komputasi yang dibutuhkan untuk menyusun jadwal

untuk dua kelas tidak mencapai 1 detik. Adapun perhitungan *rank* dan *score* untuk 10 skenario tahap pertama dapat diamati pada Tabel 4.9.

Tabel 04.8 Hasil Percobaan Pertama Dua Kelas

Skenario	CR	MR	<i>Fitness</i>	Lama Komputasi	Memori
1	0,1	0,1	1	0,391053333	2422320
2	0,2	0,2	0,999866667	0,192256667	2361325,333
3	0,3	0,3	0,9999	0,604656667	2436122,667
4	0,4	0,4	0,9999	0,353776667	2368874,667
5	0,5	0,5	0,999833333	0,172606667	2334066,667
6	0,6	0,6	0,999933333	0,49786	2364704
7	0,7	0,7	0,9999	0,936553333	2408834,667
8	0,8	0,8	0,999933333	0,538176667	2361016
9	0,9	0,9	0,999933333	0,439773333	2341738,667
10	1	1	0,999833333	0,45427	2350778,667

Skenario 1 menghasilkan jadwal dengan rata-rata nilai *fitness* tertinggi dengan nilai *fitness* 1, rata-rata lama komputasi 0,39 detik dan rata-rata penggunaan memori 2422320 *byte*. Skenario dengan rata-rata waktu komputasi tercepat diraih oleh skenario 5 dengan rata-rata lama komputasi 0,172606667 dan rata-rata penggunaan memori 2334066 *byte* juga sebagai skenario dengan penggunaan memori paling sedikit dan mencapai nilai *fitness* 0,999833333. Skenario 5 memperoleh total skor 26 sebagai skenario terbaik pada tahap pengujian pertama. Skor untuk setiap skenario dapat diamati pada Tabel 4.9.

Tabel 04.9 Skor Setiap Skenario Tahap Pertama Dua Kelas

Skenario	Skor <i>Fitness</i>	Skor Waktu	Skor Memori	Total
1	10	7	2	19
2	7	9	6	22
3	8	2	1	11
4	8	8	4	20
5	6	10	10	26

6	9	4	5	18
7	8	1	3	12
8	9	3	7	19
9	9	6	9	24
10	6	5	8	19

Setelah mendapatkan kombinasi *crossover rate* dan *mutasi rate* seragam terbaik untuk penjadwalan dua kelas, nilai *crossover rate* dan *mutasi rate* akan digunakan untuk pengujian tahap kedua dengan nilai tidak seragam. Secara keseluruhan kombinasi *crossover rate* dan *mutasi rate* dapat menyusun jadwal dengan *fitness* mencapai 0,9 dan bahkan skenario 28 dengan kombinasi *crossover rate* 0,9 dan *mutasi rate* 0,5 dapat mencapai nilai *fitness* 1. Hasil pengujian untuk percobaan tahap kedua dapat diamati pada Tabel 4.10.

Tabel 04.10 Hasil Percobaan Tahap Kedua Dua Kelas

Skenario	CR	MR	<i>Fitness</i>	Lama Komputasi	Memori
11	0,5	0,1	0,999833333	0,46125	2421970,667
12	0,5	0,2	0,999833333	0,622926667	2407984
13	0,5	0,3	0,9998	0,548843333	2393405,333
14	0,5	0,4	0,999866667	0,748523333	2409352
15	0,5	0,5	0,999833333	0,172606667	2334066,667
16	0,5	0,6	0,9998	0,888803333	2428114,667
17	0,5	0,7	0,9999	0,642026667	2379920
18	0,5	0,8	0,9999	1,95606	2544200
19	0,5	0,9	0,999766667	0,378443333	2344608
20	0,5	1	0,999966667	0,56249	2373122,667
21	0,1	0,5	0,999933333	0,46502	2399122,667
22	0,2	0,5	0,9999	0,46282	2387744
23	0,3	0,5	0,999833333	0,748526667	2412586,667
24	0,4	0,5	0,9999	0,701746667	2409480
25	0,6	0,5	0,999833333	0,405923333	2365856
26	0,7	0,5	0,9999	1,69241	2595677,333
27	0,8	0,5	0,9998	0,43224	2379685,333
28	0,9	0,5	1	0,073993333	2320440
29	1	0,5	0,9999	0,41274	2351466,667

Skenario 28 menghasilkan jadwal dengan nilai *fitness* tertinggi dengan nilai *fitness* 1, lama komputasi 0,07 detik dan penggunaan memori 2320440 *byte*. Skenario dengan waktu komputasi tercepat juga diraih oleh skenario 28 dan penggunaan memori paling sedikit juga diraih oleh skenario 28. Skenario 28 memperoleh total skor 50 sebagai skenario terbaik pada tahap pengujian kedua. Skor untuk setiap skenario dapat diamati pada Tabel 4.11. Kombinasi *crossover rate* dan *mutasi rate* terbaik untuk dua kelas adalah *crossover rate* 0,9 dan *mutasi rate* 1.

Tabel 04.11 Skor Setiap Skenario Tahap Kedua Dua Kelas

Skenario	Skor <i>Fitness</i>	Skor Waktu	Skor Memori	Total
11	5	14	5	24
12	5	9	9	23
13	4	11	11	26
14	6	6	8	20
15	5	19	19	43
16	4	4	4	12
17	7	8	13	28
18	7	2	3	12
19	3	18	18	39
20	9	10	15	34
21	8	12	10	30
22	7	13	12	32
23	5	5	6	16
24	7	7	7	21
25	5	17	16	38
26	7	3	2	12
27	4	15	14	33
28	10	20	20	50
29	7	16	17	40

4.6.2 Pengujian Tiga Kelas

Hasil pengujian untuk penjadwalan tiga kelas tahap pertama dengan *crossover rate* dan *mutasi rate* seragam dapat diamati pada Tabel 4.12. Secara umum terlihat bahwa algoritma genetika dapat menyusun jadwal dengan *fitness* diatas 0,9 meski tidak ada yang mencapai *fitness* 1. Lama komputasi yang dibutuhkan untuk menyusun jadwal untuk tiga kelas tidak mencapai 1 menit. Adapun perhitungan *rank* dan *score* untuk 10 skenario tahap pertama dapat diamati pada Tabel 4.13.

Tabel 04.12 Hasil Percobaan Pertama Tiga Kelas

Skenario	CR	MR	<i>Fitness</i>	Lama Komputasi	Memori
1	0,1	0,1	0,9771	15,71152	5635248
2	0,2	0,2	0,986666667	18,13938	5591842,667
3	0,3	0,3	0,977133333	21,09286667	5598688
4	0,4	0,4	0,9709	23,58378333	5563128
5	0,5	0,5	0,980166667	26,16591333	5532762,667
6	0,6	0,6	0,970833333	28,92945	5540106,667
7	0,7	0,7	0,983566667	31,36658	5532506,667
8	0,8	0,8	0,977166667	33,84601	5524530,667
9	0,9	0,9	0,9803	36,82700667	5517461,333
10	1	1	0,980366667	39,29422	5521960

Skenario 2 menghasilkan jadwal dengan nilai *fitness* terbaik dengan rata-rata nilai *fitness* 0,986, rata-rata lama komputasi 18,13 detik dan rata-rata penggunaan memori 5591842 *byte*. Skenario dengan waktu komputasi tercepat diraih oleh skenario 1 dengan rata-rata lama komputasi 15,7 detik. Sedangkan skenario dengan penggunaan memori paling sedikit diraih oleh skenario 9 dengan rata-rata penggunaan memori 5517461 *byte*. Skenario 2 memperoleh total skor 22

sebagai skenario terbaik pada tahap pengujian pertama. Skor untuk setiap skenario dapat diamati pada Tabel 4.13.

Tabel 04.13 Skor Setiap Skenario Tahap Pertama Tiga Kelas

Skenario	Skor <i>Fitness</i>	Skor Waktu	Skor Memori	Total
1	3	10	1	14
2	10	9	3	22
3	4	8	2	14
4	2	7	4	13
5	6	6	6	18
6	1	5	5	11
7	9	4	7	20
8	5	3	8	16
9	7	2	10	19
10	8	1	9	18

Setelah mendapatkan kombinasi *crossover rate* dan *mutasi rate* seragam terbaik untuk penjadwalan tiga kelas, nilai *crossover rate* dan *mutasi rate* akan digunakan untuk pengujian tahap kedua dengan nilai tidak seragam. Secara keseluruhan kombinasi *crossover rate* dan *mutasi rate* dapat menyusun jadwal dengan *fitness* mencapai 0,9. Hasil pengujian untuk percobaan tahap kedua dapat diamati pada Tabel 4.14.

Tabel 04.14 Hasil Percobaan Kedua Tiga Kelas

Skenario	CR	MR	<i>Fitness</i>	Lama Komputasi	Memori
11	0,2	0,1	0,9771	15,34840667	5601152
12	0,2	0,3	0,9805	14,73114333	4678432
13	0,2	0,4	0,983466667	23,84387667	5594784
14	0,2	0,5	0,974066667	26,13199333	5604405,333
15	0,2	0,6	0,983466667	28,84510667	5575640
16	0,2	0,7	0,977233333	31,48435667	5608317,333
17	0,2	0,8	0,9771	34,04516	5603506,667

18	0,2	0,9	0,9803	36,65889	5613202,667
19	0,2	1	0,977066667	39,52335667	5588728
20	0,1	0,2	0,977066667	17,96185333	5615749,333
21	0,3	0,2	0,977166667	18,03426667	5576890,667
22	0,4	0,2	0,9771	18,02706333	5570069,333
23	0,5	0,2	0,980333333	17,97386333	5561701,333
24	0,6	0,2	0,970733333	17,98159	5539992
25	0,7	0,2	0,9739	18,06653667	5513152
26	0,8	0,2	0,986833333	20,16976	4636018,667
27	0,9	0,2	0,9771	25,80399667	5510165,333
28	1	0,2	0,9771	17,97997	5517021,333
29	0,2	0,2	0,986666667	18,13938	5591842,667

Skenario 26 menghasilkan jadwal dengan rata-rata nilai *fitness* tertinggi dengan nilai *fitness* 0,98, rata-rata lama komputasi 20,16 detik dan rata-rata penggunaan memori 4636018 *byte*. Skenario dengan waktu komputasi tercepat diraih oleh skenario 12 dengan rata-rata waktu komputasi 14,7 detik. Sedangkan skenario dengan rata-rata penggunaan memori paling sedikit diraih oleh skenario 26 dengan rata-rata penggunaan memori 4636018 *byte*. Skenario 12 memperoleh total skor 56 sebagai skenario terbaik pada tahap pengujian kedua. Skor untuk setiap skenario dapat diamati pada Tabel 4.15. Kombinasi *crossover rate* dan *mutasi rate* terbaik untuk dua kelas adalah *crossover rate* 0,2 dan *mutasi rate* 0,3.

Tabel 04.15 Skor Setiap Skenario Tahap Pertama Tiga Kelas

Skenario	Skor <i>Fitness</i>	Skor Waktu	Skor Memori	Total
11	12	19	7	38
12	17	20	19	56
13	18	9	8	35
14	10	7	5	22
15	18	6	12	36
16	14	5	4	23
17	12	4	6	22
18	15	3	3	21

19	11	2	10	23
20	11	18	2	31
21	13	13	11	37
22	12	14	13	39
23	16	17	14	47
24	8	15	15	38
25	9	12	17	38
26	20	10	20	50
27	12	8	18	38
28	12	16	16	44
29	19	11	9	39

4.6.3 Pengujian Empat Kelas

Hasil pengujian untuk penjadwalan empat kelas tahap pertama dengan *crossover rate* dan *mutasi rate* seragam dapat diamati pada Tabel 4.16. Secara umum terlihat bahwa algoritma genetika dapat menyusun jadwal dengan *fitness* diatas 0,8 meski tidak ada yang mencapai *fitness* 1. Lama komputasi yang dibutuhkan untuk menyusun jadwal untuk empat kelas tidak mencapai 1 menit. Adapun perhitungan *rank* dan *score* untuk 10 skenario tahap pertama dapat diamati pada Tabel 4.17.

Tabel 04.16 Hasil Percobaan Pertama Empat Kelas

Skenario	CR	MR	<i>Fitness</i>	Lama Komputasi	Memori
1	0,1	0,1	0,909	21,81965667	5790944
2	0,2	0,2	0,914566667	24,47425667	5713288
3	0,3	0,3	0,8955	27,95016	5699781,333
4	0,4	0,4	0,917333333	31,84186333	5660936
5	0,5	0,5	0,911833333	34,48262333	5669664
6	0,6	0,6	0,911733333	37,15247	5660717,333
7	0,7	0,7	0,911666667	40,79007	5664776
8	0,8	0,8	0,9175	44,60815	5622872
9	0,9	0,9	0,922966667	48,19286333	5655285,333
10	1	1	0,903633333	52,02455667	5633109,333

Skenario 9 menghasilkan jadwal dengan nilai *fitness* terbaik dengan rata-rata nilai *fitness* 0,92, rata-rata lama komputasi 48,19 detik dan rata-rata penggunaan memori 5655285 *byte*. Skenario dengan waktu komputasi tercepat diraih oleh skenario 1 dengan rata-rata lama komputasi 21,8 detik. Sedangkan skenario dengan penggunaan memori paling sedikit diraih oleh skenario 8 dengan rata-rata penggunaan memori 5622872 *byte*. Skenario 8 dengan kombinasi *crossover rate* dan *mutasi rate* 0,8 memperoleh total skor 22 sebagai skenario terbaik pada tahap pengujian pertama. Skor untuk setiap skenario dapat diamati pada Tabel 4.17.

Tabel 04.17 Skor Setiap Skenario Tahap Pertama Empat Kelas

Skenario	Skor <i>Fitness</i>	Skor Waktu	Skor Memori	Total
1	3	10	1	14
2	7	9	2	18
3	1	8	3	12
4	8	7	6	21
5	6	6	4	16
6	5	5	7	17
7	4	4	5	13
8	9	3	10	22
9	10	2	8	20
10	2	1	9	12

Setelah mendapatkan kombinasi *crossover rate* dan *mutasi rate* seragam terbaik untuk penjadwalan empat kelas, nilai *crossover rate* dan *mutasi rate* akan digunakan untuk pengujian tahap kedua dengan nilai tidak seragam. Secara keseluruhan kombinasi *crossover rate* dan *mutasi rate* dapat menyusun jadwal dengan *fitness* mencapai 0,9. Hasil pengujian untuk percobaan tahap kedua dapat diamati pada Tabel 4.18.

Tabel 04.18 Hasil Percobaan Tahap Kedua Empat Kelas

Skenario	CR	MR	<i>Fitness</i>	Lama Komputasi	Memori
8	0,8	0,8	0,9175	44,60815	5622872
11	0,1	0,8	0,911733333	44,69250667	5782301,333
12	0,2	0,8	0,911833333	44,96432333	5728458,667
13	0,3	0,8	0,908866667	44,68972333	5682272
14	0,4	0,8	0,9063	44,8469	5695525,333
15	0,5	0,8	0,900933333	44,51785667	5638618,667
16	0,6	0,8	0,9146	44,82604333	5659434,667
17	0,7	0,8	0,925966667	44,68135333	5660128
18	0,9	0,8	0,917533333	44,34119667	5628106,667
19	1	0,8	0,917466667	44,65079333	5622394,667
20	0,8	0,1	0,917333333	19,23846333	5653898,667
21	0,8	0,2	0,917266667	22,95975333	5636117,333
22	0,8	0,3	0,914666667	26,91878667	5649693,333
23	0,8	0,4	0,9035	32,95331333	5649314,667
24	0,8	0,5	0,903566667	36,30687333	5646826,667
25	0,8	0,6	0,906166667	38,31413667	5650810,667
26	0,8	0,7	0,9062	42,60422667	5632229,333
27	0,8	0,9	0,903533333	51,04568667	5630154,667
28	0,8	1	0,900933333	52,55508	5633778,667

Skenario 17 menghasilkan jadwal dengan rata-rata nilai *fitness* tertinggi dengan nilai *fitness* 0,92, rata-rata lama komputasi 44,68 detik dan rata-rata penggunaan memori 5660128 *byte*. Skenario dengan waktu komputasi tercepat diraih oleh skenario 11 dengan rata-rata waktu komputasi 44,69 detik. Sedangkan skenario dengan rata-rata penggunaan memori paling sedikit diraih oleh skenario 19 dengan rata-rata penggunaan memori 5622394 *byte*. Skenario 19 memperoleh total skor 47 sebagai skenario terbaik pada tahap pengujian kedua. Skor untuk setiap skenario dapat diamati pada Tabel 4.19. Kombinasi *crossover rate* dan *mutasi rate* terbaik untuk dua kelas adalah *crossover rate* 0,2 dan *mutasi rate* 0,3.

Tabel 04.19 Skor Setiap Skenario Tahap Kedua Empat Kelas

Skenario	Skor <i>Fitness</i>	Skor Waktu	Skor memori	Total
11	11	19	2	32
12	12	18	3	33
13	10	17	5	32
14	9	16	4	29
15	3	15	13	31
16	13	14	7	34
17	19	13	6	38
18	18	12	18	48
19	17	11	19	47
20	16	10	8	34
21	15	9	14	38
22	14	8	10	32
23	4	7	11	22
24	6	6	12	24
25	7	5	9	21
26	8	4	16	28
27	5	3	17	25
28	3	2	15	20

4.7 Perbandingan Dengan Penelitian Sebelumnya

Sari (2015) melakukan penelitian optimasi penjadwalan mata pelajaran menggunakan algoritma genetika dengan studi kasus SMP N 1 Gondang Mojokerto. Dalam penelitian tersebut hanya menetapkan satu *constrain* berupa tabrakan jadwal mengajar guru. Parameter yang diuji pada penelitian tersebut adalah ukuran populasi dan kombinasi *crossover rate* dan *mutation rate*. Sedangkan parameter yang digunakan untuk menentukan hasil terbaik hanya nilai *fitness*. Hasil penelitian tersebut menyimpulkan bahwa algoritma genetika tidak dapat menghasilkan jadwal dengan nilai *fitness* 1 atau tidak terdapat tabrakan jam mengajar. Kesimpulan lain yang didapat adalah nilai parameter algoritma genetika berpengaruh terhadap hasil optimasi. Parameter yang kecil menyebabkan area pencarian semakin sempit, sedangkan parameter yang besar membutuhkan waktu

komputasi lebih lama dan tidak menjamin menghasilkan nilai yang optimal. Sari tidak menyimpulkan kombinasi parameter terbaik pada penelitiannya.

Suwirmayanti (2016) juga melakukan penelitian yang serupa dengan *constraint* berupa tabrakan jam mengajar. Penelitian yang dilakukan Suwirmayanti melakukan pengujian pada parameter algoritma genetika berupa ukuran populasi atau banyak kromosom, *crossover rate* dan *mutation rate*. Penelitian tersebut menghasilkan aplikasi yang dapat digunakan untuk menyelesaikan penjadwalan mata pelajaran dengan waktu lebih singkat dibandingkan dengan melakukan penyusunan jadwal secara manual. Kombinasi parameter algoritma genetika terbaik yang disarankan dalam penelitian tersebut adalah banyak kromosom 80, *crossover rate* 0,4 dan *mutation rate* 0,001.

Saputra (2019) melakukan penelitian serupa dengan studi kasus SMP N 1 Peunaron. Constrain yang digunakan dalam penelitiannya juga hanya berupa tabrakan jam mengajar guru. Hasil penelitian Saputra adalah aplikasi penjadwalan yang dikembangkan dengan Visual Basic Net 2008. Saputra tidak melakukan pengujian parameter algoritma genetika dalam penelitiannya, juga tidak disimpulkan apakah algoritma genetika dapat menyusun jadwal pelajaran dengan sempurna atau tidak.

Perbedaan penelitian ini dengan penelitian sebelumnya terdapat dalam beberapa hal. Perbedaan pertama adalah *constrain* yang digunakan, dimana penelitian sebelumnya hanya menggunakan satu *constrain* berupa tabrakan jadwal mengajar. Sedangkan penelitian ini menggunakan dua bentuk *constrain*, *hard constrain* dan *soft constrain*. Dimana *hard constrain* terdiri dari tabrakan jam

mengajar guru dan terjadwalnya guru pada hari dimana guru tersebut mengambil libur. Sedangkan *soft constrain* pertama adalah terjadwalnya guru yang sama dalam satu kelas di hari yang sama. *Soft constrain* kedua adalah terjadwalnya pelajaran yang sama di hari yang sama dalam satu kelas.

Penelitian ini juga melanjutkan penelitian sebelumnya dengan melakukan pengujian parameter algoritma genetika lebih banyak dan beragam daripada yang telah dilakukan sebelumnya. Parameter tidak diuji secara terpisah sebagaimana yang dilakukan Sari dalam penelitiannya, melainkan pengujian dilakukan dengan keterkaitan antar parameter. Penelitian ini menguatkan kesimpulan dari penelitian sebelumnya yang mengatakan bahwa terdapat keterkaitan antara parameter algoritma genetika dengan tingkat keoptimalan algoritma genetika.

Perbedaan terakhir terdapat pada parameter yang digunakan untuk menentukan keoptimalan algoritma genetika. Penelitian sebelumnya hanya menggunakan nilai *fitness* untuk menentukan tingkat keoptimalan algoritma genetika, berbeda dengan penelitian ini yang menggunakan 3 parameter untuk mengukur keoptimalan algoritma genetika. Ketiga parameter tersebut adalah nilai *fitness*, waktu komputasi dan penggunaan memori.

4.8 Kajian Integrasi Islam dan Sains

Al-Quran surat al-Ashr mengingatkan manusia tentang betapa penting dan berharganya waktu. Allah menegaskan bahwa seluruh manusia dalam kerugian, kecuali mereka yang beriman dan beramal shaleh. Termasuk dalam orang-orang yang merugi adalah orang yang tidak dapat memanfaatkan waktu dengan baik. Al-

Quran surat Az-Zumar [39] ayat 56 menerangkan bahwa orang yang tidak memanfaatkan waktu dengan baik adalah orang yang merugi.

وَالْعَصْرِ
إِنَّ الْإِنْسَانَ لَفِي خُسْرٍ إِلَّا الَّذِينَ آمَنُوا وَعَمِلُوا الصَّالِحَاتِ وَتَوَاصَوْا بِالْحَقِّ وَتَوَاصَوْا بِالصَّبْرِ

Artinya : “Demi masa, sungguh, manusia berada dalam kerugian, kecuali orang-orang yang beriman dan mengerjakan kebajikan serta saling menasihati untuk kebenaran dan saling menasihati untuk kesabaran.” (Al-Quran Surat Al-Ashr ayat 1-3)

Orang yang merugi adalah orang yang tidak memanfaatkan waktu yang Allah berikan untuk beramal shaleh. Surat Az-Zumar ayat 56 menjelaskan bahwa waktu yang diberikan Allah wajib dikelola dengan sebaik mungkin, agar tidak menjadi golongan orang yang menyesal. Untuk memanfaatkan waktu dengan baik diperlukan manajemen waktu yang baik, salah satunya adalah dengan menjadwalkan rutinitas dengan baik. Dengan menjadwalkan kegiatan dengan baik dan mentaatinya maka disiplin waktu akan terwujud.

مَنْ وَأُولَئِكَ الْخَيْرَاتُ فِي وَيُسَارِعُونَ الْمُتَكِرِّ عَنْ وَيَنْهَوْنَ بِالْمَعْرُوفِ وَيَأْمُرُونَ الْآخِرِ وَالْيَوْمِ بِاللَّهِ يُؤْمِنُونَ
الصَّالِحِينَ

Artinya : “Mereka beriman kepada Allah dan hari akhir, menyuruh (berbuat) yang makruf, dan mencegah dari yang mungkar dan bersegera (mengerjakan) berbagai kebajikan. Mereka termasuk orang-orang saleh.” (Al-Quran Surat Ali Imran ayat 114)

Quran surat Ali Imran ayat 114 juga menjelaskan ciri orang yang shaleh adalah beriman kepada hari akhir, menyuruh kepada kebaikan, mencegah dari

kemungkaran dan bersegera mengerjakan kebajikan. Dalam ayat tersebut disebutkan bersegera mengerjakan kebajikan, dimana kata bersegera bermakna untuk tidak menia-nyiakan waktu. Salah satu cara agar dapat bersegera melakukan kebajikan adalah dengan melakukan penjadwalan yang baik, sebagaimana yang diusahakan dalam penelitian ini. Penelitian ini selain bertujuan menyusun jadwal pelajaran juga berusaha mempersingkat waktu yang dibutuhkan untuk menyusun sebuah jadwal.



BAB V KESIMPULAN DAN SARAN

5.1 Kesimpulan

Penelitian yang telah dilakukan membuktikan bahwa algoritma genetika dapat digunakan untuk menyusun jadwal pelajaran dengan cukup baik. Adapun kombinasi variabel paling optimal yang didapatkan dari hasil pengujian adalah berbeda untuk setiap kasus penjadwalan. Dari hasil pengujian didapatkan hasil sebagai berikut: *crossover rate* 0,5, *mutasi rate* 0,2 dan jumlah populasi 10 merupakan kombinasi terbaik untuk menyusun jadwal 35 kelas. Sedangkan untuk menyusun jadwal 2 kelas kombinasi terbaiknya adalah *crossover rate* 0,9 dan *mutasi rate* 0,5. *Crossover rate* 0,2 dan *mutasi rate* 0,3 merupakan kombinasi terbaik untuk menyusun jadwal untuk 3 kelas. Sedangkan untuk menyusun jadwal 4 kelas kombinasi terbaiknya adalah *crossover rate* 1 dan *mutasi rate* 0,8.

5.2 Saran

Penelitian selanjutnya dapat melakukan pengujian pada perbandingan bobot error soft constraint dan hard constraint pada metode algoritma genetika. Pengujian kombinasi juga perlu diperbanyak dengan kombinasi *crossover rate* dan *mutasi rate* yang lebih beragam. Penelitian ini juga dapat dilanjutkan dengan menambahkan metode pencarian lain untuk menyempurnakan hasil dari algoritma genetika, seperti Tabu Search dan lainnya. Algoritma genetika adalah algoritma yang cukup random, sehingga bisa juga menggunakan algoritma lainnya untuk mencoba menentukan parameter algoritma genetika. Algoritma genetika sendiri bisa digunakan untuk menentukan parameter terbaik untuk algoritma genetika yang digunakan untuk menyusun penjadwalan.

DAFTAR PUSTAKA

- Admi, S., dan Gen, M., (2003). *Double Spanning Tree-based Genetic Algorithm for Two Stage Transportation Problem, the International Journal of Knowledge-based Engineering System, Vol 7, No 4, 214-221*
- Admodiwiro, Soebagio. 2000. *Manajemen Pendidikan*, Jakarta: PT Ardadizya.
- Arifudin, Riza (2011). *Optimasi Penjadwalan Proyek Dengan Penyeimbangan Biaya Menggunakan Kombinasi CPM dan Algoritma Genetika. Jurnal Masyarakat Informatika, Volume 2, Nomor 4, ISSN 2086 – 4930. Doi: 10.14710/jmasif.2.4.1-14*
- Aulia, Indra, Erna Budhiarti N dan Muhammad Anggia M. (2012). *Penerapan Harmony Search Algorithm dalam Permasalahan Penjadwalan Flow Shop. Jurnal Dunia Teknologi Informasi Vol. 1, No. 1, hal 1-7*
- Beligiannis, G. N., dkk. (2008). *Applying evolutionary computation to the school timetabling problem: the Greek case. Computers and Operations Research, 35, 1265–1280. Doi: 10.1016/j.cor.2006.08.010*
- Budhi, G. S., Gunadi, K., & Wibowo, D. A. (2015). *Genetic Algorithm for Scheduling Courses. Intelligence in the Era of Big Data, 51–63. doi:10.1007/978-3-662-46742-8_5*
- Chang Wook Ahn, & Ramakrishna, R. S. (2002). *A genetic algorithm for shortest path routing problem and the sizing of populations. IEEE Transactions on Evolutionary Computation, 6(6), 566–579. doi:10.1109/tevc.2002.804323*
- Dwivedi, Varshika dkk, (2012). *Travelling Salesman Problem using Genetic Algorithm. International Journal of Computer Applications (IJCA). India.*

- Gen, M. dan R. Cheng., (2000). *Genetic Algorithms and Engineering Optimization*. New York: John Wiley & Sons.
- Hidayatulloh, Taufik. (2015). *Perancangan Sistem Penjadwalan Pembelajaran Menggunakan Graph Coloring*. *Jurnal Informatika*. Vol. II, No. 2. doi:10.31311/ji.v2i2.123
- Pillay, N. (2014). A survey of school timetabling research. *Ann Oper Res* 218:261. Doi:10.1007/s10479-013-1321-8
- Pinedo, Michael L. (2012) *Scheduling Teori, Algorithms and Systems*. New York: Springer. doi: 10.1007/978-1-4614-2361-4
- Saputra, Edi, dkk. (2019). Perancangan Aplikasi Penjadwalan Mata Pelajaran Dengan Menerapkan Algoritma Genetika pada SMP N 1 Peunaron. Medan: Jurnal Pelita Informatika. ISSN 2301-9425. Hal 117-123.
- Sari, DDP, Mahmudy, WF & Ratnawati, DE (2015) Optimasi Penjadwalan Mata Pelajaran Menggunakan Algoritma Genetika (Studi Kasus : SMP N 1 Gondang Mojokerto) DORO: Repository Jurnal Mahasiswa PTIIK Universitas Brawijaya, vol 5, no. 13
- Schaerf, Andrea. (1999). *A survey of Automated Timetabling*. *Artificial Intelligence Review* 13: 87–12. Netherlands: Kluwer Academic Publishers
- Suwirmayanti, NLGP, Sudarsana, I Made & Darmayasa, Suta (2016) Penerapan Algoritma Genetika Untuk Penjadwalan Mata Pelajaran. *Jurnal of Applied Intelligent System*, Vol.1, No.3, Oktober 2016 : 220-233
- Lesmana, Nunung Indra. (2016). *Penjadwalan Produksi Untuk Meminimalkan Waktu Produksi Dengan Menggunakan Metode Branch And Bound*. *Jurnal Teknik Industri*, Vol. 17, No. 1, pp. 42-50 ISSN 2527-4112

Valouxis, C., & Housos, E. (2003). *Constraint programming approach for school timetabling*. *Computers and Operations Research*, 30, 1555–1572, doi: 10.1016/S0305-0548(02)00083-7

