TUGAS AKHIR – KS 141501

OPTIMASI PENJADWALAN MATA KULIAH OTOMATIS DENGAN MENGGUNAKAN ALGORITMA *ITERATED LOCAL SEARCH - HYPER HEURISTIC*

NUR ACHMAD SETIADI
NRP 05211440000084

Dosen Pembimbing :
Ahmad Muklason, S.Kom., M.Sc., Ph.D.

DEPARTEMEN SISTEM INFORMASI
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

TUGAS AKHIR – KS 141501

OPTIMASI PENJADWALAN MATA KULIAH OTOMATIS DENGAN MENGGUNAKAN ALGORITMA ITERATED LOCAL SEARCH – HYPER HEURISTIC

NUR ACHMAD SETIADI
NRP 05211440000084

Dosen Pembimbing :
Ahmad Muklason, S.Kom., M.Sc., Ph.D.

DEPARTEMEN SISTEM INFORMASI
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

Halaman ini sengaja dikosongkan.

FINAL PROJECT – KS 141501

AUTOMATED COURSE TIMETABLING OPTIMIZATION USING *ITERATED LOCAL SEARCH – HYPER HEURISTIC ALGORITHM*

NUR ACHMAD SETIADI
NRP 05211440000084

Supervisor :
Ahmad Muklason, S.Kom., M.Sc., Ph.D.

DEPARTMENT OF INFORMATION SYSTEMS
Faculty of Information Technology and Communication
Sepuluh Nopember Institute of Technology
Surabaya 2018

Halaman ini sengaja dikosongkan.

LEMBAR PENGESAHAN

OPTIMASI PENJADWALAN MATA KULIAH OTOMATIS DENGAN MENGGUNAKAN ALGORITMA *ITERATED LOCAL SEARCH – HYPER HEURISTIC*

TUGAS AKHIR

Disusun Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Departemen Sistem Informasi
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember

Oleh:

NUR ACHMAD SETIADI
NRP. 05211440000084

Surabaya, Juli 2018

KEPALA
DEPARTEMEN SISTEM INFORMASI

Dr. Ir. Aris Tjahyanto, M.Kom.
NIP.19650310 199102 1 001

Halaman ini sengaja dikosongkan

LEMBAR PERSETUJUAN

OPTIMASI PENJADWALAN MATA KULIAH OTOMATIS DENGAN MENGGUNAKAN ALGORITMA *ITERATED LOCAL SEARCH – HYPER HEURISTIC*

TUGAS AKHIR

Disusun Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Departemen Sistem Informasi
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember

Oleh :

NUR ACHMAD SETIADI
NRP. 05211440000084

Disetujui Tim Penguji : Tanggal Ujian : Juli 2018
Periode Wisuda : September 2018

Ahmad Muklason, S.Kom., M.Sc., Ph.D **(Pembimbing I)**

Edwin Riksakomara, S.Kom., M.T. **(Penguji I)**

Faizal Mahananto, S.Kom, M.Eng., Ph.D. **(Penguji II)**

Halaman ini sengaja dikosongkan

OPTIMASI PENJADWALAN MATA KULIAH OTOMATIS DENGAN MENGGUNAKAN ALGORITMA *ITERATED LOCAL SEARCH – HYPER HEURISTIC*

Nama Mahasiswa : NUR ACHMAD SETIADI
NRP : 05211440000084
Departemen : SISTEM INFORMASI FTIK-ITS
Dosen Pembimbing : Ahmad Muklason, S.Kom., M.Sc., Ph.D

ABSTRAK

Permasalahan tentang penjadwalan mata kuliah merupakan topik yang masih banyak diselesaikan dengan pendekatan berbagai macam metode. Hal ini berguna untuk mendapatkan hasil penjadwalan yang lebih optimal. Selama ini penjadwalan masih sering dilakukan secara manual, sehingga memerlukan waktu relatif lama. Dari hal inilah aplikasi berbasis java penjadwalan otomatis diharapkan mampu menjadi pengganti dari penjadwalan manual yang dilakukan selama ini. Penggunaan aplikasi berbasis java penjadwalan mata kuliah otomatis nantinya akan mampu mepersingkat waktu pembuatan jadwal kuliah, dan memberikan waktu untuk dosen pengajar dan mahasiswa yang lebih optimal, sehinga proses belajar mengajar dapat berjalan lebih optimal. Pada tugas akhir ini bertujuan untuk membuat aplikasi berbasis java r penjadwalan mata kuliah otomatis dengan menggunakan algoritma iterated local search – hyper heuristic. Hasil dari tugas akhir ini, diharapkan dapat membantu pihak departemen sebagai penyelenggara perkuliahan untuk membuat jadwal mata kuliah yang lebih optimal. Sehingga proses belajar mengajar mampu berjalan lebih baik, bagi dosen pengajar maupun dari pihak mahasiswa.

Pada tugas akhir ini akan dihasilkan suatu aplikasi berbasis java penjadwalan yang dapat menghasilkan jadwal mata kuliah secara otomatis yang mampu meminimalkan nilai pinalti dari jadwal yang telah dibuat sebelumnya. Algoritma penjadwalan

yang digunakan pada tugas akhir ini adalah iterated local search – hyper heuristic.
Aplikasi java penjadwalan yang dibangun pada tugas akhir ini mampu menghasilkan jadwal mata kuliah otomatis dengan seluruh hard contrain terpenuhi dengan nilai pinalti 550 untuk semester gasal yang pada awalnya nilai pinalti pada jadwal manual yang didapat adalah 1579 dan pada semester genap dapat menurunkan nilai pinalti dari 1360 menjadi 598.

Kata Kunci : penjadwalan, penjadwalan mata kuliah, penjadwalan otomatis, iterated local sesarch

AUTOMATED COURSE TIMETABLING OPTIMIZATION USING *ITERATED LOCAL SEARCH – HYPER HEURISTIC ALGORITHM*

Name : NUR ACHMAD SETIADI
NRP : 05211440000084
Department : INFORMATION SYSTEMS FTIK-ITS
Supervisors : Ahmad Muklason, S.Kom., M.Sc.,Ph.D.

ABSTRACT

The problem of course timetabling is a topic that is still largely solved by various methods approach. This is useful for getting more optimal timetabling results. During this timetabling is still often done manually, so it takes a relatively long time. From this case java based timetabling application automatically expected to be a replacement of manual timetabling done so far. The use of java based applications timetabling automatic will be able to shorten the timing of making the schedule, and give time for lecturers and students more optimal, so that the learning process can run more optimally. In this final project aims to make java based application automatic course timetabling by using iterated local search - hyper heuristic algorithm. The result of this final project is expected to help the department as the lecture provider to make the optimal course schedule. So that the learning process can run better, for lecturers as well as from the students.

In this final task, will be generated a java based course timetabling application that can generate automatic course schedules that can minimize penalty value from the schedule that has been made before. The scheduling algorithm used in this final project is iterated local search - hyper heuristic.

The java course timetabling application built on this final project is able to generate automatic course schedule with all hard contrain fulfilled with 550 penalty value for the first semester of which initially penalty value on manual schedule

obtained is 1579 and even semester can decrease penalty value from 1360 to 598.

Keywords: timetabling, course timetabling, automatic timetabling, iterated local sesarch

KATA PENGANTAR

Segala puji bagi Tuhan Yang Maha Kuasa atas rahmat dan karunia-Nya, sehingga penulis dapat menyelesaikan laporan tugas akhir dengan judul: "OPTIMALISASI PENJADWALAN MATA KULIAH OTOMATIS DENGAN MENGGUNAKAN *ALGORITMA ITERATED LOCAL SEARCH – HYPER HEURISTIC*" Tugas akhir ini merupakan salah satu syarat kelulusan pada Jurusan Sistem Informasi, Fakultas Teknologi Informasi dan Komunikasi, Institut Teknologi Sepuluh Nopember Surabaya.

Dalam proses pengerjaan tugas akhir ini, banyak sekali bantuan yang telah penulis dapatkan baik pengetahuan, dukungan moral dan doa dari berbagai pihak. Atas berbagai bantuan tersebut, penulis mengucapkan terima kasih yang sebesar-besarnya kepada:

1. Allah Subhanahu Wa Ta'ala yang senantiasa melimpahkan berkah dan rahmat-Nya selama penulis mengerjakan Tugas Akhir.
2. Bapak Sugani dan Ibu Mastingah selaku orang tua dan wali penulis, terimakasih atas bimbingan, doa, dan motivasi yang tak pernah henti diberikan kepada penulis.
3. Bapak Ahmad Mukhlason, S.Kom, M.Sc. Ph.D. selaku dosen pembimbing yang telah meluangkan waktu dan pikiran beliau untuk membimbing dan mengarahkan penulis dalam pengerjaan tugas akhir ini.
4. Bapak Edwin Riksa Komara, S.Kom., M.T serta Bapak Faisal Mahananto, S.Kom, M.Eng, Ph.D selaku dosen

penguji yang memberikan saran ataupun kritik yang membangun dalam proses pengerjaan tugas akhir ini.

5. Kakak penulis, Nur Listiani dan Dwi Cahyaningsih yang juga selalu mendoakan penulis untuk menyelesaikan perkuliahan.
6. Zulfa Qurrotul ‘Aini, yang selalu memberi semangat kepada penulis untuk mengerjakan tugas akhir dan menyelesaikan perkuliahan.
7. Rekan satu tim penulis, Gusti Bagus Syahrani dan Redian Galih, Muhammad Iqbal I. yang telah menemani penulis dan bersedia diajak diskusi dalam pengerjaan tugas akhir.
8. Seluruh Dosen dan Karyawan yang telah memberikan ilmu dan membantu penulis selama menjalani masa perkuliahan di Jurusan Sistem Informasi ITS.
9. Teman-teman dan adik-adik dari laboratorium Rekayasa Data dan Inteligensi Bisnis (RDIB) yang telah memberikan dukungan dan bantuan kepada penulis.
10. Teman – teman OSISRIS yang selalu membantu dalam proses belajar selama perkuliahan di Departemen Sistem Informasi ITS.
11. Serta seluruh pihak yang telah membantu penulis dalam mengerjakan tugas akhir ini yang tidak mungkin disebutkan satu per satu.

Penyusunan laporan ini masih jauh dari sempurna, untuk itu saya menerima adanya kritik dan saran yang membangun untuk perbaikan di masa mendatang. Semoga buku tugas akhir ini dapat memberikan manfaat bagi pembaca.

Surabaya, Juli 2018
Penulis,

(Nur Achmad Setiadi)

Halaman ini sengaja dikosongkan.

DAFTAR ISI

Halaman ini sengaja dikosongkan.

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR GRAFIK

DAFTAR KODE PROGRAM

Halaman ini sengaja dikosongkan.

BAB I
PENDAHULUAN

Pada bab ini, akan dijelaskan tentang Latar Belakang Masalah, Perumusan Masalah, Batasan Masalah, Tujuan Tugas Akhir, Manfaat Kegiatan Tugas Akhir dan Relevansi dengan laboratorium RDIB.

1.1. Latar Belakang

Dalam dunia pendidikan, penjadwalan mata kuliah merupakan hal yang sangat penting untuk berlangsungnya kegiatan belajar. Dimana jadwal mata kuliah digunakan mahasiswa dan dosen pengajar sebagai jadwal wajib belajar ataupun mengajar yang harus diikuti selama satu semester. Jadwal mata kuliah yang ada harus disesuaikan seoptimal mungkin sehingga tidak ada jadwal yang berbenturan dari pihak mahasiswa maupun pihak dosen pengajar.

Dari hasil wawancara yang dilakukan, didapatkan bahwa pembuatan jadwal mata kuliah dilakukan secara manual dengan waktu tiga sampai empat hari. Selain itu, banyaknya revisi menjadikan proses pembuatan jadwal mata kuliah menjadi lebih lama. Belum lagi jika terdapat perubahan mendadak, maka proses ini harus diulang kembali untuk disesuaikan dengan perubahan yang terjadi. Kesulitan yang terjadi adalah pembagian waktu mengajar dosen yang harus dilakukan terlebih dahulu, dengan mempertimbangkan ruangan yang dimiliki dari departemen. Selain itu, memastikan bahwa jadwal mata kuliah departemen tidak berbebenturan dengan jadwal yang ada pada UPMB.

Terdapat beberapa metode atau pendekatan yang dapat digunakan untuk menyelesaikan permasalahan penjadwalan. Salah satu pendekatan yang ada adalah *iterated local search*. Dengan penggunaan pendekatan *iterated local search* diharapkan adanya sebuah sistem penjadwalan mata kuliah

yang mampu mengahasilkan jadwal mata kuliah yang lebih optimal dan membutuhkan waktu yang lebih singkat.

1.2. Perumusan Masalah

Berdasarkan uraian latar belakang yang telah dijelaskan, maka rumusan masalah dari tugas akhir ini, yaitu :

1. Bagaimana membuat model matematis untuk permasalahan penjadwalan mata kuliah di Departemen Teknik Industri, Institut Teknologi Sepuluh Nopember Surabaya?
2. Bagaimana membuat aplikasi berbasis *java* penjadwalan mata kuliah otomatis yang dapat memenuhi semua aturan dan *constrain* yang telah ditentukan oleh Departemen Teknik Industri, Institut Teknologi Sepuluh Nopember Surabaya?
3. Bagaimana menerapkan algoritma *iterated local search* untuk menyelesaikan permasalahan penjadwalan mata kuliah?
4. Bagaimana kinerja optimalisasi yang menerapkan algoritma *iterated local search*?
5. Bagaimana perbandingan hasil antara penjadwalan mata kuliah secara manual dan penjadwalan mata kuliah yang menerapkan algoritma *iterated local search*?

1.3. Batasan Pengerjaan Tugas Akhir

Batasan pemasalahan dalam tugas akhir ini, yaitu :

a. Data yang digunakan dalam tugas akhir ini yaitu data mata kuliah, data mahasiswa dan data ruang yang diadakan pada semester gasal dan genap tahun 2017 – 2018 di Departemen Teknik Industri, Institut Teknologi Sepuluh Nopember Surabaya.
b. Hasil dari tugas akhir ini adalah sebuah aplikasi penjadwalan mata kuliah otomatis yang dibangun dengan bahasa pemrograman *java* dan berbasis lokal.
c. Aplikasi berbasis *java* tersebut akan digunakan untuk membuat jadwal mata kuliah secara otomatis di

Departemen Teknik Industri, Institut Teknologi Sepuluh Nopember Surabaya.

1.4. Tujuan Tugas Akhir

Tujuan yang hendak dicapai dalam pengerjaan tugas akhir ini adalah membuat aplikasi berbasis *java* penjadwalan mata kuliah otomatis yang optimal untuk penjadwalan mata kuliah di Departemen Teknik Industri, Institut Teknologi Sepuluh Nopember Surabaya dengan menggunakan algoritma *iterated local search – hyper heuristic*. Aplikasi berbasis *java* penjadwalan mata kuliah otomatis tersebut diharapkan bisa menggantikan sistem manual yang digunakan sebelumnya dan menghasilkan jadwal mata kuliah yang optimal bagi mahasiswa.

1.5. Manfaat Tugas Akhir

Manfaat yang diberikan dengan adanya tugas akhir ini adalah sebagai berikut:

1. Manfaat Teoritis

   Adanya tugas akhir ini diharapkan dapat menambah khasanah keilmuan dalam bidang sistem informasi khususnya mengenai implementasi algoritma *iterated local search – hyper heuristic* untuk membuat aplikasi berbasis java penjadwalan mata kuliah otomatis.

2. Manfaat Praktis
   a. Bagi Pihak Departemen Teknik Industri Institut Teknologi Sepuluh Nopember Surabaya sebagai Klien
   b. Tugas akhir ini diharapkan dapat membantu Departemen Teknik Industri, Institut Teknologi Sepuluh Nopember Surabaya sebagai penyelenggara perkuliahan untuk membuat jadwal mata kuliah secara otomatis sehingga dapat menghemat waktu dan dapat menghasilkan jadwal mata kuliah yang optimal bagi mahasiswa. Selain itu, penelitian ini juga dapat

digunakan untuk membantu departemen ataupun perguruan tinggi yang lain dalam pembuatan jadwal mata kuliah otomatis.

3. Bagi Penelitian Berikutnya

   Tugas akhir ini diharapkan dapat memberikan motivasi bagi peneliti lain untuk melakukan pengembangan terhadap studi mengenai penjadwalan mata kuliah dengan menggunakan algoritma atau pendekatan lainnya.

1.6. Relevansi

Topik tugas akhir ini yaitu mengenai optimasi penjadwalan mata kuliah. Topik tersebut sesuai dengan bidang ilmu riset operasi yang menjadi cakupan *research roadmap* pada laboratorium Rekayasa Data dan Inteligensi Bisnis (RDIB).

BAB II
TINJAUAN PUSTAKA

Sebelum melakukan pengerjaan tugas akhir, penulis melakukan tinjauan pustaka dari beberapa penelitian sebelumnya dan dasar teori yang sesuai dengan topik tugas akhir.

2.1. Penelitian Terdahulu

Pada sub bab ini akan diterangkan mengenai beberapa penelitian terdahulu yang telah dilakukan dan memiliki relevansi dengan tugas akhir ini. Penelitian terdahulu tersebut dapat dilihat pada tabel 5.1, 5.2, dan 5.3.

Tabel 2.1 Penelitian Terdahulu (1)

Nama Peneliti	Gigih Yudha Hutama[1], Dian Kusumawardani[2], Putri Cahyaning Bwananesia[3]
Tahun Penelitian	2017
Judul Penelitian	Optimasi Penjadwalan Ujian Otomatis Dengan Menggunakan Algoritma *Greedy - Hill Climbing Hyperheuristic*, Optimalisasi Penjadwalan Ujian Otomatis Dengan Menggunakan Algoritma *Greedy Simulated Annealing Hyper-Heuristic*, Optimasi Penjadwalan Ujian Otomatis Dengan Menggunakan Algoritma *Greedy – Late Acceptance – Hyper Heuristic.*
Penjelasan Singkat	Penjadwalan ujian merupakan salat satu permasalahan yang dihadapi perguruan tinggi. Dimana peniliti menggunakan algoritma yang berbeda namun tetap berbasis *hyperheuristic* untuk membandingkan hasil optimasi penjadwalan ujian. Pada penelitian ini memperhatikan *soft constrain* dan *hard constrain* yang dimiliki departemen Sistem Informasi, ITS. Dari ketiga peneliti algoritma *Greedy Simulated Annealing Hyper Heuristic*. Dimana dengan algoritma tersebut mendapatkan jadwal yang feasible dan iterasi yang lebih sedikit diandingkan algoritma yang lain. Dari hasil yang didapat

	menunjukkan bahwa hasil yang telah dibuat dapat diimplementasikan di departemen Sistem Informasi, ITS
Hasil Penelitian	Penelitian diatas membahas tentang penjadwala ujian dengan basis *hyper heuristic*. Permasalahan yang diangkat adalah timetabling. Hal ini bisa menjadi referensi untuk pembuatan optimasi *course timetabling* dengan permasalahan ujian diganti dengan permasalahan mata kuliah yang ada di perguruan tinggi.

Tabel 2.2 Penelitian Terdahulu (2)

Nama Peneliti	Soria-alcaraz, Jorge A, Özcan Ender, Swan Jerry ,Kendall Graham, Carpio Martin[4]
Tahun Penelitian	2016
Judul Penelitian	*Iterated local search using an add and delete hyper-heuristic for university course timetabling*
Penjelasan Singkat	Pada penelitian ini, menggunakan dataset dari *International Timetabling Competition* (ITC) 2007 track 2 dan track 3. Hasil dari ITC 2007 dibandingkan dengan hasil menggunakan algoritma *iterated local search* (ILS) dengan *improvement* menggunakan *add-delete hyperheuristic*. Penelitian ini menggunakan pendektan ILS yang diperkuat dengan *hyperheuristic* berdasarkan operasi *add-delete*. Didapatkan bahwa mengkombinasikan operasi *add-delete* dengan *iterated local search* pedekakatan *hyperheuristic* lebih effisien.
Hasil Penelitian	Pendekatan dari algoitma yang digunakan pada penelitian diatas dapat dijadikan referensi untuk membuat penelitan yang akan dilakukan. Penggunaan algoritma *iterated local search* untuk memnentukan optimasi yang lebih efisien.

Tabel 2.3 Penelitian Terdahulu (3)

Nama Peneliti	Alaykiran Kemal, Hacibeyoglu Mehmet[5]
Tahun Penelitian	2016

Judul Penelitian	*Using Iterated Local Search to Solve the Course Timetabling Problem at Engineering Faculty of Necmettin Erbakan University*
Penjelasan Singkat	Pada penelitian[5], studi kasus dilakukan pada Fakultas Teknik di Universitas Necmattin Erbankan di Turki. Dimana universitas tersebut masih merupakan universitas yang tergolong baru dan berkembang dengan cepat. Oleh karena itu peniliti ingin memaksimalkan pembuatan jadwal mata kuliah pada tiap semsernya. Disini menggunakan format yang ada pada *International Timetabling Competition* (ITC) 2007. Tujuanya adalah untuk jumlah total ruangan yang dipakai dan memaksimalkan perkuliahan mahasiswa disana. Dengan menggunakan algoritma iterated local search dan hasil dianalisis secara komprehensif. Dengan menunjukkan hasil dapat mengurangi jumlah waktu tiap harinya. Selain itu periode 8 hari dengan kondisi yang telah ditentukan daat terpenuhi dengan algoritma ILS.
Hasil Penelitian	Penilitian diatas dapat menjadi referensi dalam permasalahan *course timetabling* yang akan dikerjakan. Dimana algoritma yang dipakai sama menggunakan ILS, yang mampu memenuhi *soft constrain* dan *hard constrain* yang telah ditentukan.

Tabel 2.4 Penelitian Terdahulu (4)

Nama Peneliti	Jockey Satria Wijaya, Wiwik Angraeni, Ahmad Muklason[6]
Tahun Penelitian	2017
Judul Penelitian	*Advanced Traveler Information System:* Optimasi Rencana Perjalanan Dengan *Orienteering Problem Model* Dan *Iterative Local Search With Hill Climbing* (Studi Kasus: Trayek Angkot Kota Surabaya)
Penjelasan Singkat	Penelitian[6] meneliti tentang kasus kemacetan lalu lintas yang ada di surabaya. Peneliti membuat sebuah *traveller information system* yang digunakan

	turis atau wisatawan yang di surabaya untuk mempermudah mencari rute yang harus dilalui menggunakan angkot. Penggunaan algoritma *iterated local search* digunakan untuk mencari solusi model yang optimal dari model yang telah dibentuk menggunakan orienteering problem. Dimana dar hasil yang didapatkan bahwa algoritma ILS dapat menyelesaikan permasalahan OP dengan cepat dan efisien. Dan didapatkan hasil yang optimal dari permasalahan yang terjadi.
Hasil Penelitian	Dari penelitian diatas dapat digunaka dapat menjadi referensi dalam menggunakan algoritma *iterated local search* untuk penelitian yang dilakukan. Menggunakan tahapan ILS yang telah dijelaskan pada penelitian diatas.

2.2. Penjadwalan Mata Kuliah

Penjadwalan adalah suatu kajian matematis yang bertujuan untuk mencari solusi yang optimal pada penyusunan, pengelompokan, pengurutan atau pemilihan objek diskrit yang biasanya jumlahnya terbatas (*finite number*)[7]. Selain itu, Penjadwalan merupakan suatu kegiatan alokasi sumber daya dengan memiliki kendala (batasan) yang diberikan kepada suatu objek seperti di ruang-waktu, sedemikian rupa untuk memenuhi sedekat mungkin set tujuan yang diinginkan[8].

Permasalahan penjadwalan mata kuliah bisa dilihat dengan menggunakan permodelan matematis. Dalam kasus ini untuk dalam menyelesaikan *hard constrain* yang diberikan dibutuhkan pembuatan LPH *list*, LPA *list* dan MMA *Matrix*[4]. LPH *list* berisikan *timeslot* untuk setiap *event* yang dijadwalkan. LPA *List* berisikan daftar ruang kelas dengan kapasitas yang sesuai yang diperbolehkan untuk setiap acara. Sedangkan MMA *Matrix* adalah matriks simetris yang berisi jumlah mahasiswa yang berkonflik dengan pasangan kejadian tertentu, contohnya jumlah mahasiswa yang terdaftar dalam mata kuliah tertentu (kursus ini tidak boleh ditugaskan pada waktu yang sama untuk mahasiswa manapun).

Tabel 2.5 contoh LPH *list*

Events	*Timeslots*
e_o = Ekologi Industri	(t_1) or (t_3) or (t_5)
.	.
.	.
.	.
e_n	(t20) or (t50)

Tabel 2.6 Contoh LPA *list*

Events	Ruang
e_o = Ekologi Industri	L1 or L2
.	.
.	.
.	.
e_n	A

Tabel 2.7 Contoh MMA *matrix*

	e_1	e_2	e_3	e_4	e_5
e_1	-	2	4	0	1
e_2	2	-	1	1	2
e_3	1	0	-	2	3
e_4	0	0	0	-	1
e_5	1	2	2	5	-

Berikut variabel keputusan yang digunkan dalam penyelesaian permasalahan penjadwalan kuliah[4] :

- Qn : Nama dari *event* / mata kuliah
- en : pelabelan dari *event*
- T : jumlah *timeslot* tersedia
- tn : pelabelan *timeslot*
- A : jumlah ruangan tersedia
- Ln : pelabelan nama ruangan

Selanjutnya mendefinisikan fungsi tujuan ke dalam persamaan matematika. Pada penjadwalan mata kuliah, fungsi tujuan yang dimiliki adalah meminimalkan nilai pelanggaran *dari soft cosntrain* yang dimiliki. Berikut persamaan dari fungsi tujuan penjadwalan mata kuliah otomatis[4] :

$$X_{a,b,c,d} = \begin{cases} 1 & \text{, jika mata kuliah } d \text{ dialokasikan pada hari } a \text{ waktu } b \text{ di ruang } c \\ 0 & \text{, selainnya} \end{cases}$$

Minimize P = $\sum_{a \in D} \sum_{b \in T} \sum_{c \in R} \sum_{d \in M} X_{a,b,c,d} \times N_{ab}$
Dengan :

- P : Nilai pinalti
- D : Himpunan hari perkuliahan
- T : Himpunan jam perkuliahan
- R : Himpunan ruang perkuliahan
- M : Himpunan mata kuliah yang dibuka pada semester tersebut
- N_{ab}: Nilai ketidakpuasan pelaksanaan mata kuliah pada hari dan jam tertentu

Secara khusus, penjadwalan mempunyai definisi sebagai berikut. “ Permasalahan penjadwalan adalah permasalahan yang mempunyai empat parameter, yaitu T (himpunan dari times / alokasi waktu), R (himpunan dari *resources* / sumber daya), M (himpunan dari *meetings* / pertemuan), dan C (himpunan dari *constraints* / batasan). Penjadwalan digunakan untuk mengalokasikan waktu dan sumber daya pada pertemuan tertentu dengan memenuhi batasan paling maksimal.”[9]. Dalam penelitian lainnya, menjelaskan bahwa terdapat dua batasan dalam penyusanan penjadwalan yaitu, *hard constrain*

(harus terpenuhi) dan *soft constrain* (diupayakan untuk terpenuhi)[10]. Solusi dikatakan valid apabila tidak ada *hard constrain* yang dilanggar. Contoh umum dari *hard constrain* dalam penjadwalan mata kuliah sebagai berikut :

- Seorang dosen hanya dapat mengajarkan mata kuliah untuk satu lokasi pada waktu tertentu.
- Seorang mahasiswa hanya dapat mengikuti kuliah untuk satu lokasi pada waktu tertentu.
- Hari aktif perkuliahan adalah hari Senin sampai dengan Jumat
- Mata kuliah dengan bobot tiga SKS dijadwalkan satu kali pertemuan dalam seminggu.
- Dosen menjabat di jadwalkan pada waktu yang tidak bertabrakan dengan kepentingannya.

Sedangkan *soft constrain* adalah ketentuan yang berusaha untuk dipenuhi agar solusi yang dihasilkan semaksimal mungkin. Sehingga tidak ada pihak yang merasa dirugikan. Contoh dari *soft constrain* sendiri adalah sebagai berikut :

- Seorang dosen yang meminta waktu mengajar di pagi hari saja.
- Dalam sehari mahasiswa maksimal dua mata kuliah yang harus diikuti.

Penentuan *constrain* ini dapat bervariasi tergantung dari kebijakan instansi terkait. Solusi yang dapat memenuhi semua *hard constraints* disebut sebagai solusi yang mungkin (*feasible solution*). Apabila solusi tersebut juga dapat memenuhi semua *soft constraints* (tidak ada pelanggaran apapun terhadap *soft constraints*), maka solusi itu disebut sebagai solusi yang sempurna (*perfect solution*)[11]. Akan tetapi, untuk menghasilkan sebuah solusi yang optimal, adanya solusi yang sempurna menjadi tidak begitu penting. Contohnya dalam hal penjadwalan mata kuliah, mungkin saja terjadi pelanggaran

terhadap salah satu yaitu jarak jadwal mata kuliah selanjutnya untuk mahasiswa adalah dua SKS.

Menurut teori kompleksitas komputasi, permasalahan penjadwalan ujian diklasifikasikan sebagai NP-*complete problem*[12]. Dengan demikian, belum ada algoritma eksak yang mampu menemukan solusi optimal dalam waktu polinomial. Oleh karena itulah, permasalahan ini masih menjadi permasalahan yang menarik untuk dikaji. Pada beberapa literatur juga disebutkan beberapa *benchmark dataset* untuk permasalahan penjadwalan ujian ini. *Dataset* yang umum digunakan adalah *Carter dataset* dan *International Timetabling Competition 2007 dataset*

Pada tugas akhir ini, dataset yang akan digunakan adalah data dosen pengajar, data mahasiswa dan data mata kuliah semester genap tahun ajaran 2017 – 2018 di Departemen Teknik Industri, Institut Teknologi Sepuluh Nopember Surabaya. Penjelasan mengenai dataset ini akan dijabarkan pada sub bab 5.2.2..

2.3. Dataset UTS & UAS.

Data yang digunakan pada tugas akhir ini adalah data dosen pengajar, data mahasiswa dan data mata kuliah semester genap tahun ajaran 2017 – 2018 di Departemen Teknik Industri, Institut Teknologi Sepuluh Nopember Surabaya. Data yang didapatkan antara lain : daftar mata kuliah, daftar dosen pengajar, daftar mahasiswa aktif di semester tersebut, dan daftar ruang mengajar. Selain terdapat jadwal dari UPMB sebagai hard constrain dalam pembuatan jadwal mata kuliah. Tabel 2.8 menunjukkan ringkasan dari data yang didapatkan

Tabel 2.8 Ringkasan data

Semester (2017-2018)	Jumlah Mata Kuliah	Jumlah Mahasiswa	Jumlah Dosen	Jumlah Ruang
Gasal	41	698	33	10
Genap	40	601	33	9

Dari data yang didaptkan tadi, dari pihak Departemen Teknik Industri membuat jadwal mata kuliah secara manual dengan data tersebut. Dari jadwal yang telah ada, nantinya akan dibandingkan dengan hasil dari optimasi jadwal mata kuliah otomatis dari tugas akhir ini.

Data yang telah didaptakan akan di lakukan pra-proses terlebih dahulu untuk mendapatkan hasil data yang sesuai dan dapat digunakan untuk proses selanjutnya. Pada Tugas akhir ini menggunakan format dari dataset Socha [13], sehingga data yang didapat akan disesuaikan dengan format yang sesuai dengan dataset referensi.

2.4. Algoritma *Iterrated Local Search*

Algoritma *Iterated Local Search* merupakan merupakan teknik *metaheuristic* dan *global optimization*. Algoritma ini adalah perpanjangan dari *Multi-Restart Search* dan dipertimbangkan sebagai *neighborhood* dari banyaknya pencarian dua fase seperti G*reedy Randomized Adaptive Search Procedure* (GRASP) dan *Variable Neighborhood Search* (VNS)[14].

Tujuan dari *Iterative Local Search* adalah untuk meningkatkan pencarian *stokastik* dengan melakukan sampling yang lebih luas di lingkungan kandidat solusi dan menggunakan teknik pencarian lokal untuk menyaring solusi menjadi optimal[13].

Pseudo code umum untuk algoritma *iterated local search* ditunjukkan pada kode 5.1[15]. Dapat dilihat bahwa iterated local search akan terus melakukan perubahan data hingga tidak ada lagi solusi lebih yang dapat ditemukan dan jumlah iterasi telah mencapai batas maksimal. Dari kode 5.1 ditunjukkan bahwa *iterated local search* memiliki 4 komponen umum[15], yaitu :

- Pembentukan Solusi Awal
- Perturbation

- Local Search
- Kriterian Penerimaan Solusi

```
Procedure Iterative Local Search
s₀ = Generate Initial Solution
s* = LocalSearch(s₀)
      Repeat
      s' = Perturbation (s*, history)
      s*' = LocalSearch (s')
      s* = AcceptanceCriterion (s*, s*',
      history)
      until termination condition met
end
```

Kode 2.1 Pseducode *iterated local search*

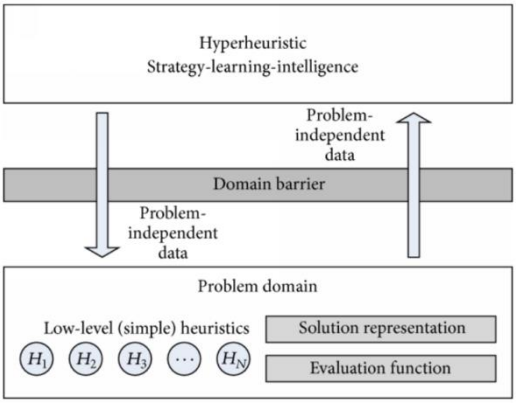
Dalam pemakaiannya, tingkat kesulitan dari *iterative ocal search* terdapat pada penyimpanan sejarah atau *history*. Jika tidak memiliki hal tersebut, maka pemakaian *iterative local search* tidak memiliki memori atau ingatan. Secara umum, *iterative local search* tanpa ingatan ini yang sering digunakan, meskipun penelitian membuktikan bahwa menggunakan ingatan atau memori dapat meningkatkan hasil pencarian[16].
Pada penelitian kali ini penulis menggnukan dua buah metode *iterated local search*, yaitu *move* dan *swap*. *Move* berfungsi untuk menukar *event* dengan timeslot terpilih. Sedangkan *swap* berfungsi untuk menukar antar *event* yang terpilih. Sehingga dapat merubah hasil dari pinalti dengan algoritma *iterated local search.*

2.5. Algoritma *Hyper Heruristic*

Hyper-heuristic adalah (*meta-*)*heuristic* yang beroperasi pada tingkat yang lebih tinggi untuk memilih atau menghasilkan serangkaian *low level (meta-)heuristic* dalam upaya memecahkan masalah pengoptimalan yang sulit[4]. Tujuan dari hyper-heuristic adalah untuk menghasilkan heuristic yang

bersifat generik sehingga dapat digunakan pada set permasalahan yang berbeda.

Hyper-heuristic didefinisikan sebagai kumpulan pendekatan yang bertujuan untuk mengotomasi proses. Otomasi yang dilakukan tersebut biasanya dilakukan melalui metode *machine learning.* Tujuannya adalah memilih dan mengombinasikan *heuristic* yang sederhana atau menghasilkan *heuristic* baru dengan komponen *heuristic* yang sudah ada, sehingga dapat menyelesaikan permasalahan pencarian komputasi yang sangat sulit dilakukan secara manual.



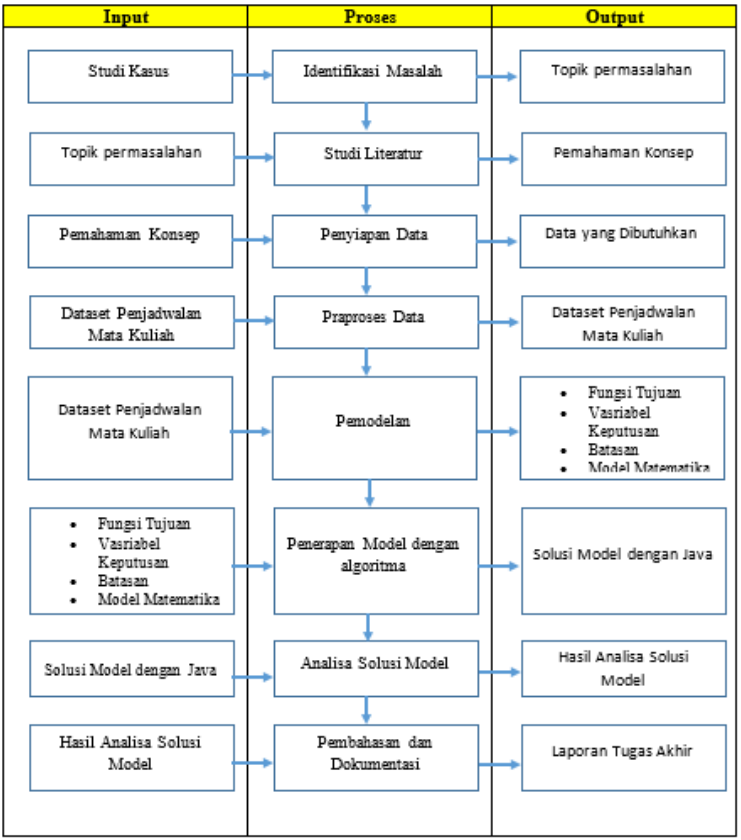
Gambar 2.1 *Hyper-heuristic framework*

Pada gambar 2.1 menunjukkan *framework* dari *hyper-heuristic*[17]. *Hyper-heuristic* melakukan pencarian di atas *solution space* dari *low-level heuristic*, tidak seperti *heuristic* sederhana atau *meta-heuristic*, yang melakukan pencarian secara langsung di atas *solution space*. *Domain barier* adalah komponen kunci dari *frameworks*, yang mencegah pengetahuan masalah lewat dari *low-level* ke *high-level*. Sehingga *hyperheuristic* hanya tahu tentang adanya *low-level heuristic* secara keseluruhan. Hal ini dapat membuat hyperheuristic

mampu untuk membuat keputusan dan menyimpan informasi yang berguna untuk mengoptimalkan *problem solution.*

BAB III
METODOLOGI

Pada bab ini akan dijelaskan mengenai alur metodologi yang akan dilakukan dalam tugas akhir ini. Metodologi ini juga digunakan sebagai pedoman untuk melaksanakan tugas akhir agar terarah dan sistematis. Gambar 3.1 menunjukkan metodologi tugas akhir yang digunakan.



Gambar 3.1 Metodologi Tugas Akhir

3.1. Identifikasi Masalah

Tahap identifikasi masalah adalah tahap untuk memahami permasalahan penjadwalan mata kuliah yang terjadi di Departemen Teknik Industri, Institut Teknologi Sepuluh Nopember Surabaya. Penjadwalan masih dialkukan secara manual sehingga menyebabkan banyaknya revisi dalam pembuatan jadwal mata kuliah. Pada tahap ini juga, menetapkan tujuan, dan menentukan batasan dari permasalahan yang akan diselesaikan melalui tugas akhir yang dikerjakan. Penetapan tujuan berfungsi untuk mengetahui hasil yang akan diporeleh dari tugas akhir ini, sedangkan batasan masalah digunakan sebagai batas dalam pengerjaan tugas akhir.

3.2. Studi Literatur

Pada tahap studi literatur, mengumpulkan penelitian-penelitian yang telah dilakukan sebelumnya, atau buku dan dokumen yang terkait dengan tugas akhir yang dikerjakan. Pada tugas akhir ini, mengusulkan topik optimasi penjadwalan mata kuliah otomatis. Sehingga pada tahap ini mencari algoritma yang sesuai dengan permasalahan yang ada. Sehingga solusi yang dihasilkan dapat maksimal.

Tugas akhir ini, menggunakan algoritma *iterated local search* yang diperkuat menggunakan *hyper heuristic* yang sesuai dengan permasalahan pada *course timetabling*. Hal ini mengacu pada referensi utama yang digunakan adalah *iterated local search using an add and delete hyper-heuristic for university course timetabling*. Literatur lain yang digunakan dalam tugas akhir ini yaitu literatur mengenai permasalahan penjadwalan mata kuliah, *algoritma iterated local search* dan *hyper heuristic*.

3.3. Penyiapan Data

Pada tahap penyiapan data, dilakukan pengumpulan data-data awal yang nantinya akan digunakan dalam pengerjaan tugas akhir. Pengumpulan data ini dilakukan dnegan wawancara yang

terlebih dahulu telah menyiapkan *interview protocol* untuk penggalian data. Data yang diperoleh dari hasil wawancara adalah data mata kuliah pada tiap semester, data dosen pengajar, data mahasiswa dan data ruangan yang digunakan untuk perkuliahan di Departemen Teknik Industri, Institut Teknologi Sepuluh Nopember Surabaya. Data yang didapat merupakan data pada semester ganjil dan genap tahun ajarana 2017- 2018. Batasan yang dipakai adalah dalam pengambilan data hanya menggunakan data S1.

3.4. Praposes Data

Pada tahap praproses data, memahami data yang telah diperoleh pada tahapan sebelumnya. Pemahaman dilakukan untuk tipe datanya, variabel yang ada pada data, dan jumlah data itu sendiri. Setelah itu, dilakukan praproses data untuk menghasilkan data set yang akan dibuat model. Data yang digunakan antara lain : data mata kuliah, data dosen pengajar, data mahasiswa dan data ruangan. Pada Departemen Teknik Industri, Institut Teknologi Sepuluh Nopember Surabaya, untuk alokasi waktu yang diberikan adalah menggunakan waktu SKS yaitu 50 menit ditambah 5 menit untuk waktu peralihan mata kuliah. Jadwal dimulai dari jam 7.30 sampai dengan 16.10 WIB sehingga pada setiap hari memiliki 10 *timeslot*. Tahapan praproses data menghasilkan dataset yang dapat diselesaikan dengan algoritma iterated local search. Untuk performatan dataset menggunakan format Socha[13].

3.5. Pemodelan

Pada tahap pemodelan, permasalahan dianalisis untuk menentukan fungsi tujuan dan batasan masalah. Fungsi tujuan dan batasan masalah ditentukan dengan menerjemahkan permasalahan ke dalam model matematika. Yang dapat diselesaikan dengan menginisiasi, sebagai berikut :

a. Variabel Keputusan

Membuat solusi optimal pembuatan jadwal mata kuliah otomatis. Yang telah dijelaskan pada sub bab 5.2.1 Penjadwalan Mata Kuliah.

b. Fungsi Tujuan

Meminimalkan nilai pelanggaran terhadap *soft constraint*.

c. Batasan

Memenuhi semua *hard constrain* yang diberikan. Dan berusaha memenuhi *soft constrain* yang ada

3.6. Penerapan Model dengan Algoritma

Pada tahap ini, model yang telah disiapkan diimplementasikan untuk algoritma *iterated local search* dalam bahasa *java*. Dimana hasil tersebut yang nantinya menjadi solusi dari permasalahan pada tugas akhir ini. Dalam penerapannya, metode *iterated local search* yang pertam kali adalah menentukan solusi awal yang akan digunakan sebagai awal untuk mencari iterasi berikutnya sehingga menghasilkan solusi yang lebih baik. Setelah itu dalam *iterated local search*, melakukan perbaikan dari solusi awal menggunakan *local search*. Pada *loacal search*, dicari suatu pekerjaan atau mata kuliah pada posisi *i* dipindahan dan dimasukkan kembali pada semua kemungkinan posisi, $j \neq i$. Sehingga, jka terdapat peningkatan kualitas solusi, maka solusi di *update* dan proses diatas terulang kembali. *Local Search* dilakukan hingga tidak ada lagi solusi yang lebih optimal. Setelah itu melakukan *perturbation*, dengan cara memodifikasi solusi yang terakhir didapat dengan cara melakukan satu kali perpindahan. Dapat dilakukan dengan *random* atau berurutan dalam perpindahan mata kuliah/pekerjaan. Perubahan kecil yang terjadi digunakan untuk menemukan solusi optimal yang lebih baik daripada sebelumnya. Setelah ditemukan solusi yang baru akan diperbaiki lagi dengan menggunakan *local search*. *Initial feasible solution* adalah hal terakhir yang harus dibuat, hal ini untuk menghindari penuruna kualitas solusi yang dihasilkan.

Lalu hasil tersebut akan diperkuat dengan penggunaan algoritma *hyper heuristic*. Sehingga solusi akhir yang diperoleh hasil iterasi merupakan hasil paling optimal yang didapatkan. Hasil akhir dari tahapan ini adalah aplikasi berbasis java menerapkan algoritma *iterated local search* yang telah di tingkatkan dengan *hyper heuristic* untuk penjadwalan mata kuliah otomatis.

3.7. Analisi Solusi Model

Pada tahap analisa solusi model, dilakukan perbandingan dari hasil yang didapatkan dengan penerapan algoritma *iterated local search* dalam pembuatan jadwal mata kuliah otomatis, dibandingkan dengan penjadwalan mata kuliah secara manual. Hasil perbandingan ini didapatkan dari nilai pelanggaran terhadap soft constrain yang paling minimal. Sehingga dapat diketahui penjadwalan mana yang lebih baik dan optimal

3.8. Pembahasan dan Dokumentasi

Pada tahap ini, melakukan pendokumentasian terhadapa hal-hal yang telah dilakukan selama pengerjaan tugas akhir ini. Serta mendokumentasikan hasil analisis yang telah dilakukan. Output pada tahap ini adalah laporan tugas akhir. Dimana laporan tugas akhir ini nantinya sebagai bukti dari seluruh pengerjaan penilitian yang dilakukan.

Halaman ini sengaja dikosongkan.

BAB IV
PERANCANGAN

Pada bab ini akan dijelaskan bagaimana rancangan dari penelitian tugas akhir yang meliputi subjek dan objek dari tugas akhir, pemilihan subjek dan objek tugas akhir serta bagaimana tugas akhir akan dilakukan.

4.1. Pengumpulan dan Deskripsi Data

Pada tugas akhir ini, data yang digunakan adalah data yang didapatkan dari Departemen Teknik Industri meliputi jadwal kuliah pada semester gasal dan semester genap tahun ajaran 2017/2018. Data yang digunakan meliputi data frs yang telah dilakukan oleh mahasiswa Teknik Industri pada semester tersebut, data ruang belajar yang digunakan dan data dosen yang mengajar. Selain itu juga menggunakan data jadwal yang telah dibuat oleh Departemen Teknik Industri yang nantinya kan dijadikan perbandingan dengan hasil TA yang akan dilakukan.

4.1.1. Deskripsi Data Mahasiswa

Data mahasiswa yang ditampilkan dalam bentuk format Microsoft Excel yang didalamnya berisi beberapa data sebagai berikut : NRP, Mata Kuliah yang diambil, jumlah SKS mata kuliah, dan kelas mata kuliah. Data tersebut akan ditunjukkan pada tabel 4.1.

Tabel 4.1 Sebagian data mahasiswa

No	NRP	Mata Kuliah	SKS	Kelas
1	2510100082	Analisa Keputusan	3	A
2	2513100146	Analisa Keputusan	3	A
3	2513100178	Analisa Keputusan	3	A
4	2513100701	Analisa Keputusan	3	A
5	2514100002	Analisa Keputusan	3	A

No	NRP	Mata Kuliah	SKS	Kelas
6	2514100003	Analisa Keputusan	3	A
7	2514100008	Analisa Keputusan	3	A
8	2514100011	Analisa Keputusan	3	A
9	2514100022	Analisa Keputusan	3	A
10	2514100025	Analisa Keputusan	3	A
11	2514100033	Analisa Keputusan	3	A
12	2516100002	Analisis dan Estimasi Biaya	3	A
13	2516100008	Analisis dan Estimasi Biaya	3	A
14	2516100019	Analisis dan Estimasi Biaya	3	A
15	2516100029	Analisis dan Estimasi Biaya	3	A
16	2516100032	Analisis dan Estimasi Biaya	3	A
17	2516100039	Analisis dan Estimasi Biaya	3	A
18	2516100041	Analisis dan Estimasi Biaya	3	A
19	2516100043	Analisis dan Estimasi Biaya	3	A
20	2516100046	Analisis dan Estimasi Biaya	3	A
21	2516100052	Analisis dan Estimasi Biaya	3	A

Terdapat dua data mahasiswa, yaitu data mahasiswa pada semester gasal dan data mahasiswa pada semester genap. Kedua data tersebut mempunyai format yang sama seperti yang ditampilkan pada tabel 4.1. Diringkas yang ditunjukkan pada tabel 4.2

Tabel 4.2 Ringakasan data mahasiswa

No	Semester	Jumlah Mahasiswa	Jumlah *Event*
1	Gasal	687	111
2	Genap	601	93

4.1.2. Deskripsi Data Mata Kuliah

Data selanjutnya adalah data mata kuliah yang di jadwalkan pada setiap semesternya. Pada data tersebut ditulis dalam format Microsoft Excel yang mempunyai kolom yang berisi data sebagai berikut : mata kuliah, kelas, sks, ruang. Yang ditunjukkan pada tabel 4.3.

Tabel 4.3 sebagian data mata kuliah

No	**Mata Kuliah**	**Kelas**	**SKS**	**Ruang**
1	Analisa Keputusan	A	3	IE-108
2	Analisa Produktivitas	A	3	IE-106
3	Analisis dan Estimasi Biaya	A	3	IE-105
4	Analisis dan Estimasi Biaya	B	3	IE-109
5	Analisis dan Estimasi Biaya	C	3	Lab. MM
6	Analisis dan Estimasi Biaya	D	3	IE-106
7	Analisis dan Estimasi Biaya	Q	3	IE-108
8	Aplikasi Ergonomi Industri	A	3	IE-105
9	Computer Integrated Manufacturing	A	3	IE-303D
10	Data Mining	A	3	IE-106
11	Ergo Safety	A	3	IE-105
12	Ergonomi Industri	A	3	IE-103
13	Ergonomi Industri	C	3	IE-102
14	Ergonomi Industri	D	3	IE-103
15	Ergonomi Industri	Q	3	IE-104
16	Faal dan Biomenikanika Kerja	A	3	IE-105
17	Makro Ergonomi	A	3	IE-102
18	Manajemen Distribusi	A	3	Lab. MM

19	Manajemen Lingkungan Industri	A	3	IE-108
20	Ergonomi Industri	C	3	IE-102

Selain itu juga terdapat data *timeslot* yang digunakan untuk mendeskripsika waktu yang diadwalkan pada mata kuliah tersebut. Data tersebut dapat dilihat pada tabel 4.4.

Tabel 4.4 data *timeslot*

Senin	**Selasa**	**Rabu**	**Kamis**	**Jumat**
07.30-08.20	07.30-08.20	07.30-08.20	07.30-08.20	07.30-08.20
08.25-09.15	08.25-09.15	08.25-09.15	08.25-09.15	08.25-09.15
09.20-10.10	09.20-10.10	09.20-10.10	09.20-10.10	09.20-10.10
10.15-11.05	10.15-11.05	10.15-11.05	10.15-11.05	10.15-11.05
11.10-12.00	11.10-12.00	11.10-12.00	11.10-12.00	13.00-13.50
12.05-12.55	12.05-12.55	12.05-12.55	12.05-12.55	13.55-14.45
13.30-14.20	13.30-14.20	13.30-14.20	13.30-14.20	14.50-15.40
14.25-15.15	14.25-15.15	14.25-15.15	14.25-15.15	15.45-16.35
15.20-16.10	15.20-16.10	15.20-16.10	15.20-16.10	16.40-17.00
16.15-17.05	16.15-17.05	16.15-17.05	16.15-17.05	
17.10-18.00	17.10-18.00	17.10-18.00	17.10-18.00	

Data *timeslot* yang digunakan pada semester gasal dan genap adalah sama. Sehingga pada prosesnya nanti data *timeslot* yang digunakan hanya satu dan disesuaikan dengan jadwal masing-masing mata kuliah.

Ringksan dari tabel matakuliah ditunjukkan pada tabel 4.5

Tabel 4.5 Ringkasan data mata kuliah

No	Semester	Jumlah *Event*	Jumlah Ruang	Jumlah *Timeslot*
1	Gasal	111	10	53
2	Genap	93	9	53

4.2. Pra Proses Data

Pada tahap pra proses data, dilakukan pembuatan data yang akan digunakan sebagai *file* masukan untuk aplikasi penjadwalan mata kuliah otomatis. Data tersebut disesuaikan dengan format *Socha*. Format *Socha* berupa file dengan ekstensi tim. Selain itu pada praproses data juga membuat file dengan ekstensi sol yang berisi data dari jadwal manual yang telah dibuat.

4.2.1. Data TIM

Data dengan format .tim merupakan data yang berisikan tiga matriks bilangan *biner* yang membandingkan antara mahasiswa dengan *event*, ruang dengan *features*, dan *event* dengan *features*. *Features* disini dinyatakan secara umum pada setiap *event* yang dilakukan *features* yang harus ada adalah LCD dan AC. Sehingga pada pembuatan file .tim dibutuhkan data pada 4.1.1 yang meliputi *event* yang diambil mahasiswa, *event* yang berjalan pada semester tersebut, dan *features* serta jumlah SKS pada masing-masing *event*. Pada bagian ini dilakukan pemetaan kode dari *event* yang berjalan sehingga dapat memudahkan dalam pembacaan file nanti. Dapat dilihat pada tabel 4.6

Tabel 4.6 Sebagian data kode *event*

No	**Mata Kuliah**	**Kelas**	**Kode *Event***
1	Analisa Keputusan	A	001
2	Analisa Produktivitas	A	002
3	Analisis dan Estimasi Biaya	A	003
4	Analisis dan Estimasi Biaya	B	004
5	Analisis dan Estimasi Biaya	C	005

No	Mata Kuliah	Kelas	Kode *Event*
6	Analisis dan Estimasi Biaya	D	006
7	Analisis dan Estimasi Biaya	Q	007
8	Aplikasi Ergonomi Industri	A	008
9	Computer Integrated Manufacturing	A	009
10	Data Mining	A	010
11	Ergo Safety	A	011
12	Ergonomi Industri	A	012
13	Ergonomi Industri	B	013
14	Ergonomi Industri	C	014
15	Ergonomi Industri	D	015
16	Ergonomi Industri	Q	016
17	Faal dan Biomenikanika Kerja	A	017
18	Makro Ergonomi	A	018
19	Manajemen Distribusi	A	019
20	Manajemen Lingkungan Industri	A	020

Data mentah yang sebelumnya disimpan dalam format Microsoft Excel tersebut diolah dengan bantuan *Pivot Table*. Dengan menggunakan *Pivot Table*, dapat diketahui mata kuliah apa saja yang diujikan dan jumlah mahasiswa yang mengambil mata kuliah tersebut. Selanjutnya dibuat dalam format bilangan *biner*, matriks biner *event* mahasiswa, matriks *biner* ruang *features* dan matriks *biner event features*. Setelah itu menggunakan bantuan *add-in* Excel yaitu KuTools untuk merubah bentuk *biner* menjadi vertikal ke bawah sesuai format dari socha yang berekstensi .tim. Baris pertama pada file tersebut berisi jumlah *event*, jumlah ruang, jumlah *features*, jumlah mahasiswa. Baris selanjutnya adalah kapasitas ruang yang dimiliki masing-masing ruang. Selanjutnya matriks biner *event* dengan mahasiswa, lalu matriks *biner* ruang dengan *features*, diteruskan matriks *biner event* dan *features* dan yang terakhir jumlah sks masing-masing *event* yang disimpan dalam txt file yang nantinya dirubah menjadi ekstensi .tim. Tampilan

final data yang sudah berbentuk format .tim dapat dilihat pada tabel 4.7.

Tabel 4.7 Sebagian data .tim

Baris ke-	**Gasal.tim**	**Genap.tim**
1	111 10 2 687	93 9 2 601
2	45	45
3	45	45
4	45	45
5	45	45
6	45	45
7	45	45
8	45	45
9	70	70
10	45	45
11	45	0
12	1	0
13	0	0
14	0	0
15	0	0
16	0	0
17	0	0
18	0	0
19	0	0
20	0	0
21	0	0
22	0	1
23	0	0
24	0	0
25	0	0
26	0	1
27	0	0
28	0	0
29	0	0
30	0	0

4.2.2. Data SOL

Data dengan ekstensi .sol merupakan data dari jadwal manual yang telah dibuat oleh Departemen Teknik Industri yang di kodekan berdasarkan format dari .sol. Data tersebut berisi jumlah *event*, jumlah *timeslot*, ruang, dan *timeslot*. Yang dilakukan pertam kali adalah mengkodekan ruang yang digunakan. Dapat dilihat pada tabel 4.8.

Tabel 4.8 Data kode ruang

No	Ruang	Kode Ruang
1	IE-101	1
2	IE-102	2
3	IE-103	3
4	IE-104	4
5	IE-105	5
6	IE-106	6
7	IE-108	7
8	IE-109	8
9	Lab.MM	9
10	IE-303D	10

Kode ruang digunakan sebagai acuan dalam pembuatan file input SOL. Selanjutnya dilakukan pengkodean terhadap *timeslot* yang dimiliki. Yang dikodekan berurutan dengan hari untuk memudahkan dalam pembacaan nantinya. Rentang kode disesuaikan dengan jumlah timeslot yang dimiliki yaitu anatar 1 sampai dengan 53. Hal ini berlaku untuk semester gasal dan genap. Pengkodean ini memudahkan program nantinya untuk membantu menhitung nilai pinalti yang didapat dari jadwal manual yang telah dibuat. Pengkodean timeslot dapat dilihat pada tabel 4.9.

Tabel 4.9 Data kode *timeslot*

Senin	**Kode**	**Selasa**	**Kode**	**Rabu**	**Kode**	**Kamis**	**Kode**	**Jumat**	**Kode**
07.30-08.20	1	07.30-08.20	12	07.30-08.20	23	07.30-08.20	34	07.30-08.20	45
08.25-09.15	2	08.25-09.15	13	08.25-09.15	24	08.25-09.15	35	08.25-09.15	46
09.20-10.10	3	09.20-10.10	14	09.20-10.10	25	09.20-10.10	36	09.20-10.10	47
10.15-11.05	4	10.15-11.05	15	10.15-11.05	26	10.15-11.05	37	10.15-11.05	48
11.10-12.00	5	11.10-12.00	16	11.10-12.00	27	11.10-12.00	38	13.00-13.50	49
12.05-12.55	6	12.05-12.55	17	12.05-12.55	28	12.05-12.55	39	13.55-14.45	50
13.30-14.20	7	13.30-14.20	18	13.30-14.20	29	13.30-14.20	40	14.50-15.40	51
14.25-15.15	8	14.25-15.15	19	14.25-15.15	30	14.25-15.15	41	15.45-16.35	52
15.20-16.10	9	15.20-16.10	20	15.20-16.10	31	15.20-16.10	42	16.40-17.00	53
16.15-17.05	10	16.15-17.05	21	16.15-17.05	32	16.15-17.05	43		
17.10-18.00	11	17.10-18.00	22	17.10-18.00	33	17.10-18.00	44		

Data yang telah dikodekan dibuat menjadi satu dan disimpan dalm bentuk csv yang kemudian nantinya dirubah dalam format .sol. Tampilan final dari data format.sol dapat dilihat pada tabel 4.10.

Tabel 4.10 Sebagain data format .sol

Baris ke-	**Gasal.sol**	**Genap.sol**
1	111 53	93 53
2	7 7 8 9	2 15 16 17
3	6 4 5 6	5 7 8 9
4	5 37 38 39	4 1 2 3
5	8 34 35 36	6 18 19 20
6	9 34 35 36	8 10 11
7	6 37 38 39	3 5 6
8	7 37 38 39	1 5 6
9	5 26 27 28	2 5 6
10	10 29 30 31	8 21 22
11	6 40 41 42	5 37 38 39
12	5 40 41 42	8 18 19 20
13	3 29 30 31	7 29 30 31
14	1 12 13 14	7 26 27 28
15	2 12 13 14	8 29 30 31
16	3 12 13 14	4 7 8 9
17	4 29 30 31	5 18 19 20
18	5 12 13 14	3 7 8 9
19	2 29 30 31	1 15 16 17
20	9 1 2 3	6 1 2 3
21	7 18 19 20	7 12 13 14
22	8 40 41 42	3 15 16 17
23	5 1 2 3	4 18 19 20
24	1 29 30 31	1 37 38 39
25	1 49 50 51	2 37 38 39
26	2 49 50 51	3 37 38 39
27	3 49 50 51	4 37 38 39
28	4 49 50 51	1 40 41 42
29	5 49 50 51	2 40 41 42
30	7 42 43	1 34 35 36

4.3. Formulasi Fungsi Tujuan

Pada tahap formulasi fungsi tujuan adalah mendefinisikan *hard constrain* dan *soft constrain* yang perlu dipenuhi saat penjadwalan mata kuliah, menjadi model matematis. Hal ini bertujuan untuk memudahkan dalam pemahanan konsep dari fungsi tujuan yang dinginkan.

4.3.1. Variabel Keputusan

Variabel keputusan adalah variabel yang harus memenuhi semua *hard constraint* yang telah ditentukan sebelumnya. Variabel keputusan ini digunakan untuk memastikan bahwa setiap *event* sudah dijadwalkan. Persamaan matematisnya ditulis pada persamaan (1).

$$X_{itr}\begin{cases}1, & jika\ event\ ke-i\ dijadwalkan \\ & pada\ timeslot\ ke-t\ dan\ ruang\ ke-r \\ 0, & jika\ tidak\end{cases} \quad (1)$$

Dengan :
X adalah variabel keputusan
$i = \{1,2,3,\dots,e\}$ adalah urutan *event*
$t = \{1,2,3,\dots,m\}$ adalah urutan *timeslot* pada jadwal
$r = \{1,2,3,\dots,n\}$ adalah urutan dari ruang mengajar

4.3.2. Model Matematis *Hard Constraints*

Hard constraint yang digunakan dalam tugas akhir ini ada empat. Pertama, semua *event* harus dijadwalkan. Kedua, tidak boleh ada jadwal yang bentrok antar *event* yang diambil mahasiswa yang sama. Ketiga, total mahasiswa yang mengikuti perkuliahan tidak melebihi kapasitas total ruang yang tersedia dan yang ke empat ruang dan *timeslot* tidak dijadwalkan bersamaan.

Untuk memastikan bahwa jadwal yang dibuat telah memenuhi *hard constraint* dimana setiap *event* harus dijadwalkan, maka dinyatakan dengan persamaan matematis (2).

$$\sum_{i=1}^{e}\sum_{t=1}^{m}\sum_{r=1}^{n} X_{itr} = 1 \quad (2)$$

Dimana X_{it} adalah *event* ke-i yang dijadwalkan pada slot mtakuliah ke-t. Selanjutnya adalah memastikan bahwa tidak ada jadwal yang bentrok antar *event* yang diambil mahasiswa yang sama (3).

$$\sum_{i=1}^{e-1} \sum_{j=i+1}^{e} C_{ij}.V_{ij} = 0$$

$$V_{ij} \begin{cases} 1 \; jika \; t_i = t_j \\ 0 \end{cases} \quad (3)$$

Dengan :

C_{ij} = jumlah mahasiswa peserta *event* i dan j

V_{ij} = *vector* yang menunjukkan apakah mata kuliah i dan mata kuliah j bentrok

t_i = *timeslot* dimana *event* ke i dijadwalkan

Untuk memastikan bahwa jadwal yang dibuat telah memenuhi *hard constraint* ketiga dimana total mahasiswa setiap *event* tidak melebihi kapasitas ruang mengajar pada *event* ke-j maka dinyatakan dengan persamaan matematis (4).

$$\sum_{i=1}^{e} S_i X_{itr} \leq \sum_{r=1}^{n} P_r \quad (4)$$

Dengan :

$i = \{1,2,3, \dots, n\}$ adalah urutan mata kuliah

S_i = jumlah mahasiswa untuk event ke-i

P_r = jumlah kapasitas ruang untuk ruang ke-r

Yang kelima adalah memastikan bahwa *timeslot* dan ruang tidak dijadwalkan pada waktu bersamaan. Sehingga pada setiap timeslot yang dimiliki maka hanya dijadwalkan pada satu ruang. Hal ini sama dengan yang ada pada persamaan (1) dimana event ke i diajdwalkan pada timeslot t pada ruang r maka nilai nya adalah 1

4.3.3. Model Matematis *Soft Constraints*

Pada tugas akhir yang dibuat kali ini, *soft constrain* yang digunakan ada tiga, semua soft constrain dibuat untuk mengoptimasi penjadwalan mata kuliah. Untuk memastikan bahwa jadwal optimasi lebih baik, adapun soft contrain yang digunakan sebagai berikut :

1. Membatsasi mahasiswa mengambil lebih dari dua event dalam satu hari.

$$\sum_{s=1}^{o}\sum_{h=1}^{m} Y_{sh} \leq 2$$

$s = \{1,2,3, \dots, o\}$ adalah urutan mahasiswa
$h = \{1,2,3, \dots, m\}$ adalah urutan hari
Y_{sh} adalah nilai mahasiwa ke s pada hari ke h

2. Membatasi mahasiswa mengambil satu event dalam satu hari.

   Dalam pemodelan matematis, soft contrain pertama dan kedua dapat dinyatakan dalam persamaan matematis.

$$\sum_{s=1}^{o}\sum_{h=1}^{m} Y_{sh} > 1$$

   $s = \{1,2,3, \dots, o\}$ adalah urutan mahasiswa
   $h = \{1,2,3, \dots, m\}$ adalah urutan hari
   Y_{sh} adalah nilai mahasiwa ke s pada hari ke h

3. Membatasi mahasiswa mengambil event yang ada pada timeslot terakhir. Dinyatakan dalam persamaan matematis.
   $A_{si} = 0$, Untuk t mod 11 = 0

   $s = \{1,2,3, \dots, o\}$ adalah urutan mahasiswa
   $i = \{1,2,3, \dots, m\}$ adalah urutan event

A_{si} adalah nilai mahasiswa ke s pada event ke i

4.3.4. Fungsi Tujuan

Fungsi tujuan yang digunakan untuk memastikan bahwa jadwal yang dihasilkan telah optimal atau tidak adalah dengan cara meminimalkan nilai pinalti (P) yang dinyatakan dalam persamaan matematis (5).

$$Minimize\ P = P_1 + P_2 + P_3 \quad (5)$$

Dengan :

P_1 = nilai pinalti mahasiswa mengambil lebih dari dua mata kuliah dalam satu hari

P_2 = nilai pinalti mahasiswa mengambil hanya satu mata kuliah dalam satuu hari

P_3 = nilai pinalti mahasiswa mengambil mata kuliah pada jam terakhir

Pinalti yang digunakan dalam tugas akhir ini berjumlah tiga. Yang pertama di menghitung jumlah mahasiswa yang dijadwalkan sehari tiga kali atau lebih dinyatakan sebagai P_1. Persamaan matemastis dari P_1 dinyatakan pada persamaan (6).

$$P_1 = \sum_{s=1}^{o} \sum_{h=1}^{m} Y_{sh} \quad (6)$$

Dengan :

$s = \{1,2,3,\dots,o\}$ adalah urutan mahasiswa

$h = \{1,2,3,\dots,m\}$ adalah urutan hari

Y_{sh} adalah nilai mahasiswa ke s pada hari ke h dalam variabel biner yang didefisinikan pada persamaan sebagai berikut :

$$Y_{sh} \begin{cases} 1, & jika\ pada\ hari\ h\ mahasiswa\ s\ dijadwalkan\ lebih\ dari\ dua\ event \\ 0, & jika\ tidak \end{cases}$$

P_2 adalah menghitung nilai dari mahasiswa dalam sehari mengambil *event* hanya 1. Persamaan matemastis dari P_2 dinyatakan pada persamaan (7).

$$P_2 = \sum_{s=1}^{o} \sum_{h=1}^{m} Y_{sh} \qquad (7)$$

Dengan :
$s = \{1,2,3, \dots, o\}$ adalah urutan mahasiswa
$t = \{1,2,3, \dots, m\}$ adalah urutan hari
Y_{sh} adalah nilai mahasiswa ke s pada hari ke h dalam variabel biner yang didefisinikan pada persamaan sebagai berikut :

$$Y_{sh} \begin{cases} 1, & jika\ pada\ hari\ h\ mahasiswa\ s\ dijadwalkan\ hanya\ satu\ event \\ 0, & jika\ tidak \end{cases}$$

Dan yang ketiga menghitung nilai dari mahasiswa mengambil jadwal *event* terakhir di setiap harinya dinayatakan sebagai P_3. P_3 dimodelkan dengan persamaa matematis (8).

$$P_3 = \sum_{s=1}^{o} \sum_{i=1}^{m} A_{si} \qquad (8)$$

Dengan :
$s = \{1,2,3, \dots, o\}$ adalah urutan mahasiswa
$i = \{1,2,3, \dots, m\}$ adalah urutan timeslot
A_{si} adalah nilai mahasiwa ke s pada event ke i dalam variabel biner yang didefisinikan pada persamaan sebagai berikut:

$$A_{si} \begin{cases} 1, & jika\ pada\ event\ i\ mahasiswa\ dijadwlakan \\ & pada\ t\ terakhir\ di\ hari\ tersebut \\ 0, & jika\ tidak \end{cases}$$

4.3.5. Pembentukan *Conflict Matrix*

Conflix matrix adalah matriks yang berisi nilai dari banyaknya mahasiswa yang mengambil dua *event* secara bersamaan. *Conflix matrix* dibuat dari file input .sol yang telah dibuat sebelumnya. Untuk mengetahui nilai C_{ij}, data mahasiswa diubah kedalam bentuk matriks dua dimensi dengan jumlah kolom dan baris sejumlah *event* yang dilaksanakan. Setiap selnya berisi jumlah mahasiswa yang mengikuti *event* C_{ij}, sehingga dapat memastikan tidak ada jadwal yang bertabrakan oleh mahasiswa. *Conflix matrix* dapat dilihat pada tabel 4.11.

Tabel 4.11 Sebagian conflix matrix

	001	002	003	004	005	006	007	008	009
001	0	3	0	0	0	0	0	1	1
002	3	0	0	0	0	0	0	3	0
003	0	0	0	0	0	1	0	0	0
004	0	0	0	0	0	0	0	2	0
005	0	0	0	0	0	0	0	0	0
006	0	0	1	0	0	0	0	1	0
007	0	0	0	0	0	0	0	0	0
008	1	3	0	2	0	1	0	0	0
009	1	0	0	0	0	0	0	0	0

4.4. Pembentukan *Initial Solution*

Initial solution merupakan jadwal baru yang dibuat secara otomatis dengan memperhatikan *conflix matrix*. Dalam pembuatan initial solution menggunakan algritma greedy. Jadwal dibuat berdasarkan urutan *event* yang paling tinggi nilai di dalam *conflix matrix*. Jadwal hasil dari initial solution ini yang nantinya akan dioptimalkan dengan menggunakan algoritma *iterated local search – hyper heuristic*. Pembentukan *initial solution* menggunkan *largest degree*, dimana *event* yang memiliki nilai *conflix* terbesar dijadwalkan terlebih dahulu.

4.5. Penerapan Algoritma *Iterated Local Search – Hyper Heuristic*

Penerapan algoritma *iterated local search – hyper heuristic* ditujukan untuk mengoptimasi jadwal dihasilkan oleh *initial solution* pada tahapan sebelumnya. Pada algoritma ini menggunakan *local search move* dan *swap* untuk pengoptimalnnya. *Hyper heuristic*, di gunakan untuk memilih *low level heuristic* yang digunakan untuk mengoptimasi jadwal *initial solution*. Hal ini bertujuan untuk mendapatkan hasil yang lebih baik dibandingkan jika hanya menggunakan satu *low level heuristic*.

Halaman ini sengaja dikosongkan.

BAB V
IMPLEMENTASI

Pada bab ini berisi tentang implemnetasi algoritma *iterated local search – hyper heuristic* yang digunakna untuk membuat penjadwalan mata kuliah otomatis. Pada penerapannya menggunakan bahas *java* dan menggunakan *tools* Neatbens IDE 8.2. adapun *method* yang digunakan adalah sebagai berikut.

5.1. Membaca *Input File*

Hal pertama yang dilakukan adalah membuat *method* untuk membaca input file .tim dan .sol. Karena pada tugas akhir ini memeiliki dua input file sesuai dengan penjelasan pada 4.2. Pertama membuat *method* utuk membaca file format .tim yang digunakan untuk membuat conflix matrix nantinya.

```
void bacaTim() throws FileNotFoundException, IOException{
    File fileTim = new File("GasalTiFix.tim");
    BufferedReader b = new BufferedReader(new FileReader(fileTim));
    String arrayTim = b.readLine();
    String [] listData = arrayTim.split(" ");
      jmlStudent = Integer.parseInt(listData[3]);
      jmlEvent = Integer.parseInt(listData[0]);
      jmlRoom = Integer.parseInt(listData[1]);
      jmlFeatures = Integer.parseInt(listData[2]);
      room = new int [jmlRoom];
        for (int i = 0; i < jmlRoom; i++) {
            room[i] = Integer.parseInt(b.readLine());
        }

        //membuat matrix event x student
        eventStudent = new int[jmlStudent] [jmlEvent];
        for (int i = 0; i < jmlStudent; i++) {
            for (int j = 0; j < jmlEvent; j++) {
                eventStudent [i][j] = Integer.parseInt(b.readLine());
            }
        }
```

```
//membuat matrix event x features
eventFeatures = new int[jmlEvent] [jmlFeatures];
for (int i = 0; i < jmlEvent; i++) {
    for (int j = 0; j < jmlFeatures; j++) {
        eventFeatures [i][j] = Integer.parseInt(b.readLine());
    }
}

//jumlah SKS tiap event
numberSks = new int[jmlEvent];
for (int i = 0; i < jmlEvent; i++) {
        numberSks [i] = Integer.parseInt(b.readLine());
}

//jumlah mahasiswa yang mengambil di tiap event
numberEventStudent = new int[jmlEvent];
for (int i = 0; i < jmlEvent; i++) {
    for (int j = 0; j < jmlStudent; j++) {
        numberEventStudent [i] = numberEventStudent [i] + eventStudent[j][i];
    }
}
```

Kode 5.1. *Method* baca file tim

Pada *method* untuk membaca file tim, ada beberapa variable yang di baca yaitu jumlah mahasiswa, jumlah *event*, jumlah ruang dan jumlah *features*. Setelah itu membuat array untuk menyimpan jumlah kapasitas masing-masing ruang. Setelah itu membuat matriks *event* dengan mahasiswa, matriks ruang dengan *features* dan matriks *event* dengan *features*. Setelah itu juga terdapat array dari jumlah SKS masing – masing *event*. Dan yang terakhir adalah array dua dimensi untuk menyimpan nilai dari jumlah mahasiswa yang mengambil tiap *event*.

Selanjutnya adalah *method* untuk membaca file input .sol. *Method* yang digunakan disini nantinya hanya untuk membca jadwal manual yang sudah ada sebelumnya seperti dijelaskan pada 4.2.

```
void bacaSol() throws FileNotFoundException, IOException{
    File fileSol = new File("RoomTsGasal.sol");
    BufferedReader c = new BufferedReader(new FileReader(fileSol));
    String arraySol = c.readLine();
        String [] listSol = arraySol.split(" ");
        jmlEventl = Integer.parseInt(listSol[0]);
        jumlahTS = Integer.parseInt(listSol[1]);
        room = new int [jmlEvent];
        String isiSol;
        String [] ruangTS = new String [jmlEventl];
        TS = new int [jmlEventl][];
        for (int i = 0; i < jmlEventl; i++) {
            isiSol = c.readLine();
            ruangTS = isiSol.split(" ");
            room[i] = Integer.parseInt(ruangTS[0]);
            TS[i] = new int[ruangTS.length-1];
            String [] temp = new String [ruangTS.length-1];
            for (int j = 1; j < ruangTS.length; j++) {
                temp[j-1] = ruangTS[j];
                TS[i][j-1] = Integer.parseInt(ruangTS[j]);
            }
        }
}
```

Kode 5.2. *Method* baca file sol

Pada *method* ini ada beberapa variable yang diidentifikasikan udari *method* yaitu jumlah *event* dan jumlah *timeslot*. Selanjutnya membuat array untuk menyimpan data ruang tiap *event* dan yang terakhir membuat array dua dimensi untuk menyimpan *timeslot* dari tiap *event*nya.

5.2. Pembuatan *Conflict matrix*

Dalam pembuatan conflix matrix terlebih dahulu membuat *method* untuk mengetahui nilai mahasiswa mengambil *event* apa saja yang disimpan dalam arraylist.

```
public void makeMatrix() {
    studentEventMatrix = new ArrayList<>(jmlStudent);
    for (int i = 0; i < jmlStudent; i++) {
        ArrayList<Integer> temp = new ArrayList<>();
        for (int j = 0; j < jmlEvent; j++) {
            if (eventStudent[i][j] == 1) {
                temp.add(j);
            }
        }
        studentEventMatrix.add(temp);
    }
}
```

Kode 5.3. *Method* nilai mahasiswa *event*

Conflix matrix perlu dibuat untuk mengetahui jumlah mahasiswa yang mengambil dua *event*. Dimana nilai baris dan kolom disesuaikan dengan jumlah *event* yang ada. Conflix matrix ini disimpan dalam array dua dimensi yang didapatkan dari *method* baca file tim di bab 5.1.

```
int [][] CM(){
    conflixMatrix = new int[jmlEvent][jmlEvent];
    for (int i = 0; i < studentEventMatrix.size(); i++) {
        for (int j = 0; j < studentEventMatrix.get(i).size() - 1; j++) {
            for (int k = j + 1; k < studentEventMatrix.get(i).size(); k++) {
                Integer eventi = (studentEventMatrix.get(i).get(j));
                Integer eventj = (studentEventMatrix.get(i).get(k));
                conflixMatrix[eventi][eventj] += 1;
                conflixMatrix[eventj][eventi] += 1;
            }
        }
    }
    return conflixMatrix;
}
```

Kode 5.4. *Method* conflix matrix

5.3. Pembuatan Initial Solution

Initial Solution adalah jadwal baru yang dibuat secara otomatis dengan menggunakan algoritma *greedy* yang nantinya akan

dioptimalkan menggunakan algoritma *Iterated Local Search – Hyper Heuristic*. *Initial Solution* merupakan jadwal baru yang memenuhi *hard constrain* sehingga bisa dikatan jadwal tersebut *feasible*. Dari jadwal *initial solution*, belum dilakukan pengoptimalan terhadapa jadwal yang telah dibuat secara otomatis dikarenakan pada tahapan pembuatan *intitial solution* belum melakukan iterasi untuk mengoptimalkan jadwal yang dihasilkan. Dalam pembuatan *initial solution*, untuk memastikan jadwal tidak berbenturan dan sesuai dengan kapistas ruang yang dimiliki maka dibuat *method* OkToSlot dan OkToRoom.

Pertama yang dilakuan adalah membuat *method* yang berfungsi untuk menginisiasi variabel – variabel yang akan digunakan untuk membuat *initial solution* yang ditunjukkan pada kode 5.5.

```
void buatJadwal(int[][] dataCM, int[] numberEventStudent, int jmlTS, int kapasitasKelas, int []room, int jmlRoom) {
    for (int i = 0; i < dataCM.length; i++) {
        MK.add(new MataKuliah(i, dataCM[i].length, numberEventStudent[i]));
    }
    remainCapacity = new int[jmlTS];
    for (int i = 0; i < remainCapacity.length; i++) {
        remainCapacity[i] = kapasitasKelas;

    }
    remainRoom = new int[jmlTS];
    for (int i = 0; i < remainRoom.length; i++) {
        remainRoom[i] = 0;
    }
    subjectTimeSlot = new ArrayList<>(numberEventStudent.length);
    for (int i = 0; i < numberEventStudent.length; i++) {
        ArrayList<Integer> temp = new ArrayList<Integer>();
        for (int j = 0; j < numberEventStudent[i]; j++) {
         temp.add(0);
        }
    subjectTimeSlot.add(temp);}
}
```

Kode 5.5 *method* buatJadwal

Pada method buatJadwal, dilakukan beberapa inisiasi variabel yaitu *arraylist* MK sebagai inisiasi dari daftar *event* yang terjadi konflik. Selanjutnya menginisiasi *array* remainCapacity yang panjangnya sesuai jumlah *timeslot* dan berisi total kapasitas kelas yang bisa digunakan. *Array* remainRoom digunakan pada *method* okToRoom untuk memastikan jumlah ruang yang digunakan tidak melebihi *timeslot.* Pada awal inisiasi remainRoom diberikan nilai nol. SubecjtTimeslot diinisiasi sebagai *arraylist* yang digunakan untuk menyimpan nilai

jadwal hasil dari *initial solution*. Pada awal inisiasi subjectTimeslot diberikan nilai nol.

```
boolean okToSlot(int lecture, int slot, int[][] dataCM, int numberSks) {
    for (int i = 0; i < dataCM[lecture].length; i++) {
        int ithAdjLecture = dataCM[lecture][i];
        for (int j = 0; j < subjectTimeSlot.get(ithAdjLecture).size(); j++) {
            for (int k = 0; k < numberSks; k++) {
                if (subjectTimeSlot.get(ithAdjLecture).get(j) == (slot + k)) {
                return false;
                }
            }
        }
    }
    ArrayList<Integer> timeslotsDay = new ArrayList<Integer>();
    for (int i = 0; i < numberSks; i++) {
        timeslotsDay.add((int)Math.ceil((double)(slot+i)/11));
    }
    Integer day = timeslotsDay.get(0);
    for (int i = 1; i < timeslotsDay.size(); i++) {
        if (day != timeslotsDay.get(i)) {
            return false;
        }
    }
    return true;
}
```

Kode 5.6. *Method* OkToSlot

Pada *method* OkToSlot, memastikan bahwa tidak ada *event* yang diambil oleh dua mahasiswa dijadwalkan secara bersamaan. Dan juga memastikan bahwa *timeslot* yang dibuat akan disesuaikan dengan jumlah SKS yang dimiliki masing – masing dari *event*. Yang dilakukan pertama kali adalah menginisiasi *event* yang akan dilakukan pengecekan jika *timeslot* yang dijadwalkan sama dengan *event* yang ada pada *conflix matrix* maka nilai yang didapat adalah *false*. Selanjutnya didalam *okToSlot* juga dilakukan pengecekan terhadap *timeslot* yang dijadwalkan dipastikan berada pada satu hari yang sama. Dengan mengubah *timeslot* yang akan dicek menjadi hari dengan menggunakan fungsi *modulo*. Pada *method* yang dibuat, ditentukan *modulo* 11 karena timeslot tiap harinya berjumlah 11 *timeslot*. Setelah itu menggunakan fungsi *if* jika nilai pada satu event itu tidak sama maka menghasilkan nilai *false*. Jika *method*

okToSlot bernilai *true* maka dialnjutkan pengecekan pada method *okToRoom*.

```
boolean okToRoom(int index, int session, int[] numberEventStudent, int numberSks, int jmlRoom) {
    int a = numberEventStudent[index];
    for (int i = 0; i < numberSks; i++) {
        if (remainRoom[(session - 1) + i] <jmlRoom) {
        remainRoom[(session - 1) + i]++;
    }
        else{
        return false;
        }
    }
    return true;
}
```

Kode 5.7. *Method* OkToRoom

Pada *method* OkToRoom berfungsi untuk memastikan bahwa jumlah mahasiswa pada tiap *event* tidak melibihi dari kapasitas ruang sehingga dapat dijadwalkan dan tidak melanggar *hard constrain* yang ada. Selain itu pada *method* okToRoom memastikan bahwa penjadwalan *event* terhadap *timeslot* tidak melebihi jumlah ruang yang bisa digunakan. Yang digunakan pertama adalah membuat fungsi *if* untuk memastikan *timeslot* yang digunakan pada penjadwalan kurang dari jumlah ruang. Sehingga jika bernilai benar menambah nilai pada *array* remainRoom.

Selanjutnya jadwal dibuat secara otomatis dengan *method* explore, yang berfungsi untuk menggabungkan fungsi OkToSlot dan OkToRoom, jika sama – sama bernilai benar atau tidak ada yang bentrok ataupun melebihi kapasitas maka *event* itu akan dijadwalakan pada *timeslot* tersebut. Apabila ada yang bentrok maka dilakukan pengecekan pada *timeslot* setelahnya sampai meneukan *timeslot* yang tidak bentrok. Inisiasi variabel realMK seabagai jadwal timeslot semestara masing – masing event yang dijadwalkan. Selanjutnya mengguanakan fungsi if untuk memastikan bahwa pada kedua method okToRoom dan okToSlot bernilai true. Sehingga jika nilai yang dihasilkan adalah true maka membuat inisiasi variabel a yang digunkan untuk menyimpan timeslot sementara sesuai dengan jumlah sks yang dimiliki event tersebut. Setelah itu variabel a dimasukkan

ke dalam arraylist subjectTimeslot untuk menghasilkan jadwal dari initial solution.

```
boolean explore(int lecture, int slot, int[][] dataCM, int[] numberEventStudent, int[] numberSks) {
    if (lecture >= dataCM.length) {
        return true;
    }
    int realMK = MK.get(lecture).getIndeks();
    if (okToSlot(realMK, slot, dataCM, numberSks) && okToRoom(realMK, slot, numberEventStudent)) {
        ArrayList<Integer> temp = new ArrayList<Integer>();
        int a = slot;
        for (int i = 0; i < numberSks[realMK]; i++) {
            temp.add(a++);
        }
            subjectTimeSlot.set(realMK, temp);
        for (int i = 1; i <= remaincapacity.length; i++) {
            if (explore(lecture + 1, i, dataCM, numberEventStudent, numberSks)) {
                return true;
            }
        }
    }
    return false;
}
```

Kode 5.8. *Method* explore

5.4. Membuat *Method* Batasan Masalah

Dalam membuat jadwal secara otomatis perlu juga dipastikan bahwa jadwal yang dibuat tidak melanggar batasan yang telah ditentukan. Dimana batasan yang dimiliki adalah empat sesuai yang dijaleaskan pada bab 4.3.

Method batasan yang pertama adalah *method* untuk memastikan bahwa semua *event* telah terjadwal. Dimana sehingga dapat dipastikan *event* yang dipilih mahasiswa sudah memiliki jadwalnya.

```
public void printSchedule() {

    for (int i = 0; i < subjectTimeSlot.size(); i++) {
          if (i <= 8) {
            System.out.print("000" + (i + 1) + " ");
        } else {
            System.out.print("00" + (i + 1) + " ");
        }
        if (subjectTimeSlot.get(i).equals(0)) {
            System.out.println("tidak terjadwal");
        } else {
            System.out.println(subjectTimeSlot.get(i));
        }
}
```

Kode 5.9. *Method* hard constrain 1

Pada *method* ini diketahui apabila ada *event* yang belum terjadwal maka akan keluar pemberitahuan bahwa *event* ke *i* tidak terjadwal. Setelah itu membuat *method* hard constrain kedua untuk memastikan bahwa *event* yang dijadwalkan pada *timeslot t* tidak bentrok dengan mahasiswa yang mengambil dua *event* atau lebih.

```
void hc2(int [][] CM){
    for (int i = 0; i < CM.length; i++) {
        System.out.println();
        for (int j = 0; j < CM.length; j++) {
            if(CM[i][j] > 0){
                System.out.println("event " + i + " dan " + j );
                System.out.println("timeslotnya " + subjectTimeSlot.get(i) + " dan " + subjectTimeSlot.get(j));
                if(subjectTimeSlot.get(i).equals(subjectTimeSlot.get(j))){
                System.out.println("not feasible");
                }
                else{
                System.out.println("feasible");
                }
            }
        }
    }
}
```

Kode 5.10. *Method* hard constrain 2

Event akan tidak feasible apabila pada conflix matrix bernilai lebih dari 0, dan ternyata ada *timeslot* yang sama sehingga *timeslot* pada kedua *event* tidak feasible. Selanjutnya yaitu memastikan bahwa jumlah mahasiswa tiap *event* tidak melebihi kapasitas ruang yang dimiliki sebagai hard constrain yang ketiga.

```
void hc3 (int [] numberEventStudent, int [] numberRoom, int [] room){
    for (int i = 0; i < numberEventStudent.length; i++) {
        if (numberEventStudent[i] > numberRoom[room[i]-1]) {
            System.out.println("Event "+ i +" not feasible");
        } else {
            System.out.println("Event "+ i +" feasible");
        }
    }
}
```

Kode 5.11. *Method* hard constrain 3

Pada *method* 5.11 saat jumlah mahasiswa melebihi kapasitas ruang maka dinyatakan *event* tersebut tidak feasible. Batasan yang terakhir memastikan bahwa tidak ada ruang dan *timeslot* yang dijadwalkan bersamaan. Jika nilai ruang dan *timeslot* lebih dari 1 maka ruang dan *timeslot* tersebut tidak feasible untuk dijadwalkan.

```
void hc4 (int jmlRoom, int jmlTS, int[][]ruangTS){
    for (int i = 0; i < jmlRoom; i++) {
        for (int j = 0; j < jmlTS; j++) {
            if (ruangTS[i][j]>1) {
                System.out.print("Ruang: " + i + ", Timeslot:" + j + " not feasible");
            } else {
                System.out.println("Ruang:" + i + ", Timeslot:" + j + " feasible\n");
            }
        }
    }
}
```

Kode 5.12. *Method* hard constrain 5

5.5. Membuat *Method Soft Constrain*

Setelah membuat jadwal yang *feasible*, maka dibuatlah *method* untuk menentukan jadwal yang dihasilkan sudah meminimalkan nilai pinalti. Method ini yang nantinya berfungsi untuk menghitung pinalti yang dimiliki dari jadwal yang telah dibuat.

Yang pertama dilakukan adalah membuat data mahasiswa mengambil *timeslot* apa saja, yang nantinya dari hasil *array* tersebut di *modulo* 11 sesuai jumlah *timeslot* perharinya. Sehingga jadwal yang dihasilkan sesuai dengan mahasiswa tersebut mengambil di hari apa saja.

```
//mengubah kedalam bentuk arraylist dan sort
studentTimeslotIsi= new ArrayList<List<Integer>>(studentEventMatrix.size());
for (int i = 0; i < studentEventMatrix.size(); i++) {
      String[] jamkuliah = jam_mahasiswa[i].split(",");
    ArrayList<Integer> temp = new ArrayList<Integer>();
    for (int j = 0; j < studentEventMatrix.get(i).size(); j++) {
        temp.add(studentEventMatrix.get(i).get(j));
    }
    Collections.sort(temp);
    studentTimeslotIsi.add(temp);
}
```

Kode 5.13 membuat *arraylist* mahasiswa *timeslot*

Pada kode 5.13, langkah pertama yang dilakukan adalah mengambil data *array* mahasiswa mengambil *event.* Selanjutnya *array* tersebut dibuat menjadi *array* yang menunjukkan mahasiwa mendapatkan *timeslot* apa saja.

```
studentTimeslotDay = new ArrayList<List<Integer>>(studentTimeslotIsi.size());
double tshari = 11;
for (int i = 0; i < studentTimeslotIsi.size(); i++) {
    ArrayList<Integer> temp = new ArrayList<Integer>();
    for(double x : studentTimeslotIsi.get(i)){
        temp.add((int)Math.ceil(x/tshari));
    }
    studentTimeslotDay.add(temp);
}
```

Kode 5.14 membuat *arraylist* mahasiswa per hari

Pada kode 5.14, *arraylist* yang telah dibuat sebelumnya di ubah menjadi hari dengan menggunakan fungsi *Math.ceil*. sehingga dapat dibuat *arraylist* yang menunjukkan mahasiswa mendapatkan *timeslot* apa saja.

Setelah itu, masuk pada *soft constrain* yang telah dibuat, yang pertama yaitu pada satu hari mahasiswa diusahakan tidak dijadwalkan lebih dari dua kali. Dan yang kedua pada satu hari mahasiswa tidak dijadwalkan hanya satu kali, yang ditunjukkan pada kode 5.14.

```
int plp2 = 0;
for (int i = 0; i < studentTimeslotIsi.size(); i++) {
    for (int j = 1; j < 6; j++) {
        int hitung = Collections.frequency(studentTimeslotDay.get(i), j);
        if (hitung >2 || hitung == 1) {
        plp2++;
        }
   }
       System.out.println(plp2);
}
```

Kode 5.15 membuat fungsi sc 1 dan sc 2

Pada kode 5.15 menggunakan fungsi *if* dimana apabila nilai lebih dari dua maka masuk pinalti untuk *soft constrain* pertama dan apabila sama dengan satu juga terhitung pinalti untuk *soft contsrain* dua.

```
//2. Soft Constrain untuk P3
int p3 = 0;
for (int i = 0; i < studentTimeslotIsi.size(); i++) {
    for(double x : studentTimeslotIsi.get(i)){
        if ((x % 11) == 0) {
            p3++;
        }
    }
      System.out.println(p3);
}
//Menghitung nilai soft constrain total
pTotal = plp2 + p3;
System.out.println(pTotal);
```

Kode 5.16 membuat fungsi sc 3

Pada kode 5.16 menggunakan *array* mahasiswa mengambil *timeslot*, apabila di *modulo* 11 sesuai jumlah *timeslot* perhari nilai nya adalah nol maka terkena pinalti. Lalu nilai pinalti tadi dijumlah untuk mendapatkan nilai pinalti total.

5.6. Penerapan Algoritma *Iterated Local Search – Hyper Heuristic*

Penerapan algoritma *iterated local search – hyper heuristic* bertujuan untuk mengoptimasi jadwal yang telah didapat dari *initial solution*. Selain itu hasil dari algoritma *iterated local search* diharapkan nilai dari pinalti (*soft constrain*) lebih kecil dibandingkan *initial solution* yang dihasilkan. Dan juga lebih baik dibandingkan jadwal manual yang telah dibuat.

5.6.1. Algoritma Local Search

Dalam menajalankan algoritma *iterated local search* diperlukan *low level heuristic* sebagai *method* untuk mendapatkan hasil penjadwalan yang lebih optimal. *Low level heuristic* yang digunakan adalah *move* dan *swap*. *Method move* dan *swap* digunakan *local search* dan *perturbation* yang dijalankan pada algoritma *iterated local search*.
Sebelum membuat *method local search* diperlukan *method* setVariabel untuk menginisiasi nilai yang akan digunakan pada *class iterated local search*.

```
public void setVariabelILS(int [][] dataCM, ArrayList<List<Integer>> subjectTimeSlot, int [] numberSks,
        int [] remainRoom, int jmlRoom, int jmlTS, int jmlEvent, int [][] eventStudent, int [][] CM, int pTotal){
    this.dataCM = dataCM;
    this.subjectTimeSlot = subjectTimeSlot;
    this.jadwalAwal = (ArrayList) subjectTimeSlot.clone();
    this.jadwalBaru = (ArrayList) subjectTimeSlot.clone();
    this.numberSks = numberSks;
    this.eventStudent = eventStudent;
    this.jmlRoom = jmlRoom;
    this.jmlTS = jmlTS;
    this.jmlEvent = jmlEvent;
    this.CM = CM;
    this.pSol = pTotal;
}
```

Kode 5.17 *method* setVariabels

Pada kode 5.17 menginisiasi variabel yang digunakan. Variabel yang dipanggil pada *method* ini adalah *array* dataCM, *arraylist* subjectTimeslot, *array* numberSks, *array* remainRoom, *array* eventStudent, *array* CM, variabel jmlRoom, variabel jmlEvent, dan variabel jmlTS. Dimana semua variabel tersebut nilainya di *return* dari kelas sebelumnya.

Selanjutnya membuat *metod* move sebagai *local search* pertama yang digukanan. *Method* move ditunjukkan pada kode 5.17

```
void move() {
this.jadwalBaru = (ArrayList) jadwalAwal.clone();
randomLecture = (int) (Math.random() * subjectTimeSlot.size());
      System.out.println(randomLecture);
    int randomSlot;
    do {
        randomSlot = (int) ((Math.random() * jmlTS)-1);
    } while (randomSlot == this.jadwalAwal.get(randomLecture).get(0));
      System.out.println(randomSlot);
    if (ilsOkToSlot(randomLecture, randomSlot, dataCM, numberSks[randomLecture]) &&
            ilsOkToRoom(randomSlot, numberSks[randomLecture], jmlRoom, jadwalBaru)) {
        ArrayList<Integer> temp = new ArrayList<Integer>();
        for (int i = 0; i < numberSks[randomLecture]; i++) {
            temp.add(randomSlot+i);
        }
            jadwalBaru.set(randomLecture, temp);
    }
      for (int i = 0; i < jadwalBaru.size(); i++) {
          System.out.println(i + "  "+jadwalBaru.get(i));
      }
}
```

Kode 5.18 *method* move

Kode 5.18 menunjukkan fungsi dari *local search move* yang digunakan untuk mengoptimalkan jadwal dari *initial solution*. Pada *method* move, dilakukan inisiasi untuk jadwal awal yang digunakan. Selanjutnya melakukan *random* untuk mengambil *event* yang akan di pindahkan *timeslotnya*. Langkah selanjutnya melakukan *random* untuk *timeslot* yang akan di pindahkan sesuai dengan hasil *random event* yang didapat sebelumnya. Setelah itu melakukan pengecekan terhadap okToSlot dan okToRoom sehingga jadwal yang dihasilkan memenuhi *hard constrain* yang telah ditentukan. Apabila nilai dari okToSlot dan okToRoom adalah *true*, maka *event random* yang didapat *timeslot* diganti dengan *random timeslot* yang didapatkan. Sehingga menghasilkan jadwal baru yang digunakan pada iterasi selanjutnya.

Low lovel heuristic selanjuntya adalah *swap*. *Method swap* digunakan untuk memindahkan *event* dengan *event* lainnya disesuaikan dengan jumlah sks yang sama. Kode program dari *method swap* dapa dilihat pada kode 5.19

```
void swap(){
    int swRandomLecture;
    this.jadwalBaru = (ArrayList) jadwalAwal.clone();
    swRandomLecture = (int) ((Math.random() * jmlEvent));
    int swRandomLecture2 = (int) ((Math.random() * jmlEvent));
      System.out.println(swRandomLecture);
      System.out.println(numberSks[swRandomLecture]);
      System.out.println(swRandomLecture2);
      System.out.println(numberSks[swRandomLecture2]);
    while (swRandomLecture == swRandomLecture2  || numberSks[swRandomLecture] != numberSks[swRandomLecture2]) {
        swRandomLecture2 = (int) ((Math.random() * jmlEvent));
    }
    if (ilsOkToRoom(jadwalBaru.get(swRandomLecture2).get(0), numberSks[randomLecture], jmlRoom, jadwalBaru )
            && ilsOkToRoom(jadwalBaru.get(swRandomLecture).get(0), numberSks[randomLecture], jmlRoom, jadwalBaru)) {
        List<Integer> temp = jadwalBaru.get(swRandomLecture);
        jadwalBaru.set(swRandomLecture,jadwalBaru.get(swRandomLecture2));
        jadwalBaru.set(swRandomLecture2, temp);
    }
      for (int i = 0; i < subjectTimeSlot.size(); i++) {
          System.out.println(i + "  "+jadwalBaru.get(i));
      }
}
```

Kode 5.19 *method* swap

Pada *method* swap dilakukan dua inisiasi untuk mendapatkan dua nilai *random* dari *event* yang akan di *swap*. Pada *method* ini juga dilakukan inisiasi jadwal awal yang akan di *swap* sesuai dengan *event* yang dihasilkan. Pertama melakukan pengecekan dengan menggunkana fungsi *while* dimana jika nilai *random* dari *event* satu dan *event* kedua bernilai sama maka dilakukan *random* lagi untuk nilai *event* kedua. Dan jika nilai sks dari kedua *event* juga tidak sama maka dilakukan *random* lagi pada *event* yang kedua. Setelah didapatkan nlai *random* yang berbeda dari kedua *event* maka menggunakan fungsi *if* untuk melakukan pengecekan terhadap okToRoom dari kedua *event*. Jika berniali *true* maka jadwal baru dibuat dengan menggantikan nilai *random event* pertama ke *timeslot* nilai *random event* kedua, dan nilai *random event* kedua menggantikan nilai *timeslot event random* pertama.

5.6.2. Algoritma *Iterated Local Search*

Pada penerapan algoritma *iterated local search*, menggunakan jadwal *initial solution* sebagai solusi awal yang dioptimalkan. Dengan menggunakan *local search move* untuk mendapatkan nilai pinalti yang lebih kecil. Sehingga jadwal yang didapat menjadi jadwal sementara. Selanjutnya dilakukan *perturbation* dengan menggunakan *method* swap dengan memindahkan *event* satu dengan yang lain disesuaikan dengan jumlah sks

yang dimiliki. Hasil yang diterima apabila nilai dari pinlati lebih kecil dari sebelumnya sehingga menghasilkan jadwal baru yang lebih optimal.
Pertama dilakukan inisiasi untuk membatasi *iterasi* yang dilakukan seperti kode 5.20.

```
int counterSearch = 1;
int maxSearchLoop = 1000;
```

Kode 5.20 inisiasi *iterasi*

Pada kode 5.20, digunakan untuk menginisiasi banyaknya *iterasi* yang akan dilakukan. setelah itu masuk pada *local search* pertama yaitu *move*.

```
while(counterSearch <= maxSearchLoop){
    move();
        int oldPinaltiMove = hitungPinalti(jadwalAwal);
        int newPinaltiMove = hitungPinalti(jadwalBaru);
        if (newPinaltiMove < oldPinaltiMove) {
            jadwalAwal = (ArrayList) jadwalBaru.clone();
        }
```

Kode 5.21 ils *move*

Kode 5.21 menunjukkan fungsi *local search move*, dimana *move* dilakukan sebanyak *iterasi* yang telah dilakukan. setelah itu melakukan inisiasi untuk menghitung niali pinalti awal dengan nilai pinalti yang baru. Menggunakan fungsi *if*, jika nilai pinalti yang beru lebih kecil dari yang lama, maka jadwal awal untuk memulai *iterasi* yang baru akan diganti dengan jadwal dari nilai pinalti yang baru.
Selanjutnya melakukan *perturbation* untuk memperbaiki nilai dari pinalti baru yang sudah didapat sebelumnya sehingga menghasilkan nilai pinalti yang baru yang lebih kecil.
Pada kode 5.12 dapat dilihat bahwa jadwal awal yang didapat dari *local search* sebelumnya di*update* untuk dilakukan inisiasi untuk memulai *method* swap yang ada pada *perturbation*. Setelah melakukan *swap* dilakukan inisiasi untuk melihat nilai pinalti baru yang dihasilkan, dan menghitung pinalti baru dari *method* swap.

```
swap();
    int oldPinaltiSwap = hitungPinalti(jadwalAwal);
    int newPinaltiSwap = hitungPinalti(jadwalBaru);
    if (newPinaltiSwap < oldPinaltiSwap) {
        jadwalAwal = (ArrayList) jadwalBaru.clone();
    }
```

Kode 5.22 ils *swap*

Menggunakan fungsi *if* untuk mengganti menjadi jadwal baru apabila nilai pinalti yang baru dari *swap* lebih kecil dibandingkan nilai pinalti sebelumnya. Proses ini dilakukan sejumlah *iterasi* yang telah dilakukan.

```
hc2(jadwalAwal);
    pr.println("Nilai Pinalti Yang baru : "+hitungPinalti(jadwalAwal)+"\n");
for (int i = 0; i < jadwalAwal.size(); i++) {
    pr.println("Event ke " + (i+1) + " Timeslot ke \t"+ jadwalAwal.get(i));
}
```

Kode 5.23 cek jadwal

Pada kode 5.23, digunakan untuk melakukan pengecekan dari jadwal baru yang telah dioptimasi. Pertama melakukan pengecekan dengan menggunakan method hc2 untuk memastikan bahwa jadwal tersebut adalah jadwal yang *feasible*. Selanjutnya dihitung nilai akhir dari pinalti setelah dialkukan optimasi dan dialkukan fungsi untuk mencetak jadwal hasil optimasi dengan menggunakan algoritma *iterated local search*.

Halaman ini sengaja dikosongkan.

BAB VI
HASIL DAN PEMBAHASAN

Bab ini akan menjelaskan hasil yang didapatkan dari penelitian ini dan pembahasan secara keseluruhan yang didapatkan dari penelitian.

6.1. Data Uji Coba

Data yang digunakan adalah data mahasiswa serta jadwal mata kuliah semester gasal dan semester genap tahun ajaran 2017-2018 Departemen Sistem Informasi, Institut Teknologi Sepuluh Nopember. Pada data yang didapatkan, dapat diketahui bahwa pada semester gasal jumlah event yang dimiliki sebanyak 111 dan jumlah mahasiswa adalah 687, sedangkan pada semester genap jumlah event yang dimiliki sebanyak 601 dengan jumlah mahasiswa sebanyak 601.

6.2. Lingkungan Uji Coba

Lingkungan uji coba merupakan kriteria perangkat pengujian yang digunakan untuk penerapan penjadwalan mata kuliah otomatis dengan menggunakan algoritma *iterated local search – hyper heuristic* dalam penyelesaian permasalahan pada tugas akhir ini. Lingkungan uji coba meliputi perangkat keras dan perangkat lunak yang digunakan. Spesifikasi dari perangkat keras yang digunakan dalam implementasi metode ditunjukkan pada Tabel 6.1.

Tabel 6.1 Lingkungan Uji Coba Perangkat Keras

Perangkat Keras	Spesifikasi
Jenis	Notebook
Processor	Intel® Core™ i5-3510M CPU @ 2.50 GHz 2.50 GHz
RAM	4.00 GB (3.89 usable)
System type	64-bit Operating System, x64-based processor
Hard Disk Drive	750 GB

Selain perangkat keras, terdapat lingkungan perangkat lunak yang digunakan dalam implementasi metode seperti yang ditunjukkan pada Tabel 6.2.

Tabel 6.2 Lingkungan Uji Coba Perangkat Lunak

Perangkat Lunak	Spesifikasi
Sistem Operasi	Windows 10
Bahasa Pemrograman	Java
Tools	Netbeans IDE 8.2 Microsoft Excel 2016

6.3. Fungsi Tujuan dari Jadwal Manual

Fungsi tujuan yang digunakan dalam tugas akhir ini adalah meminimalkan *pinalti*. Sebelum membuat jadwal mata kuliah otomatis, dilakukan perhitungan nilai *pinalti* dari jadwal mata kuliah yang dibuat secara manual berdasarkan persamaan berikut dengan kode yang telah dijelaskan pada bab sebelumnya.

$$P = P_1 + P_2 + P_3$$

Dari hasil perhitungan, diketahui bahwa nilai *pinalti* untuk jadwal mata kuliah yang dibuat secara manual sebesar 1579 untuk semester gasal dan 1360 untuk semester genap. Namun, jadwal mata kuliah yang dibuat secaramanual belum memenuhi semua hard constrain yang diberikan.

6.4. Hasil Pembuatan Initial Solution

Initial Solution yang dihasilkan dari algoritma *greedy* yang di implementaskan di bahasa *java* menghasilkan jadwal yang *feasible*. Solusi awal tersebut adalah solusi yang telah memenuhi batasan dimana tidak ada jadwal mata kuliah yang bentrok dan tidak melebihi total kapasitas ruang. Tabel 6.3 menunjukkan hasil jadwal mata kuliahyang diperoleh dari *intial solution.*

Tabel 6.3 Jadwal mata kuliah

Kode	Gasal	Genap
	Timeslot	Timeslot
001	1 2 3	1, 2, 3
002	4 5 6	4, 5, 6
003	1 2 3	1, 2, 3
004	1 2 3	7, 8, 9
005	1 2 3	1, 2
006	4 5 6	1, 2
007	1 2 3	1, 2
008	7 8 9	1, 2
009	4 5 6	1, 2
010	7 8 9	3, 4, 5
011	12 13 14	3, 4, 5
012	7 8 9	3, 4, 5
013	7 8 9	7, 8, 9
014	7 8 9	4, 5, 6
015	12 13 14	4, 5, 6
016	4 5 6	6, 7, 8
017	1 2 3	12, 13, 14
018	15 16 17	15, 16, 17
019	15 16 17	18, 19, 20
020	18 19 20	23, 24, 25
021	12 13 14	15, 16, 17
022	23 24 25	12, 13, 14
023	26 27 28	26, 27, 28
024	15 16 17	26, 27, 28
025	12 13 14	26, 27, 28
026	18 19 20	26, 27, 28
027	18 19 20	12, 13, 14
028	23 24 25	15, 16, 17
029	10 11	15, 16, 17
030	1 2	7, 8, 9
031	1 2	18, 19, 20
032	1 2	10, 11
033	1 2	6, 7
034	3 4	10, 11
035	10 11	10, 11
036	18 19 20	10, 11

Kode	Gasal	Genap
	Timeslot	Timeslot
037	12 13 14	18, 19, 20
038	23 24 25	3, 4
039	15 16 17	3, 4
040	26 27 28	3, 4
041	7 8 9	6, 7
042	29 30 31	12, 13
043	29 30 31	10, 11
044	29 30 31	10, 11
045	29 30 31	9, 10
046	4 5 6	10, 11
047	7 8 9	23, 24, 25
048	26 27 28	6, 7, 8
049	7 8 9	29, 30, 31
050	3 4	29, 30, 31
051	3 4	18, 19, 20
052	3 4	29, 30, 31
053	4 5	14, 15, 16
054	10 11	23, 24, 25
055	5 6	34, 35, 36
056	12 13	34, 35, 36
057	5 6	34, 35, 36
058	21 22	34, 35, 36
059	5 6	34, 35, 36
060	7 8 9	37, 38, 39
061	7 8 9	37, 38, 39
062	12 13 14	37, 38, 39
063	12 13 14	37, 38, 39
064	12 13 14	40, 41, 42
065	15 16 17	15, 16, 17
066	15 16 17	8, 9, 10
067	15 16 17	5, 6, 7
068	12 13 14	40, 41, 42
069	26 27 28	40, 41, 42
070	23 24 25	40, 41, 42
071	23 24 25	45, 46, 47
072	15 16 17	23, 24, 25
073	34 35 36	23, 24, 25

Kode	Gasal	Genap
	Timeslot	Timeslot
074	34 35 36	23, 24, 25
075	34 35 36	23, 24, 25
076	34 35 36	18, 19, 20
077	21 22	48, 49, 50
078	23 24	26, 27, 28
079	21 22	12, 13, 14
080	32 33	12, 13, 14
081	21 22	12, 13, 14
082	32 33	23, 24, 25
083	21 22	48, 49, 50
084	32 33	12, 13, 14
085	37 38 39	34, 35, 36
086	37 38 39	26, 27, 28
087	37 38 39	34, 35, 36
088	37 38 39	34, 35, 36
089	37 38 39	34, 35, 36
090	37 38 39	12, 13, 14
091	26 27 28	21, 22
092	29 30 31	21, 22
093	26 27 28	21, 22
094	18 19 20	
095	18 19 20	
096	40 41 42	
097	45 46 47	
098	40 41 42	
099	40 41 42	
100	40 41 42	
101	48 49 50	
102	40 41 42	
103	51 52 53	
104	45 46 47	
105	18 19 20	
106	45 46 47	
107	26 27 28	
108	48 49 50	
109	43 44	
110	10 11	

Kode	Gasal	Genap
	Timeslot	Timeslot
111	21 22	

Jadwal yang dihasilkan oleh *initial solution* memiliki nilai *pinalti* sebesar 2057 untuk semester gasal dan 1625 untuk semester genap.

6.5. Hasil Penerapan Algoritma *Iterated Local Search*

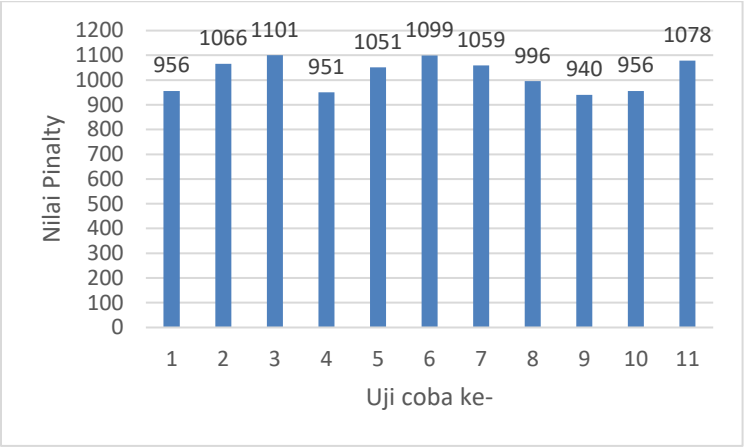
Algoritma *iterated local search* digunakan untuk mengoptimasi jadwal mata kuliah hasil dari penerapan *initial solution*. Untuk mencari nilai *pinalti* sekecil mungkin, maka dilakukan beberapa uji coba dengan mengubah parameter yang digunakan dalam algoritma *iterated local search*. Parameter yang digunakan dengan perubahan iterasi dari mulai 1000, 5000 dan 10000. Parameter ini disesuaikan dengan paper yang menjadi acuan dalam penelitian ini[4].

6.5.1. Skenario Uji Coba 1 : Iterasi 1000

Pada scenario uji coba pertama, menggunakan iterasi sejumlah 1000 untuk mendapatkan nilai pinalti yang lebih baik. Penulis dalam hal ini melakukan uji coba sebanyak sebelas kali.

6.5.1.1. Semester gasal

Pada uji coba ini variabel maxSearchLoop dilakukan inisiasi sejumlah 1000, dimana hal ini menjadikan iterasi yang dilakukan menjadi 1000 kali. Pada uji coba ini dilakukan sebanyak sebelas kali uji coba.



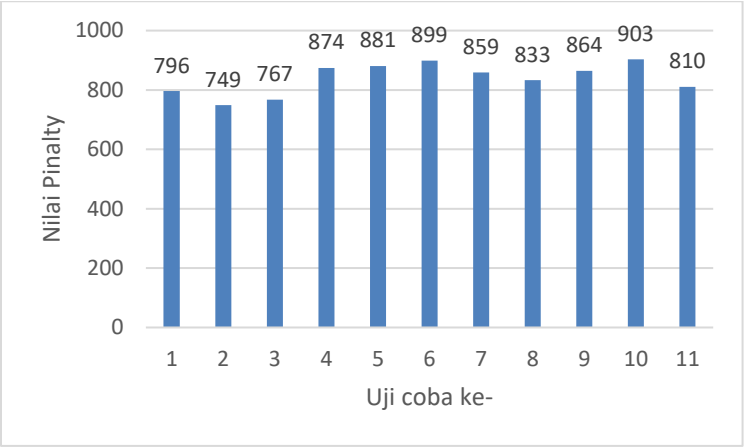
Grafik 6.1 Semester gasal iterasi 1000

Grafik hasil uji coba pada semester gasal dengan iterasi 1000 dapat dilihat pada grafik 6.1. dimana didapatkan nilai pinalti terbesar adalah 1101 dan nilai terkecil yang didapat adalah 940. Sedangkan nilai rata-rata dari hasil pinalti adalah 1023.

6.5.1.2. Semester genap

Nilai variabel maxSearchLoop yang diinisiasikan adalah 1000. Dimana hal ini berarti iterasi yang dilakukan sebanyak 1000 kali, dan uji coba ini dilakukan sebanyak sebelas kali.

Dari hasil pada grafik 6.2 dapat dilihat bahwa nilai pinalti terbesar adalah 903. Nilai minimal yang dimiliki hasil ini adalah 749 dengan hasil rata-rata dari pinalti semester genap dengan iterasi sebanyak 1000 kali adalah 839,5455. Dari hal ini dengan melakukan iterasi sebanyak 1000 kali telah terjadi peningkatan dari nilai pinalti jadwal *initial solution* yang dihasilkan sebelumnya.



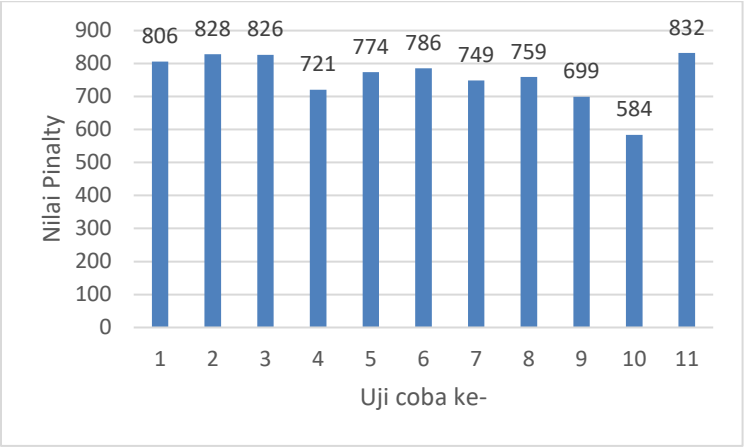
Grafik 6.2 Semester genap iterasi 1000

6.5.2. Skenario Uji Coba 2 : Iterasi 5000

Pada uji coba kedua, dilakukan perubahan pada jumlah iterasi. Nilai variabel dari maxSearchLoop dirubah menjadi 5000 sehingga menghasilkan iterasi sebanyak 5000 kali. Uji coba kedua, dilakukan sebelas kali pada masing-masing semester.

6.5.2.1. Semester gasal

Pada semester gasal, hasil dari *initial solution* dilakukan iterasi sebanyak 5000 kali dengan merubah maxSeacrhLoop. Hasil dari optimasi ini dibuat grafik yang ditunjukkan pada grafik 6.3

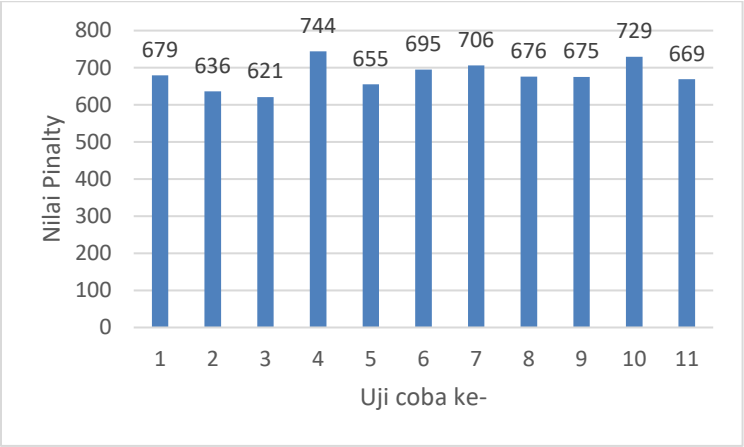


Grafik 6.3 Semester gasal 5000 iterasi

Dari grafik 6.3, dapat dilihat bahwa nilai pinalti terbesar adalah 832 dan nilai pinalti terkecil adalah 584. Hal ini menunjukkan perubahan yang lebih baik dari banyaknya jumlah iterasi. Sedangkan untuk rata – rata pinalti yng dihasilkan adalah 760,364.

6.5.2.2. Semester genap

Hal yang sama dilakukan pada hasil *initial solution* untuk semester genap. Dimana dilakukan iterasi sebanyak 5000 kali dan dilakukan uji coba sebanyak sebelas kali.
Hasil dari uji coba yang dilakukan dapat dilihat pada grafik 6.4. Dari hasil grafik 6.4 dapat dilihat bahwa nilai pinalti terkecil adalah 621 dan nilai pinalti terbesar adalah 744. Hal ini menunjukkan nilai pinalti yang dihasilkan lebih baik. Dan nilai rata – rata dari pinalti yang dihasilkan adalah 680,4545.



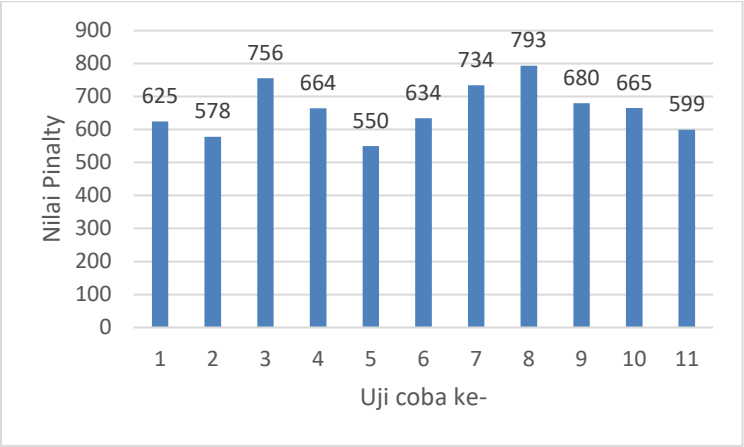
Grafik 6.4 Semester genap 5000 iterasi

6.5.3. Skenario Uji Coba 3 : Iterasi 10000

Skenario untuk uji coba yang ketiga adalah dengan merubah iterasi yang ada menjadi 10000 kali perulangan. Hal ini dilakukan dengan merubah nilai parameter maxSearchLoop menjadi 10000. Dengan dilakukan uji coba sebanyak sebelas kali untuk melihat hasil dari scenario tersebut.

6.5.3.1. Semester gasal

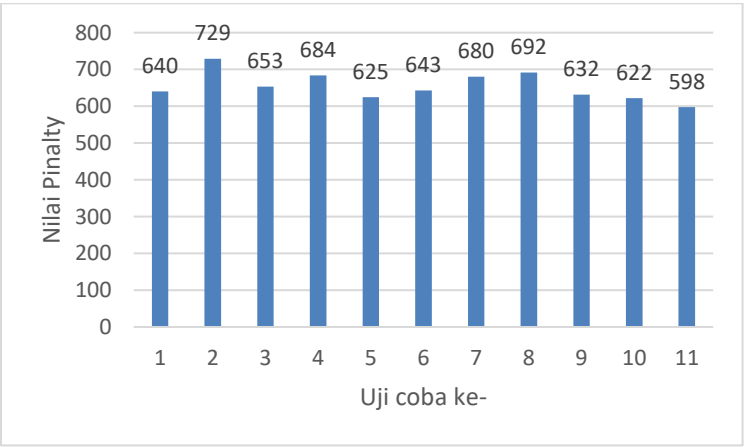
Pada jadwal semester gasal dilakukan iterasi sebanyak 10000 kali untuk mendapatkan jadwal yang lebih optimal. Pergantian masih sama dilakukan dengan merubah nilai maxSearchLopp menjadi 10000. Hasil tersebut ditunjukkan pada grafik 6.5.
Dari hasil tersebut dapat diketahui nilai pinalti terbesar yang didaptakan adalah 793. Sedangkan untuk nilai pinalti terkecil dapat menghasilkan 550. Dan rata – rata dari uji coba yang dilakukan adalah 661,636. Hasil tersebut mengalami peningkatan dari kualitas nilai pinalti yang dihasilkan.



Grafik 6.5 Semester gasal 10000 iterasi

6.5.3.2. Semester genap

Pada jadwal semester genap hal sama dilakukan dengan erubah nila maxSeacrhLoop menjadi 10000 untuk dialkukan iterasi sebanyak 10000 kali. Dimana uji coba dilakukan sebanyak sebelas kali untuk membandingkan hasil uji coba. Hasil dari ujo coba dapat dilihat pada grafik 6.6.



Grafik 6.6 Semester genap 10000 iterasi

Dari grafik 6.6, dapat diketahui bahwa nilai pinalti terbesar yang dimiliki adalah 729 dan nilai pinalti terkecil yang didapat adalah 598. Rata – rata dari nilai pinalti yang dihasilkan adalah 654,3636. Hal ini dapat menunjukkan semakin baik hasil pinalti yang diperoleh.

6.6. Perbandingan Hasil Uji Coba

Perbandingan hasil uji coba dilakukan untuk membantu membuat kesimpulan bagi penulis. Dimana pada perbandingan hasil uji coba, yang pertama dibandingkan dengan hasil uji coba yang telah dilakukan sesuai jumlah iterasi dan yang kedua membandingkan dengan hasil optimasi menggunakan algoritma yang lain yaitu *hill climbing*.

6.6.1. Perbandingan Hasil Uji Coba Iterasi

Pada langkah pertama mebandingkan hasil uji coba sesuai dengan iterasi yang telah dilakukan. hal ini bertujuan untuk melihat pengaruh jumlah iterasi yang dilakukan terhadap hasil optimasi yang didapatkan. Hasil yang diabdingkan dari iterasi 1000, 5000, dan 10000. Yang dilakukan masing – masing sebanyak sebelas kali.

Tabel 6.4 Perbandingan hasil iterasi semester gasal

Percobaan	**Iterasi**		
	1000	**5000**	**10000**
1	956	806	625
2	1066	828	578
3	1101	826	756
4	951	721	664
5	1051	774	550
6	1099	786	634
7	1059	749	734
8	996	759	793
9	940	699	680
10	956	584	665
11	1078	832	599
MAX	1101	832	793

Percobaan	Iterasi		
	1000	5000	10000
MIN	940	584	550
AVG	1023	760.3636364	661.6363636

Tabel 6.4 menunjukkan perbandingan dari hasil iterasi pada semester gasal yang dilakukan sebanyak sebelas kali. Dari data tabel tersebut. Dapat dilihat rata – rata pinalti nilai terkecil didaptkan pada iterasi 10000 dengan nilai pinalti 661,636. Semakin kecil pinalti yang diperoleh maka jadwal yang dihasilkan semakin baik. Dari data tersebut jadwal terbaik didaptkan pada iterasi 10000 percobaan ke lima dengan nilai pinalti adalah 550. Dan nilai pinalti terbesar didapat pada itersi 1000 percobaan ke tiga. Jika dibandingkan dari nilai pinalti jadwal manual yaitu 1597 dan nilai pinalti jadwal *initial solution* 2057 maka jadwal yang didapatkan sudah lebih baik lagi.

Tabel 6.5 Perbandingan hasil iterasi semester genap

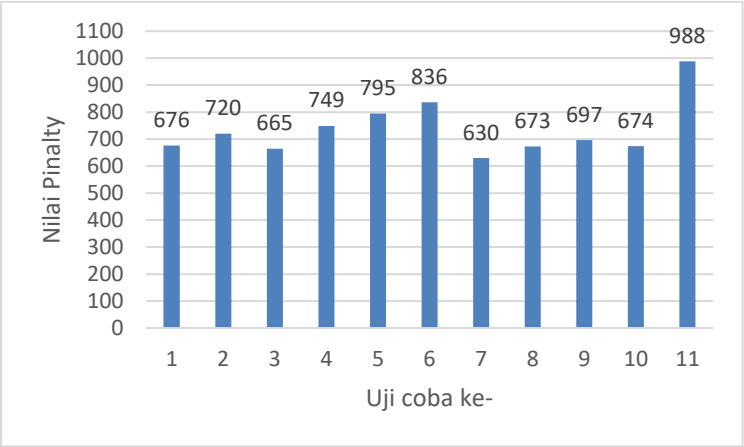
Percobaan	Iterasi		
	1000	5000	10000
1	796	679	640
2	749	636	729
3	767	621	653
4	874	744	684
5	881	655	625
6	899	695	643
7	859	706	680
8	833	676	692
9	864	675	632
10	903	729	622
11	810	669	598
MAX	903	744	729
MIN	749	621	598
AVG	839.5454545	680.4545455	654.3636364

Tabel 6.5 menunjukkan hasil perbandingan dari optimasi jadwal semester genap yang diiterasi 1000, 5000, dan 1000 kali.

Pada hasil tersebut, didapatkan rata – rata nilai pinalti terkecil ada pada saat iterasi 10000. Nilai pinalti terkecil yang diapat adalah 598 yang ada pada uji coba iterasi 10000 percobaan ke sebelas. Untuk nilai pinalti terbesar ada pada uji coba iterasi 1000 percobaan ke sepuluh dengan nilai 903. Hal ini dapat diketahui bahwa nilai pinalti terkceil diapat dari jumlah iterasi terbesar. Jika dibandingkan dengan nilai pinalti pada jadwal yang dibuat manual yaitu 1360 dan nilai pinalti pada jadwal *initial solution* yaitu 1625, maka hasil otimasi yang dilakukan sudah memperbaiki jadwal yang sudah ada sebelumnya.

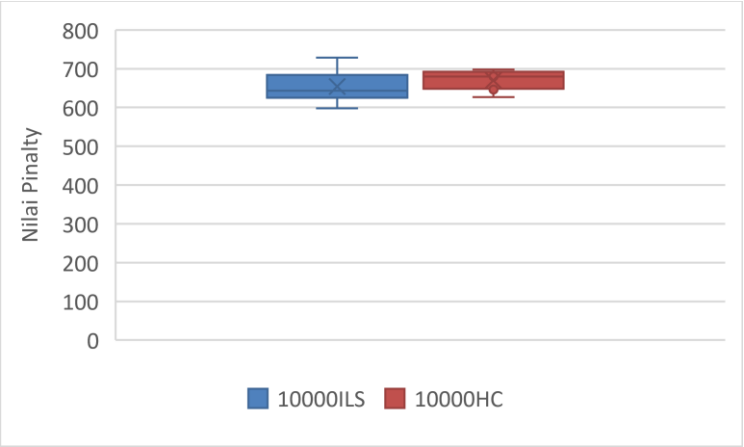
6.6.2. Perbandingan Hasil Uji Coba dengan Algoritma Hill Cimbing

Perbandingan dengan algoritma *hill climbing* diperlukan untuk melihat performa dari algoritma *iterated local search* dikarenakan algoritma *iterated local search* merupakan *improvement* dari algortima *hill climbing*.
Pada algoritma *hill climbing* dilakukan uji coba sebanyak sebelas kali pada iterai 10000. Hasil dari uji coba dapat dilihat pada grafik 6.7.



Grafik 6.7 Semester gasal *hill climbing* 10000

Pada grafik 6.7, dapat dilihat bahwa nilai pinalti terbesar adalah 988 dan nilai pinalti terkecil adalah 630. Dengan rata – rata nilai pinalti 736,636. Setelah itu, dilakukan perbandingan dengan grafik *box plot* untuk menilai persebaran yang ada pada masing – masing algoritma.



Grafik 6.8 Semester gasal perbandingan ils dan hc 10000 iterasi

Pada grafik 6.8, dapat dilihat bahwa persebaran dari algoritma iterated local search lebih baik dikarenakan nilai pinalti yang didapat lebih kecil dibandingkan hasil dari algoritma *hill climbing*.

Tabel 6.6 Hasil algortima ils dan hc semester gasal

Uji Coba ke -	Algoritma	
	ILS	HC
1	625	676
2	578	720
3	756	665
4	664	749
5	550	795
6	634	836
7	734	630

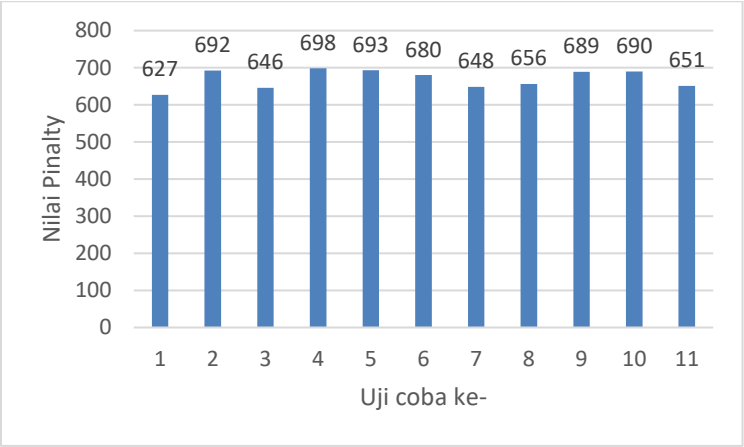
Uji Coba ke -	Algoritma	
	ILS	HC
8	793	673
9	680	697
10	665	674
11	599	988
MAX	793	988
MIN	550	630
AVG	661.6363636	736.6363636

Tabel 6.6 menunjukkan daftar hasil uji coba dari masing – masing algoritma. Data ringkasan dari perbadingan hasil optimasi masing – masing algoritma pada semester gasal dapat dilihat pada tabel 6.7.

Tabel 6.7 Ringkasan perbandingan algoritma semester gasal

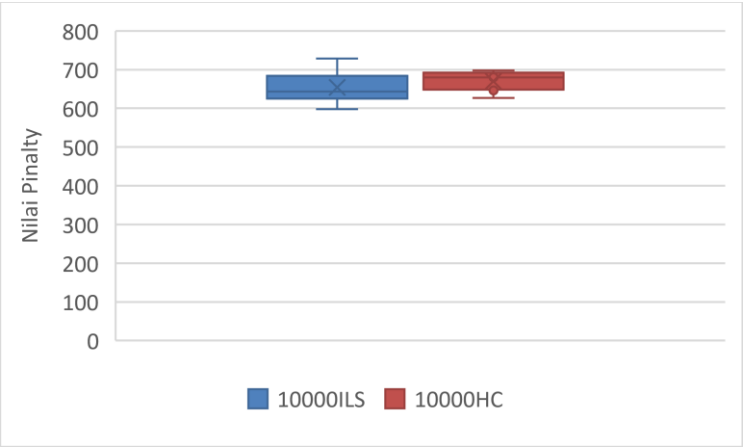
Algoritma	Rata - rata	Max	Min
ILS	661.6363636	793	550
HC	736.6363636	988	630

Dari hasil diatas, maka didapatkan jadwal terbaik dari uji coba yang telah dilakukan dengan nilai pinalti 550 menggunkan algortima *iterated local seacrh* dengan iterasi 10000.
Selanjutnya dilakukan perbandingan dengan menggunkan jadwal pada semester genap. Algoritma *hill climbing* pada jadwal semester genap dilakukan uji coba sebelas kali dengan iterasi 10000.



Grafik 6.9 Semester genap *hill climbing* 10000 iterasi

Pada grafik 6.9 dapat dilihat nilai pinalti terbesar yang didapat adalah 698 dan nilai pinalti terkecil yang diperoleh adalah 627. Rata – rata dari hasil uji coba tersebut adalah 670. Setelah itu hasil dari uji coba tersebut dibuat perbadingan dengan algoritma *iterated local search* dengan menggunakan grafik *box plot*.



Grafik 6.10 Perbandingan algoritma semester genap

Pada grafik 6.10, dapat dilihat bahwa persebaran dari algoritma *iterated local search* lebih baik dikarenakan nilai pinalti yang didapat lebih kecil dibandingkan hasil dari algoritma *hill climbing*.

Tabel 6.8 Hasil algoritma ils dan hc semester genap

Uji Coba ke -	Algoritma	
	ILS	HC
1	640	627
2	729	692
3	653	646
4	684	698
5	625	693
6	643	680
7	680	648
8	692	656
9	632	689
10	622	690
11	598	651
MAX	729	698
MIN	598	627
AVG	654.3636364	670

Tabel 6.8 menunjukkan daftar hasil uji coba dari masing – masing algoritma. Data ringkasan dari perbadingan hasil optimasi masing – masing algoritma pada semester gasal dapat dilihat pada tabel 6.9.

Tabel 6.9 Ringkasan perbandingan algoritma semester genap

Algoritma	Rata - rata	Max	Min
ILS	654.3636364	729	598
HC	670	698	627

Dari hasil pada tabel 6.9, jadwal terbaik pada semester genap dihaslkan dengan menggunakan algoritma *iterated local search* dengan nilai pinalti 598.

Tabel 6.10 menunjukkan jadwal baru terbaik yang telah dibuat otomatis dengan menerapkan algoritma *iterated local search*

yang pada semester gasal menghasilkan pinalti 550 dan pada semester genap menghasilkan pinalti 598.

Tabel 6.10 Hasil jadwal optimasi

Kode	Gasal	Genap
	Timeslot	Timeslot
001	34 35 36	3 4 5
002	34 35 36	1 2 3
003	1 2 3	18 19 20
004	1 2 3	1 2 3
005	1 2 3	1 2
006	4 5 6	1 2
007	45 46 47	1 2
008	18 19 20	1 2
009	29 30 31	8 9
010	29 30 31	40 41 42
011	23 24 25	15 16 17
012	12 13 14	12 13 14
013	23 24 25	34 35 36
014	15 16 17	4 5 6
015	23 24 25	12 13 14
016	45 46 47	18 19 20
017	26 27 28	15 16 17
018	37 38 39	26 27 28
019	12 13 14	12 13 14
020	26 27 28	23 24 25
021	29 30 31	12 13 14
022	29 30 31	29 30 31
023	7 8 9	23 24 25
024	37 38 39	15 16 17
025	12 13 14	15 16 17
026	40 41 42	48 49 50
027	18 19 20	49 50 51
028	18 19 20	3 4 5
029	8 9	6 7 8
030	15 16	48 49 50
031	16 17	23 24 25
032	16 17	47 48
033	1 2	3 4

Kode	Gasal	Genap
	Timeslot	Timeslot
034	34 35	2 3
035	32 33	3 4
036	18 19 20	51 52
037	12 13 14	15 16 17
038	12 13 14	3 4
039	18 19 20	27 28
040	15 16 17	15 16
041	26 27 28	1 2
042	1 2 3	29 30
043	7 8 9	9 10
044	4 5 6	6 7
045	7 8 9	8 9
046	7 8 9	6 7
047	51 52 53	23 24 25
048	15 16 17	12 13 14
049	7 8 9	29 30 31
050	3 4	29 30 31
051	39 40	23 24 25
052	3 4	29 30 31
053	4 5	14 15 16
054	3 4	18 19 20
055	5 6	34 35 36
056	37 38	34 35 36
057	3 4	34 35 36
058	15 16	37 38 39
059	5 6	45 46 47
060	12 13 14	3 4 5
061	18 19 20	7 8 9
062	12 13 14	40 41 42
063	12 13 14	4 5 6
064	37 38 39	41 42 43
065	15 16 17	8 9 10
066	12 13 14	8 9 10
067	15 16 17	5 6 7
068	18 19 20	37 38 39
069	26 27 28	37 38 39
070	23 24 25	6 7 8

Kode	Gasal	Genap
	Timeslot	Timeslot
071	26 27 28	26 27 28
072	15 16 17	23 24 25
073	37 38 39	26 27 28
074	4 5 6	26 27 28
075	7 8 9	23 24 25
076	29 30 31	12 13 14
077	12 13	40 41 42
078	23 24	7 8 9
079	15 16	26 27 28
080	26 27	12 13 14
081	51 52	7 8 9
082	32 33	23 24 25
083	51 52	26 27 28
084	49 50	34 35 36
085	15 16 17	34 35 36
086	34 35 36	18 19 20
087	29 30 31	12 13 14
088	37 38 39	37 38 39
089	7 8 9	4 5 6
090	7 8 9	34 35 36
091	7 8 9	8 9
092	1 2 3	31 32
093	1 2 3	45 46
094	23 24 25	
095	26 27 28	
096	40 41 42	
097	37 38 39	
098	40 41 42	
099	26 27 28	
100	40 41 42	
101	26 27 28	
102	15 16 17	
103	7 8 9	
104	4 5 6	
105	48 49 50	
106	45 46 47	
107	4 5 6	

Kode	Gasal	Genap
	Timeslot	Timeslot
108	34 35 36	
109	1 2	
110	9 10	
111	1 2	

.

BAB VII
KESIMPULAN DAN SARAN

Bab ini akan menjelaskan kesimpulan dari penelitian, beserta saran yang dapat bermanfaat untuk perbaikan di penelitian selanjutnya.

7.1. Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan, maka dapat disimpulkan sebagai berikut :

1. Algoritma *iterated local search – hyper heuristic* dapat digunakan untuk membuat jadwal mata kuliah di Departemen Teknik Industri, Institut Teknologi Sepuluh Nopember Surabaya. Jadwal yang dihasilkan adalah jadwal yang *feasible*, dimana jadwal tersebut sudah memenuhi batasan tidak ada jadwal yang bentrok dan tidak melebihi jumlah ruang yang tersedia.
2. Algoritma *iterated local search – hyper heuristic* mampu menghasilkan jadwal mata kuliah yang lebih optimal apabila dibandingkan dengan jadwal yang dihasilkan secara manual. Hal ini dibuktikan dengan nilai pinalti jadwal manual yang didapat adalah 1579 pada semester gasal dapat diturunkan menjadi 550. Dan untuk semester genap nilai pinalti awal pada jadwal manual adalah 1360 dapat diturunkan menjadi 598.
3. Pembuatan jadwal mata kuliah otomatis dengan algoritma *iterated local search – hyper heuristic* dapat menghasilkan jadwal mata kuliah yang *feasible* dan dapat mengoptimalkan jadwal mata kuliah.
4. Penjadwalan mata kuliah otomatis dapat menghasilkan jadwal yang lebih baik dibandingkan dengan jadwal yang dibuat secara manual.

7.2. Saran

Saran yang diberikan berdasarkan hasil pengerjaan tugas akhir untuk penelitian selanjutnya yaitu :

1. Pada pengerjaan tugas akhir ini, event adalah penjelasan dari mata kuliah terpilih dengan dosen pengajar. Sehingga pada penelitian selanjutnya dapat menambah parameter dosen, seperti jadwal dosen yang memiliki kepentingan, sehingga tidak bisa mengajar pada hari – hari tertentu dapat dijadikan tambahan untuk *soft constrain*.
2. Pada penelitian ini, tampilan yang ditunjukan hanya berupa tulisan. Pada penelitian berikutnya algoritma dapat diintegrasikan dengan aplikasi tertentu untuk sekaligus menghasilkan jadwal yang dapat dengan mudah dipahami pengguna.
3. Pada penelitian ini, penulis menggunakan algoritma *iterated local search – hyper heuristic* untuk menyelesaikan permasalahan penjadwalan mata kuliah. Pada penelitian berikutnya dapat menggunakan algoritma yang berbeda seperti *great deluge* untuk membandingkan hasil dengan algoritma yang lebih variatif.
4. Pada penelitian beriktunya, dapat mengganti studi kasus pendawalan mata kuliah otomatis. Karena pada penjadwalan mata kuliah otomatis tidak hanya dapat digunakan untuk Departemen Teknik Industri ITS, tetapi dapat digunakan pada Departemen yang lain maupun instansi pendidikan lain.

DAFTAR PUSTAKA

[1] G. Y. Hutama, "Optimasi Penjadwalan Ujian Otomatis Dengan Menggunakan Algoritma Greedy - Hill Climbing Hyperheuristic," 2017.

[2] Dian Kusumawardani, "Optimalisasi Penjadwalan Ujian Otomatis Dengan Menggunakan Algoritma Greedy Simulated Annealing Hyper-Heuristic," 2017.

[3] Putri Cahyaning Bwananesia, "Optimasi Penjadwalan Ujian Otomatis Dengan Menggunakan Algoritma Greedy – Late Acceptance – Hyper Heuristic," 2017.

[4] J. A. Soria-alcaraz, E. Özcan, J. Swan, G. Kendall, and M. Carpio, "Iterated local search using an add and delete hyper-heuristic for university course timetabling," *Appl. Soft Comput. J.*, vol. 40, pp. 581–593, 2016.

[5] K. Alaykiran and M. Hacibeyoglu, "Using Iterated Local Search to Solve the Course Timetabling Problem at Engineering Faculty of Necmettin Erbakan University," no. 12, pp. 40–43, 2016.

[6] J. S. Wijaya, W. Angraeni, and A. Muklason, "Advanced Traveler Information System: Optimasi Rencana Perjalanan Dengan Orienteering Problem Model Dan Iterative Local Search With Hill Climbing (Studi Kasus: Trayek Angkot Kota Surabaya)," 2017.

[7] E. L. Lawler, "Combinatorial Optimization : Networks and Matroids," *Comb. Optim. networks matroids*, pp. 1–374, 1976.

[8] A. Wren, "Scheduling, timetabling and rostering - A special relationship?," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 1996, vol. 1153, pp. 46–75.

[9] E. K. Burke *et al.*, "Applications to timetabling," in *In Handbook of Graph Theory*, 2004, pp. 445–474.

[10] E. K. Burke and S. Petrovic, "Recent research directions in automated timetabling," in *European Journal of*

Operational Research, 2002, vol. 140, no. 2, pp. 266–280.

[11] E. K. Burke, A. J. Eckersley, B. McCollum, S. Petrovic, and R. Qu, "Hybrid variable neighbourhood approaches to university exam timetabling," *Eur. J. Oper. Res.*, vol. 206, no. 1, pp. 46–53, 2010.

[12] T. B. Cooper and J. H. Kingston, "The complexity of timetable construction problems," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 1996, vol. 1153, pp. 283–295.

[13] K. Socha, J. Knowles, and M. Sampels, "A MAX-MIN Ant System for the University Course Timetabling Problem," pp. 1–13, 2002.

[14] Jason Brownlee PhD, "Iterated Local Search, Clever Algorithms: Nature-Inspired Programming Recipes." [Online]. Available: http://www.cleveralgorithms.com/nature-inspired/stochastic/iterated_local_search.html. [Accessed: 08-Mar-2018].

[15] H. Lourenço, O. Martin, and T. Stützle, "Iterated Local Search," *Handb. Metaheuristics SE - 11*, vol. 57, pp. 320–353, 2003.

[16] H. H. Hoos and T. St\:utzle, "Local Search Algorithms for SAT: An Empirical Evaluation," *J. Autom. Reason.*, vol. 24, no. 4, pp. 421–481, 2000.

[17] C. Cubillos, D. Cabrera-paniagua, E. Urra, C. Cubillos, and D. Cabrera-paniagua, "A Hyperheuristic for the Dial-a-Ride Problem with Time Windows A Hyperheuristic for the Dial-a-Ride Problem with Time Windows," no. January, 2015.

BIODATA PENULIS

Penulis mempunyai nama lengkap Nur Achmad Setiadi, dilahirkan di Rembang pada tanggal 31 Oktober 1994. Penulis menempuh pendidikan dasar di SDN Kepohagung. Setelah menamatkan sekolah dasar, penulis melanjutkan pendidikan tingkat menengah di SMP Negeri 1 Lasem dan SMA Negeri 1 Rembang. Pada tahun 2014, penulis diterima di Program Studi S1 Sistem Informasi, Fakultas Teknologi Informasi dan Komunikasi, Institut Teknologi Sepuluh Nopember Surabaya melalui jalur SNMPTN. Selama berkuliah di Departemen Sistem Informasi – FTIF – ITS, penulis aktif mengikuti kegiatan kepanitiaan, baik dalam lingkup jurusan, fakultas, maupun institut. Penulis juga pernah menjadi staf Departemen Hubungan Luar BEM FTIK ITS dan Wakil Ketua UKM PSHT di tahun kepenguruan 2015 – 2016. Pada tahun kepenguran 2016 - 2017, penulis menjabat sebagai Ketua Umum UKM Persaudaraan Setia Hati Terate ITS. Selain itu, penulis juga pernah mengikuti lomba tingkat nasional di Universitas Airlangga dan berhasil meraih juara dua dalam kategori tanding kelas J dalam kompetisi pencak silat. Penulis dapat dihubungi melalui email : nurachmadsetiadi@gmail.com.

Halaman ini sengaja dikosongkan.

LAMPIRAN A

Pada lampiran ini, merupakan datadari mata kuliah yang dilaksanakan pada semester gasal dan genap pada tahun ajaran 2017-2018. Terdapat dua tabel yang menunjukkan masing – masing semester.

Tabel A.1 Data Mata Kuliah Semester Gasal

No	Mata Kuliah	Kelas	SKS	Ruang
1	Analisa Keputusan	A	3	IE-108
2	Analisa Produktivitas	A	3	IE-106
3	Analisis dan Estimasi Biaya	A	3	IE-105
4	Analisis dan Estimasi Biaya	B	3	IE-109
5	Analisis dan Estimasi Biaya	C	3	Lab. MM
6	Analisis dan Estimasi Biaya	D	3	IE-106
7	Analisis dan Estimasi Biaya	Q	3	IE-108
8	Aplikasi Ergonomi Industri	A	3	IE-105
9	Computer Integrated Manufacturing	A	3	IE-303D
10	Data Mining	A	3	IE-106
11	Ergo Safety	A	3	IE-105
12	Ergonomi Industri	A	3	IE-103
13	Ergonomi Industri	B	3	IE-101
14	Ergonomi Industri	C	3	IE-102
15	Ergonomi Industri	D	3	IE-103
16	Ergonomi Industri	Q	3	IE-104
17	Faal dan Biomenikanika Kerja	A	3	IE-105
18	Makro Ergonomi	A	3	IE-102
19	Manajemen Distribusi	A	3	Lab. MM
20	Manajemen Lingkungan Industri	A	3	IE-108
21	Manajemen Pengetahuan	A	3	IE-109
22	Manajemen Persediaan dan Pengadaan	A	3	IE-105
23	Manajemen Resiko Korporat	A	3	IE-101
24	Matematika Optimasi	A	3	IE-101

A-2

No	Mata Kuliah	Kelas	SKS	Ruang
25	Matematika Optimasi	B	3	IE-102
26	Matematika Optimasi	C	3	IE-103
27	Matematika Optimasi	D	3	IE-104
28	Matematika Optimasi	Q	3	IE-105
29	Mekanika Teknik	A	2	IE-108
30	Menggambar Teknik	A	2	IE-101
31	Menggambar Teknik	B	2	IE-102
32	Menggambar Teknik	C	2	IE-106
33	Menggambar Teknik	D	2	Lab. MM
34	Menggambar Teknik	Q	2	IE-103
35	Metodologi Penyelesaian Masalah	A	2	IE-106
36	Metodologi Sistem Dinamik	A	3	IE-101
37	Otomasi Industri	A	3	IE-104
38	Otomasi Industri	B	3	IE-104
39	Otomasi Industri	C	3	IE-102
40	Otomasi Industri	D	3	IE-103
41	Otomasi Industri	Q	3	IE-108
42	Pemeliharaan dan Teknik Keandalan	A	3	IE-103
43	Pemeliharaan dan Teknik Keandalan	B	3	IE-104
44	Pemeliharaan dan Teknik Keandalan	C	3	IE-104
45	Pemeliharaan dan Teknik Keandalan	Q	3	IE-101
46	Penelitian Operasional II	A	3	IE-103
47	Penelitian Operasional II	B	3	IE-104
48	Penelitian Operasional II	C	3	IE-102
49	Penelitian Operasional II	Q	3	IE-106
50	Pengantar Ilmu Ekonomi	A	2	IE-109
51	Pengantar Ilmu Ekonomi	B	2	IE-105
52	Pengantar Ilmu Ekonomi	C	2	IE-108
53	Pengantar Ilmu Ekonomi	D	2	IE-104
54	Pengantar Ilmu Ekonomi	Q	2	IE-108
55	Pengantar Teknik dan Sistem Industri	A	2	IE-109

No	Mata Kuliah	Kelas	SKS	Ruang
56	Pengantar Teknik dan Sistem Industri	B	2	IE-106
57	Pengantar Teknik dan Sistem Industri	C	2	IE-105
58	Pengantar Teknik dan Sistem Industri	D	2	IE-106
59	Pengantar Teknik dan Sistem Industri	Q	2	IE-101
60	Pengetahuan Bahan Teknik	A	3	IE-108
61	Pengetahuan Bahan Teknik	B	3	IE-106
62	Pengetahuan Bahan Teknik	C	3	IE-102
63	Pengetahuan Bahan Teknik	D	3	IE-106
64	Pengetahuan Bahan Teknik	Q	3	IE-101
65	Perancangan dan Pengembangan Produk	A	3	IE-103
66	Perancangan dan Pengembangan Produk	B	3	IE-104
67	Perancangan dan Pengembangan Produk	C	3	IE-102
68	Perancangan dan Pengembangan Produk	Q	3	IE-105
69	Perancangan Fasilitas	A	3	IE-104
70	Perancangan Fasilitas	B	3	IE-108
71	Perancangan Fasilitas	C	3	Lab. MM
72	Perancangan Fasilitas	Q	3	IE-101
73	Perancangan Sistem Informasi Bisnis	A	3	IE-109
74	Perancangan Sistem Informasi Bisnis	B	3	Lab. MM
75	Perancangan Sistem Informasi Bisnis	C	3	IE-102
76	Perancangan Sistem Informasi Bisnis	Q	3	IE-108
77	Perencanaan dan Pengendalian Produksi	A1	2	IE-103
78	Perencanaan dan Pengendalian Produksi	A2	2	IE-109

No	Mata Kuliah	Kelas	SKS	Ruang
79	Perencanaan dan Pengendalian Produksi	B1	2	IE-101
80	Perencanaan dan Pengendalian Produksi	B2	2	IE-108
81	Perencanaan dan Pengendalian Produksi	C1	2	IE-102
82	Perencanaan dan Pengendalian Produksi	C2	2	IE-108
83	Perencanaan dan Pengendalian Produksi	Q1	2	IE-104
84	Perencanaan dan Pengendalian Produksi	Q2	2	IE-101
85	Perencanaan Industri II	A	3	IE-102
86	Perencanaan Industri II	B	3	IE-105
87	Perencanaan Industri II	C	3	IE-104
88	Perencanaan Industri II	D	3	IE-103
89	Perencanaan Industri II	Q	3	IE-109
90	Perencanaan Industri II	Z	3	Lab. MM
91	Permodelan Sistem Berbasis Agen	A	3	IE-102
92	Proses Manufaktur	A	3	Lab. MM
93	Proses Manufaktur	B	3	IE-101
94	Proses Manufaktur	Q	3	IE-109
95	Rekayasa Proses Bisnis	A	3	IE-105
96	Simulasi Diskrit Terapan	A	3	IE-108
97	Six Sigma	A	3	IE-105
98	Statistik Industri II	A	3	IE-101
99	Statistik Industri II	B	3	IE-103
100	Statistik Industri II	C	3	IE-104
101	Statistik Industri II	D	3	IE-102
102	Statistik Industri II	Q	3	IE-103
103	Supply Chain Management	A	3	IE-108
104	Teknik Pengendalian Kualitas	A	3	IE-101
105	Teknik Pengendalian Kualitas	B	3	IE-109
106	Teknik Pengendalian Kualitas	C	3	IE-103
107	Teknik Pengendalian Kualitas	Q	3	IE-102
108	Teori Permainan	A	3	IE-105

No	Mata Kuliah	Kelas	SKS	Ruang
109	Termodinamika	A	2	IE-103
110	Termodinamika	B	2	IE-104
111	Termodinamika	C	2	IE-105

Tabel A.2 Data Mata Kuliah Semester Genap

No	Mata Kuliah	Kelas	SKS	Ruang
1	Analisa Kemampuan Proses	3	A	IE-102
2	Analisa Keputusan	3	A	IE-105
3	Analisa Produktivitas	3	A	IE-104
4	Concurrent Engineering	3	A	IE-106
5	Ekologi Industri	2	A	IE-109
6	Ekologi Industri	2	B	IE-103
7	Ekologi Industri	2	C	IE-101
8	Ekologi Industri	2	D	IE-102
9	Ekologi Industri	2	Q	IE-109
10	Ekonomi Teknik	3	A	IE-105
11	Ekonomi Teknik	3	B	IE-109
12	Ekonomi Teknik	3	C	IE-108
13	Ekonomi Teknik	3	D	IE-108
14	Ekonomi Teknik	3	Q	IE-109
15	Enterprise Resource Planning	3	A	IE-104
16	Ergo Safety	3	A	IE-105
17	Ergonomi Kognitif	3	A	IE-103
18	Human Reliability	3	A	IE-101
19	Keputusan Multi Kriteria (MCDM)	3	A	IE-106
20	Manajemen Keuangan	3	A	IE-108
21	Manajemen Kinerja	3	A	IE-103
22	Manajemen Kualitas	3	A	IE-104
23	Manajemen Logistik	3	A	IE-101
24	Manajemen Logistik	3	B	IE-102
25	Manajemen Logistik	3	C	IE-103
26	Manajemen Logistik	3	Q	IE-104
27	Manajemen Organisasi dan Sumber Daya Manusia	3	A	IE-101
28	Manajemen Organisasi dan Sumber Daya Manusia	3	B	IE-102

No	Mata Kuliah	Kelas	SKS	Ruang
29	Manajemen Organisasi dan Sumber Daya Manusia	3	C	IE-101
30	Manajemen Organisasi dan Sumber Daya Manusia	3	Q	IE-102
31	Manajemen Persediaan dan Pengadaan	3	A	IE-105
32	Manajemen Proyek	2	A	IE-101
33	Manajemen Proyek	2	B	IE-101
34	Manajemen Proyek	2	C	IE-105
35	Manajemen Proyek	2	D	IE-102
36	Manajemen Proyek	2	Q	IE-109
37	Manajemen Transportasi Udara	3	A	Lab. MM
38	Mekanika Teknik	2	A	IE-101
39	Mekanika Teknik	2	B	IE-102
40	Mekanika Teknik	2	C	IE-103
41	Mekanika Teknik	2	D	IE-105
42	Mekanika Teknik	2	Q	IE-104
43	Metodologi Penyelesaian Masalah	2	A	IE-105
44	Metodologi Penyelesaian Masalah	2	B	IE-108
45	Metodologi Penyelesaian Masalah	2	C	IE-106
46	Metodologi Penyelesaian Masalah	2	Q	IE-109
47	Metodologi Sistem Dinamik	3	A	IE-103
48	Optimasi Metaheuristik	3	A	IE-108
49	Penelitian Operasional I	3	A	IE-104
50	Penelitian Operasional I	3	B	IE-101
51	Penelitian Operasional I	3	C	IE-102
52	Penelitian Operasional I	3	D	IE-103
53	Penelitian Operasional I	3	Q	IE-103
54	Perancangan Metode dan Stasiun Kerja	3	A	IE-105
55	Perencanaan Industri I	3	A	IE-101
56	Perencanaan Industri I	3	B	IE-102
57	Perencanaan Industri I	3	C	IE-104
58	Perencanaan Industri I	3	D	IE-105

No	Mata Kuliah	Kelas	SKS	Ruang
59	Perencanaan Industri I	3	Q	IE-103
60	Permodelan Sistem	3	A	IE-105
61	Permodelan Sistem	3	B	IE-103
62	Permodelan Sistem	3	C	IE-104
63	Permodelan Sistem	3	Q	IE-104
64	Permodelan Sistem Berbasis Agen	3	A	Lab. MM
65	Proses Manufaktur	3	A	IE-108
66	Proses Manufaktur	3	B	IE-109
67	Proses Manufaktur	3	Q	IE-109
68	Simulasi Sistem Industri	3	A	IE-102
69	Simulasi Sistem Industri	3	B	IE-104
70	Simulasi Sistem Industri	3	C	IE-103
71	Simulasi Sistem Industri	3	Q	IE-101
72	Sistem Manufaktur	3	A	IE-101
73	Sistem Manufaktur	3	B	IE-106
74	Sistem Manufaktur	3	C	IE-102
75	Sistem Manufaktur	3	D	IE-105
76	Sistem Manufaktur	3	Q	IE-109
77	Six Sigma	3	A	IE-104
78	Statistik Industri I	3	A	IE-104
79	Statistik Industri I	3	B	IE-103
80	Statistik Industri I	3	C	IE-104
81	Statistik Industri I	3	D	IE-105
82	Statistik Industri I	3	Q	IE-105
83	Supply Chain Management	3	A	IE-108
84	Sustainable Manufacturing	3	A	IE-106
85	Teknik Tata Cara dan Pengukuran Kerja	3	A	IE-101
86	Teknik Tata Cara dan Pengukuran Kerja	3	B	IE-101
87	Teknik Tata Cara dan Pengukuran Kerja	3	C	IE-102
88	Teknik Tata Cara dan Pengukuran Kerja	3	D	IE-102
89	Teknik Tata Cara dan Pengukuran Kerja	3	Q	IE-109
90	Teori Permainan	3	A	IE-104

A-8

No	Mata Kuliah	Kelas	SKS	Ruang
91	Termodinamika	2	A	IE-101
92	Termodinamika	2	B	IE-102
93	Termodinamika	2	Q	IE-103

LAMPIRAN B

Pada lampiran ini, berisi daftar kode event yang ada pada semester gasal dan genap ada tahun ajaran 2017-2018. Pada semester gasal jumlah event yang ada adalah 111 dan pada semester genap adalah 93.

Tabel B.1 Daftar kode event semester gasal

No	Mata Kuliah	Kelas	Kode Event
1	Analisa Keputusan	A	001
2	Analisa Produktivitas	A	002
3	Analisis dan Estimasi Biaya	A	003
4	Analisis dan Estimasi Biaya	B	004
5	Analisis dan Estimasi Biaya	C	005
6	Analisis dan Estimasi Biaya	D	006
7	Analisis dan Estimasi Biaya	Q	007
8	Aplikasi Ergonomi Industri	A	008
9	Computer Integrated Manufacturing	A	009
10	Data Mining	A	010
11	Ergo Safety	A	011
12	Ergonomi Industri	A	012
13	Ergonomi Industri	B	013
14	Ergonomi Industri	C	014
15	Ergonomi Industri	D	015
16	Ergonomi Industri	Q	016
17	Faal dan Biomenikanika Kerja	A	017
18	Makro Ergonomi	A	018
19	Manajemen Distribusi	A	019
20	Manajemen Lingkungan Industri	A	020
21	Manajemen Pengetahuan	A	021
22	Manajemen Persediaan dan Pengadaan	A	022
23	Manajemen Resiko Korporat	A	023
24	Matematika Optimasi	A	024
25	Matematika Optimasi	B	025
26	Matematika Optimasi	C	026
27	Matematika Optimasi	D	027

B-2

No	Mata Kuliah	Kelas	Kode Event
28	Matematika Optimasi	Q	028
29	Mekanika Teknik	A	029
30	Menggambar Teknik	A	030
31	Menggambar Teknik	B	031
32	Menggambar Teknik	C	032
33	Menggambar Teknik	D	033
34	Menggambar Teknik	Q	034
35	Metodologi Penyelesaian Masalah	A	035
36	Metodologi Sistem Dinamik	A	036
37	Otomasi Industri	A	037
38	Otomasi Industri	B	038
39	Otomasi Industri	C	039
40	Otomasi Industri	D	040
41	Otomasi Industri	Q	041
42	Pemeliharaan dan Teknik Keandalan	A	042
43	Pemeliharaan dan Teknik Keandalan	B	043
44	Pemeliharaan dan Teknik Keandalan	C	044
45	Pemeliharaan dan Teknik Keandalan	Q	045
46	Penelitian Operasional II	A	046
47	Penelitian Operasional II	B	047
48	Penelitian Operasional II	C	048
49	Penelitian Operasional II	Q	049
50	Pengantar Ilmu Ekonomi	A	050
51	Pengantar Ilmu Ekonomi	B	051
52	Pengantar Ilmu Ekonomi	C	052
53	Pengantar Ilmu Ekonomi	D	053
54	Pengantar Ilmu Ekonomi	Q	054
55	Pengantar Teknik dan Sistem Industri	A	055
56	Pengantar Teknik dan Sistem Industri	B	056
57	Pengantar Teknik dan Sistem Industri	C	057
58	Pengantar Teknik dan Sistem Industri	D	058
59	Pengantar Teknik dan Sistem Industri	Q	059

No	Mata Kuliah	Kelas	Kode Event
60	Pengetahuan Bahan Teknik	A	060
61	Pengetahuan Bahan Teknik	B	061
62	Pengetahuan Bahan Teknik	C	062
63	Pengetahuan Bahan Teknik	D	063
64	Pengetahuan Bahan Teknik	Q	064
65	Perancangan dan Pengembangan Produk	A	065
66	Perancangan dan Pengembangan Produk	B	066
67	Perancangan dan Pengembangan Produk	C	067
68	Perancangan dan Pengembangan Produk	Q	068
69	Perancangan Fasilitas	A	069
70	Perancangan Fasilitas	B	070
71	Perancangan Fasilitas	C	071
72	Perancangan Fasilitas	Q	072
73	Perancangan Sistem Informasi Bisnis	A	073
74	Perancangan Sistem Informasi Bisnis	B	074
75	Perancangan Sistem Informasi Bisnis	C	075
76	Perancangan Sistem Informasi Bisnis	Q	076
77	Perencanaan dan Pengendalian Produksi	A1	077
78	Perencanaan dan Pengendalian Produksi	A2	078
79	Perencanaan dan Pengendalian Produksi	B1	079
80	Perencanaan dan Pengendalian Produksi	B2	080
81	Perencanaan dan Pengendalian Produksi	C1	081
82	Perencanaan dan Pengendalian Produksi	C2	082
83	Perencanaan dan Pengendalian Produksi	Q1	083
84	Perencanaan dan Pengendalian Produksi	Q2	084
85	Perencanaan Industri II	A	085

B-4

No	Mata Kuliah	Kelas	Kode Event
86	Perencanaan Industri II	B	086
87	Perencanaan Industri II	C	087
88	Perencanaan Industri II	D	088
89	Perencanaan Industri II	Q	089
90	Perencanaan Industri II	Z	090
91	Permodelan Sistem Berbasis Agen	A	091
92	Proses Manufaktur	A	092
93	Proses Manufaktur	B	093
94	Proses Manufaktur	Q	094
95	Rekayasa Proses Bisnis	A	095
96	Simulasi Diskrit Terapan	A	096
97	Six Sigma	A	097
98	Statistik Industri II	A	098
99	Statistik Industri II	B	099
100	Statistik Industri II	C	100
101	Statistik Industri II	D	101
102	Statistik Industri II	Q	102
103	Supply Chain Management	A	103
104	Teknik Pengendalian Kualitas	A	104
105	Teknik Pengendalian Kualitas	B	105
106	Teknik Pengendalian Kualitas	C	106
107	Teknik Pengendalian Kualitas	Q	107
108	Teori Permainan	A	108
109	Termodinamika	A	109
110	Termodinamika	B	110
111	Termodinamika	C	111

Tabel B.2 Daftar kode event semester genap

No	Mata Kuliah	Kelas	Kode Event
1	Analisa Kemampuan Proses	A	001
2	Analisa Keputusan	A	002
3	Analisa Produktivitas	A	003
4	Concurrent Engineering	A	004
5	Ekologi Industri	A	005
6	Ekologi Industri	B	006
7	Ekologi Industri	C	007
8	Ekologi Industri	D	008
9	Ekologi Industri	Q	009
10	Ekonomi Teknik	A	010
11	Ekonomi Teknik	B	011
12	Ekonomi Teknik	C	012
13	Ekonomi Teknik	D	013
14	Ekonomi Teknik	Q	014
15	Enterprise Resource Planning	A	015
16	Ergo Safety	A	016
17	Ergonomi Kognitif	A	017
18	Human Reliability	A	018
19	Keputusan Multi Kriteria (MCDM)	A	019
20	Manajemen Keuangan	A	020
21	Manajemen Kinerja	A	021
22	Manajemen Kualitas	A	022
23	Manajemen Logistik	A	023
24	Manajemen Logistik	B	024
25	Manajemen Logistik	C	025
26	Manajemen Logistik	Q	026
27	Manajemen Organisasi dan Sumber Daya Manusia	A	027
28	Manajemen Organisasi dan Sumber Daya Manusia	B	028
29	Manajemen Organisasi dan Sumber Daya Manusia	C	029
30	Manajemen Organisasi dan Sumber Daya Manusia	Q	030
31	Manajemen Persediaan dan Pengadaan	A	031
32	Manajemen Proyek	A	032

No	Mata Kuliah	Kelas	Kode Event
33	Manajemen Proyek	B	033
34	Manajemen Proyek	C	034
35	Manajemen Proyek	D	035
36	Manajemen Proyek	Q	036
37	Manajemen Transportasi Udara	A	037
38	Mekanika Teknik	A	038
39	Mekanika Teknik	B	039
40	Mekanika Teknik	C	040
41	Mekanika Teknik	D	041
42	Mekanika Teknik	Q	042
43	Metodologi Penyelesaian Masalah	A	043
44	Metodologi Penyelesaian Masalah	B	044
45	Metodologi Penyelesaian Masalah	C	045
46	Metodologi Penyelesaian Masalah	Q	046
47	Metodologi Sistem Dinamik	A	047
48	Optimasi Metaheuristik	A	048
49	Penelitian Operasional I	A	049
50	Penelitian Operasional I	B	050
51	Penelitian Operasional I	C	051
52	Penelitian Operasional I	D	052
53	Penelitian Operasional I	Q	053
54	Perancangan Metode dan Stasiun Kerja	A	054
55	Perencanaan Industri I	A	055
56	Perencanaan Industri I	B	056
57	Perencanaan Industri I	C	057
58	Perencanaan Industri I	D	058
59	Perencanaan Industri I	Q	059
60	Permodelan Sistem	A	060
61	Permodelan Sistem	B	061
62	Permodelan Sistem	C	062
63	Permodelan Sistem	Q	063
64	Permodelan Sistem Berbasis Agen	A	064
65	Proses Manufaktur	A	065
66	Proses Manufaktur	B	066
67	Proses Manufaktur	Q	067
68	Simulasi Sistem Industri	A	068
69	Simulasi Sistem Industri	B	069

No	Mata Kuliah	Kelas	Kode Event
70	Simulasi Sistem Industri	C	070
71	Simulasi Sistem Industri	Q	071
72	Sistem Manufaktur	A	072
73	Sistem Manufaktur	B	073
74	Sistem Manufaktur	C	074
75	Sistem Manufaktur	D	075
76	Sistem Manufaktur	Q	076
77	Six Sigma	A	077
78	Statistik Industri I	A	078
79	Statistik Industri I	B	079
80	Statistik Industri I	C	080
81	Statistik Industri I	D	081
82	Statistik Industri I	Q	082
83	Supply Chain Management	A	083
84	Sustainable Manufacturing	A	084
85	Teknik Tata Cara dan Pengukuran Kerja	A	085
86	Teknik Tata Cara dan Pengukuran Kerja	B	086
87	Teknik Tata Cara dan Pengukuran Kerja	C	087
88	Teknik Tata Cara dan Pengukuran Kerja	D	088
89	Teknik Tata Cara dan Pengukuran Kerja	Q	089
90	Teori Permainan	A	090
91	Termodinamika	A	091
92	Termodinamika	B	092
93	Termodinamika	Q	093