TUGAS AKHIR – KS 141501

OPTIMASI PENJADWALAN STAF DENGAN MENGGUNAKAN ALGORITMA SELF-ADAPTIVE LEARNING HYPER-HEURISTIC (STUDI KASUS: RSIA KENDANGSARI SURABAYA).

STAFF SCHEDULING OPTIMIZATION USING SELF-ADAPTIVE ALGORITHM HYPER-HEURISTIC (CASE STUDY: KENDANGSARI MOTHER AND CHILD HOSPITAL IN SURABAYA).

FACHRUR ZAFFRINDA PRAYOGO
NRP 05211440000173

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

JURUSAN SISTEM INFORMASI
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

TUGAS AKHIR – KS141501

OPTIMASI PENJADWALAN STAF DENGAN MENGGUNAKAN ALGORITMA SELF-ADAPTIVE LEARNING HYPER-HEURISTIC (STUDI KASUS: RSIA KENDANGSARI SURABAYA).

FACHRUR ZAFFRINDA PRAYOGO
NRP 05211440000173

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

JURUSAN SISTEM INFORMASI
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

FINAL PROJECT – KS 141501

STAFF SCHEDULING OPTIMIZATION USING SELF-ADAPTIVE LEARNING ALGORITHM HYPER-HEURISTIC (CASE STUDY: KENDANGSARI'S MOTHER AND CHILD HOSPITAL IN SURABAYA).

FACHRUR ZAFFRINDA PRAYOGO
NRP 05211440000173

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

JURUSAN SISTEM INFORMASI
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

LEMBAR PENGESAHAN

OPTIMASI PENJADWALAN STAF DENGAN MENGGUNAKAN ALGORITMA SELF-ADAPTIVE LEARNING HYPER-HEURISTIC (STUDI KASUS: RSIA KENDANGSARI SURABAYA).

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:

FACHRUR ZAFFRINDA PRAYOGO
5214100173

Surabaya, 16 Januari 2018

Plh Kepala
Departemen Sistem Informasi

Edwin Riksakomara, S.Kom., M.T
NIP.196907252003121001

LEMBAR PERSETUJUAN

OPTIMASI PENJADWALAN STAF DENGAN MENGGUNAKAN ALGORITMA SELF-ADAPTIVE LEARNING HYPER-HEURISTIC (STUDI KASUS: RSIA KENDANGSARI SURABAYA).

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 :

FACHRUR ZAFFRINDA PRAYOGO
NRP. 5214100173

Disetujui Tim Penguji :Tanggal Ujian : 12 Januari 2018
Periode Wisuda : Maret 2018

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

Edwin Riksakomara, S.Kom., MT. **(Penguji I)**

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

OPTIMASI PENJADWALAN STAF DENGAN MENGGUNAKAN ALGORITMA SELF-ADAPTIVE LEARNING HYPER-HEURISTIC (STUDI KASUS: RSIA KENDANGSARI SURABAYA)

Nama Mahasiswa : Fachrur Zaffrinda Prayogo
NRP : 05211440000173
Jurusan : Sistem Informasi FTIK-ITS
Pembimbing : Ahmad Muklason, Ph.D.

ABSTRAK

Rumah Sakit Ibu dan Anak Kendangsari adalah salah satu rumah sakit yang berada di kota Surabaya. Rumah sakit adalah organisasi yang bergerak dibidang kesehatan dan harus memiliki layanan 7x24 jam seminggu sehingga salah satu masalah yang terjadi adalah kurang optimalnya tenaga kerja dalam memenuhi permintaan yang sangat besar sehingga pihak manajemen rumah sakit harus memiliki manajemen yang baik terhadap sumber daya manusia yang ada salah satu bagian terpenting adalah perawat. Permasalahan yang terjadi adalah kurang optimalnya jadwal perawat masalah ini disebut Nurse Rostering Problem. Dalam membuat sebuah jadwal yang baik diperlukan Batasan-batasan yang jelas dan harus dipertimbangkan seperti regulasi rumah sakit dimana Batasan tersebut dibagi menjadi 2(dua) yaitu hard constraint dan soft constraint.

Untuk menyelesaikan masalah yang ada maka dilakukan pembuatan jadwal yang optimal, pada penelitian ini metode yang digunakan adalah algoritma self-adaptive-learning hyper-heuristic dari metode algoritma tersebut akan dicari bagaimanakah solusi penjadwalan yang optimal sesuai dengan batasan-batasan yang ada serta membuat penjadwalan secara

otomatis dan hasil penjadwalan akan dibandingkan dengan jadwal yang sudah ada. Untuk tolak ukur sebuah jadwal sudah optimal atau belum diukur dari nilai Jain Fairness Index yang *memiliki rentang nilai 0-1, maka untuk mendapatkan hasil yang optimal juga dilakukan pengujian dengan membandingkan algoritma Self adaptive learning dengan algortima Hill Climbing.*

Berdasarkan hasil pengujian dua algoritma yaitu Self Adaptive dan Hill Climbing ditemukan hasil bahwa pada 6 divisi setelah dilakukan optimasi dengan menggunakan algoritma tersebut memiliki nilai JFI untuk divisi Farmasi hasil sebelum optimasi dari 0.92 menjadi 0.94 setelah optimasi , divisi IGD sebesar 0.91 menjadi 0.94 , divisi Bayi & NICU dari 0.89 menjadi 0.93, divisi Gizi dari 0.85 menjadi 0.83, divisi OK dari 0.80 menjadi 0.96 ,dan divisi SIM & RM dari 0.90 menjadi 0.97 .

Kata kunci : Optimasi, Hyper-heuristic, Self-Adaptive, Nurse Rostering Problem, Hill Climbing, Jain Fairness Index

STAFF SCHEDULING OPTIMIZATION USING SELF-ADAPTIVE LEARNING ALGORITHM HYPER-HEURISTIC (CASE STUDY: KENDANGSARI'S MOTHER AND CHILD HOSPITAL IN SURABAYA).

Student Name : Fachrur Zaffrinda Prayogo
NRP : 05211440000173
Departmen : Sistem Informasi FTIf-ITS
Supervisor : Ahmad Muklason, Ph.D.

ABSTRACT

Kendangsari Mother and Child Hospital are one of the hospitals located in Surabaya city. A hospital is an organization engaged in the health and must have service 7x24 hours a week because one of the less optimal workforces in meeting the huge demand. The management of the hospital should have good management of human resources, the important part is nurses The problem that occurs less optimal schedule of this problem is called Nurse Rostering Problem. In making a good schedule it is necessary that the constraints are clear and should be like the hospital regulations where the constraints are divided into 2 (two) namely the hard constraints and soft constraints.

To solve the existing problems then made the optimal schedule, in this study the method used is the algorithm self-adaptive-learning hyper-heuristic algorithm that will be searched how optimal scheduling solutions in accordance with the existing constraints and make an automatic scheduling and the result of the scheduling will be compared to an existing schedule. For the benchmark of a schedule is optimal or not measured from the value of Jain Fairness index that has the range from 0 to 1, then for optimal results are also tested by comparing the algorithm Self-adaptive learning with Hill Climbing algorithm. According to the result of the test from 2 algorithms which is Self Adaptive and Hill Climbing researcher find a result of the JFI for Pharmacy Division before the optimization is 0.92 became 0.94 after Optimization, Emergency Division is 0.91

became 0.94, Baby & NICU Division from 0.89 became 0.93, Nutrition Division from 0.85 became 0.83, OK Division from 0.80 became 0.96 , Administration and Medical record from 0.90 became 0.97
Keywords: Optimization, Hyper-heuristic, Self-Adaptive, Problem Nurse

KATA PENGANTAR

Alhamdulillah atas karunia, rahmat, barakah, dan jalan yang telah diberikan Allah SWT selama ini sehingga penulis mendapatkan kelancaran dalam menyelesaikan tugas akhir dengan judul:

OPTIMASI PENJADWALAN STAF DENGAN MENGGUNAKAN ALGORITMA SELF-ADAPTIVE LEARNING HYPER-HEURISTIC (STUDI KASUS: RSIA KENDANGSARI, SURABAYA).

Terima kasih atas pihak-pihak yang telah mendukung, memberikan saran, motivasi, semangat, dan bantuan baik materi maupun spiritual demi tercapainya tujuan pembuatan tugas akhir ini. Secara khusus penulis akan menyampaikan ucapan terima kasih yang sedalam-dalamnya kepada:

1. Orang tua penulis, Bapak Ir. Zaenul Arief dan Ibu Indrati Sulistyo, S.H. Kakak Rizkyana Zaffrindra Putri dan Adik Rachma Zaffindra Amalia yang tercinta.
2. Bapak Ahmad Muklason, S.Kom., M.Sc., Ph.D. selaku dosen pembimbing yang meluangkan waktu, memberikan ilmu, petunjuk, dan motivasi untuk kelancaran Tugas Akhir ini.
3. Bapak Edwin Riksakormara, S.Kom., MT dan Bapak Faizal Mahananto, S.Kom., M.Eng., Ph.D. selaku dosen penguji yang telah memberikan masukan untuk perbaikan tugas akhir ini.
4. Ibu Wiwik Anggraeni, S.Si., M.Kom sebagai dosen wali yang telah membantu dalam konsultasi selama perkuliahan sampai tugas akhir.
5. Seluruh civitas akademika, termasuk dosen, pada Jurusan Sistem Informasi ITS yang telah menjadi bagian hidup selama kuliah dan yang telah memberikan ilmu yang sangat berharga bagi penulis.

6. Rekan-rekan OSIRIS yang telah berjuang bersama dalam menjalani perkuliahan di Jurusan Sistem Informasi ITS.Terima kasih atas segala dukungan yang diberikan.
7. Teman-teman Warung Squad Adhen, Adit, Alden, Boy, Bram, Dito, Fadel, Fandhi, Fata, Leon, Nody, Obik, Rafi, Rama, Rysma, Satria.
8. Kepada Ibu Sylvy selaku manajer bidang SDM RSIA Kendangsari MERR Surabaya yang telah membantu dalam menyediakan data untuk tugas akhir
9. Teman-teman seperjuangan lab RDIB, SE, ADDI , Fata, Nita, Zuli, Opor, Fia, Patty, Rara, Septy, Alden, mbak Oryza yang membantu penulis dan memberikan motivasi untuk menyelesaikan tugas akhir
10. Berbagai pihak yang membantu dalam penyusunan Tugas Akhir ini dan belum dapat 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.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR KODE PROGRAM

DAFTAR TABEL

BAB I
PENDAHULUAN

Pada bab pendahuluan ini akan membahas terkait latar belakang masalah, perumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, dan relevansi terhadap pengerjaan tugas akhir.

1.1. Latar Belakang

RSIA Kendangsari Surabaya adalah Rumah Sakit Ibu dan Anak Surabaya yang dirancang unik berfokus untuk melayani kebutuhan ibu dan anak. Dalam menjalankan fungsinya RSIA Kendangsari Surabaya memberikan pelayanan kesehatan yang paripurna untuk wanita dan anak. Pelayanan kesehatan diberikan secara prima dan komprehensif bagi pasien,keluarga pasien dan provider baik perusahaan maupun asuransi.

Dalam organisasi tenaga kerja merepresentasikan sumber daya vital yang biasanya mahal dan langka. Dikarenakan biaya personel menggunakan sebagian besar biaya operasional, sangat penting untuk mengatur tenaga kerja yang ada dengan seefisien mungkin, terutama di bidang kesehatan dimana organisasi tersebut banyak yang mengalami kekurangan perawat atau pegawai lain.

Salah satu hal yang berpengaruh daam pelayanan rumah sakit adalah perawat, Karena perawat yang paling sering berinteraksi dengan pasien, perawat adalah bagian penting dari proses bisnis rumah sakit maka dari itu untuk mencapai tujuan secara optimal dibutuhkan manajemen yang baik terhadap sumber daya manusia di rumah sakit terutama perawat [1].

Dalam menjalankan proses bisnis rumah sakit, dibutuhkan pelayanan yang baik selama 24 jam dalam sehari dan tujuh hari dalam seminggu, maka penjadwalan dari perawat harus dilakukan secara optimal sehingga para perawat dapat menjalankan tugasnya dengan baik dan para perawat

mendapatkan jadwal *shift* yang adil dan sesuai dengan kebutuhan.

Beberapa masalah yang terjadi dalam penjadwalan perawat ini diantaranya adalah banyak para perawat yang mendapatkan jadwal kurang sesuai dengan kemampuan mereka seperti terlalu sering mendapatkan *shift* malam hari dan juga di akhir pekan sehingga pelayanan rumah sakit bisa menjadi menurun dan meningkatkan kemungkinan terjadinya pasien yang tidak terlayani dengan baik. Maka dari itu dibutuhkan penjadwalan yang optimal untuk perawat sehingga para perawat mendapatkan jadwal yang adil dan merata.

Nurse Rostering Problem (NRP) merupakan permasalahan umum yang sering dihadapi dalam melakukan penjadwalan perawat Karena pada NRP berurusan dengan pekerjaan, waktu libur, dan pengaturan shift untuk perawat yang bekerja di rumah sakit. Penjadwalan yang baik harus dapat memaksimalkan pemenuhan dari Batasan yang ada seperti regulasi pemeriintah, praktek kerja dan keinginan pribadi dari perawat [2]. Batasan atau *Constraint* yan gada di dalam NRP biasanya terbagi menjadi dua, yaitu *Hard Constraint* dan *Soft Constraint. Hard Constraint* adalah Batasan yang harus dipenuhi contohnya, Regulasi Rumah Sakit, lalu *Soft Constraint* merupakan Batasan yang tidak harus dipenuhi namun diusahakan untuk di penuhi contohnya, keinginan pribadi dari perawat. Ada banyak Teknik yang biasa digunakan untuk menyelesaikan masalah ini beberapa diantaranya ada *cylical scheduling, mathematical programming,* Penjadwalan heuristic, dan *Artificial Intelligence* [3].

Beberapa penelitian terkait ada yang menggunakan pendekatan yang baru bernama *hyper-heuristic* dengan menggunakan algoritma *self-adaptive learning* pendekatan

inilah yang akan diuji pada penelitian ini, karena pada penelitian sebelumnya telah diuji dan memiliki hasil yang baik pada masalah penjadwalan atau *Scheduling*, dan Metode ini sudah pernah digunakan pada kasus optimasi penjadwalan ujian pada referensi [4].

Dengan berbasis penelitian sebelumnya. Maka dalam tugas akhir ini, Penjadwalan perawat di Rumah Sakit Kendangsari Surabaya akan dioptimasukan dengan menggunakan metode algoritma *self-adaptive learning hyper heuristic.*

1.2. Perumusan Masalah

Berdasarkan latar belakang yang telah diuraikan, didapatkan perumusan permasalahan yaitu :

1. Bagaimana model matematis untuk mengatasi permasalahan penjadwalan perawat di Rumah Sakit Ibu dan Anak Kendangsari, Surabaya?
2. Bagaimana hasil dari penjadwalan otomatis dengan menggunakan algoritma *Self-Adaptive Learning* dibandingkan dengan jadwal perawat yang sudah diterapkan di Rumah Sakit Ibu dan Anak Kendangsari, Surabaya?

1.3. Batasan Pengerjaan Tugas Akhir

Batasan masalah dari penelitian ini adalah :

a. Studi kasus yang digunakan pada tugas akhir ini adalah permasalahan optimasi penjadwalan perawat di Rumah Sakit Kendangsari, Surabaya
b. Data yang akan digunakan adalah data jadwal *shift* perawat yang sudah ada, ketentuan atau regulasi rumah sakit terkait perawat, kebutuhan rumah sakit akan jumlah perawat yang akan dialokasikan.

c. Optimalitas dari jadwal dihitung berdasarkan hari libur yang diterima oleh masing-masing staff didapatkan dengan memberikan bobot pada hari libur antara minggu, sabtu, dan hari kerja.

1.4. Tujuan Tugas Akhir

Berdasarkan rumusan masalah tujuan dilakukannya tugas akhir ini adalah :

1. Pembuatan model matematis untuk mengatasi permasalahan penjadwalan perawat di Rumah Sakit Ibu dan Anak Kendangsari, Surabaya
2. Membuat penjadwalan otomatis perawat dengan menggunakan Algoritma *Self-Adaptive Learning* dan membandingkan dengan jadwal yang sudah diterapkan di Rumah Sakit Ibu dan Anak Kendangsari Surabaya.

1.5. Manfaat Tugas Akhir

Hasil dari tugas akhir ini diharapkan dapat digunakan untuk melakukan penjadwalan perawat secara optimal dan adil bagi perawat sehingga pihak rumah sakit dapat mengalokasikan tenaga kerja secara optimal.

1.6. Relevansi

Laboratorium Rekayasa Data dan Intelegensia Bisnis (RDIB) memiliki beberapa topik penelitian diantaranya adalah peramalan, optimasi, dan data mining. Topik yang saya ambil untuk tugas akhir ini adalah tentang Optimasi. Mata kuliah yang bersangkutan dengan topik ini adalah mata kuliah Riset Operasi.

Tugas ini layak untuk dijadikan sebagai tugas akhir karena tugas ini dapat membantu pihak rumah sakit dalam melakukan penjadwalan perawat sehingga dapat berpengaruh terhadap

tenaga kerja rumah sakit sehingga rumah sakit dapat meningkatkan kualitas pelayanan mereka.

BAB II
TINJAUAN PUSTAKA

Bab ini berisikan tinjauan pustaka yang akan digunakan dalam penelitian tugas akhir ini, mencakup studi sebelumnya, dasar teori dan metode yang digunakan.

2.1. Studi Sebelumnya

Dalam pengerjaan tugas akhir ini terdapat beberapa penelitian yang terkait untuk bisa dijadikan sebagai bahan studi literatur untuk menyelesaikan tugas akhir ini. Beberapa penelitian sebelumnya yang dijadikan acuan dalam pengerjaan tugas akhir disajikan dalam tabel berikut.

Tabel 2.1 Studi Literatur

Judul Paper	Nurse Rostering: Models And Algorithms For Theory, Practice And Integration With Other Problems
Penulis; Tahun	Pieter Smet ; 2015
Deskripsi Umum Penelitian	Penelitian ini menjabarkan tentang bagaimana dasar ilmu dan *basic theory* dari *Nurse Rostering* yaitu penjadwalan perawat yang bisa dilakukan di suatu rumah sakit atau organisasi kesehatan. Dalam penelitiannya juga dijelaskan bagaimana melakukan pemodelan matematis dari penjadwalan perawat, lalu dari model tersebut penelitian ini juga menjabarkan integrasi dari model ke masalah lain yang bisa terjadi dengan melakukan eksperiment terhadap beberapa algoritma dan menerapkannya berdasarkan model yang telah ada.

Keterkaitan Penelitian	Penerapan algoritma yang bisa didapatkan dari paper tersebut bisa digunakan sebagai referensi, selain itu dalam penelitian tersebut juga dijelaskan bagaimana membuat model matematis dari *Nurse Rostering* yang baik agar dapat membuat model yang sesuai dengan kebutuhan.

Tabel 2.2 Studi Literatur 2

Judul Paper	HYPER-HEURISTICS AND FAIRNESS IN EXAMINATION TIMETABLING PROBLEMS
Penulis; Tahun	Ahmad Muklason; 2017
Deskripsi Umum Penelitian	Penelitian tersebut membahas tentang bagaimana algoritma dengan tipe *hyper-heuristic* diterapkan dalam penjadwalan serta membahas masalah yang mngkin terjadi pada saat melakukan pembuatan *timetabling*, Serta menjadikan *hyper-heuristic* sebagai pendekatan baru terhadap formulasi masalah yang ada
Keterkaitan Penelitian	Penjabaran tentang *hyper-heuristic* yang ada di dalam penelitian tersebut dapat dijadikan referencsi dan pengetahuan terhadap penulis, dan bisa menerapkan algoritma optimasi dengan pendekatan *hyper-heuristic* yang dijelaskan dalam penelitian tersebut. Selain itu, dalam penelitian ini juga dijelaska tentang menggunakan algoritma *Self-Adaptive Learning* dengan pendekatan *hyper-heuristic* sehingga dapat dijadikan referensi untuk penelitian tugas akhir ini.

Tabel 2.3 Studi Literatur 3

Judul Paper	*Optimasi Penjadwalan Perawat Menggunakan Gabungan Integer Linear Programming Dan Variable Neighborhood Search*
Penulis; Tahun	Amir, Muhammad Asrar; 2017
Deskripsi Umum Penelitian	Pada penelitian ini membahas tentang bagaimana membuat penjadwalan perawat yang optimal dengan menggunakan dua metode yaitu *Integer Linear Programming,* dan *Variable Neighborhood Search* lalu membandingkan metode mana yang memiliki hasil paling optimal lalu menerapkan ke studi kasus yang ada.
Keterkaitan Penelitian	Dapat dijadikan acuan dalam membuat model matematis dari Batasan-batasan yang ditemukan dalam penelitian tersebut sehingga dapat dilakukan pembandingan baik atau tidaknya hasil dari penelitian.

2.2. Dasar Teori

2.2.1. Rumah Sakit Kendangsari Surabaya

RSIA Kendangsari Surabaya adalah Rumah Sakit Ibu dan Anak Surabaya yang dirancang unik berfokus untuk melayani kebutuhan ibu dan anak. Dalam menjalankan fungsinya RSIA Kendangsari Surabaya memberikan pelayanan kesehatan yang paripurna untuk wanita dan anak. Pelayanan kesehatan diberikan secara prima dan komprehensif bagi pasien,keluarga pasien dan provider baik perusahaan maupun asuransi.

RSIA Kendangsari Surabaya merupakan bentuk dari satu kesatuan visi dan misi yang sama oleh beberapa dokter spesialis kebidanan dan kandungan dalam upaya meningkatkan "kualitas kesehatan reproduksi kaum perempuan khususnya ibu hamil dan janin yang dikandungnya." Atas dasar keinginan tersebut maka dibentuklah PT. Sandra Buana Medika. PT. Sandra Buana Medika kemudian mengajukan ijin untuk mendirikan Rumah Sakit Ibu dan Anak pada tanggal 4 April 2009 dengan lingkup bidang usaha jasa rumah sakit swasta [5].

2.2.2. Jain Fairness Index

Jain Fairness Index adalah perhitungan untuk melakukan pengukuran terhadap kesetaraan dengan menggunakan pendekatan secara kuantitatif. Kesetaraan atau *fairness* pada *Jain Fairness Index* mengartikan keseimbangan dari alokasi sumber daya [6]. Berikut adalah persamaan dari Jain Fairness Index, nilai dari JFI ini memiliki rentang 0-1.

$$f(x) = \frac{\left(\sum_{s=1}^{S} P_s\right)^2}{\left(S * \sum_{s=1}^{S} (P_s)^2\right)} \quad \textbf{(2.1)}$$

nilai *s* meyatakan user atau pengguna yang ingin dihitung dan user ke-*s* akan menerima alokasi Ps. Jika semua user mendapatkan nilai yang sama (Nilai Ps sama) maka nilai dari *Jain Fairness Index* adalah 1, dan berarti sistem tersebut memiliki *fairness* yang sempurna (sama rata) jika perbedaan semakin jaug maka

2.2.3. Nurse Rostering Problem

Nurse Rostering Problem atau permasalahan penjadwalan perawat biasanya mempertimbangkan hari libur dan *shift* secara serentak mereka melakukan tahap pertama dalam melakukan proses penjadwalan di waktu yang sama. Input yang dimasukkan terdiri dari *outcome* dari proses pemodelan permintaan, dan deskripsi dari perarwat dalam hal kualifikasi, kontrak, ketersediaan, permintaan dan

karakteristik personal lainnya yang relevan terhadap skenario.

Tujuan dari *Nurse Rostering Problem* adalah untuk menetapkan *shift* kepada perawat untuk memenuhi tuntutan staf, dan menjadikan staf untuk patuh terhadap batasan tersebut. Periode penjadwalan dimana penugasan tersebut dibuat bergantung kepada organisasi tatapi biasanya memiliki rentang waktu empat minggu sampai tiga bulan. Pada intinya penjadwalan perawat ini adalah masalah yang *multi-objective,* dimana pada tiap *stakeholder* memiliki aspek dari solusi yang berbeda-beda, contohnya pihak manejemen ingin fokus kepada regulasi keamanan dan tingkat kepuasan keshatan, dipihak lain seperti kepala perawat ingin menghormati pilihan dan keinginan perawat sebanyak mngkin, sehingga akan menimbulkan kontradiksi yang membuat pemodelan menjadi sulit untuk menyelesaikan masalah. [1]

2.2.4. Pemodelan Matematis Nurse Rostering Problem

Berikut akan dijelaksan pemodelan dari Nurse Rostering Problem

2.2.4.1. Variabel Keputusan , Parameter

Pada umumnya *Nurse Rostering Problem* di deskripsikan oleh pandangan perawat, tugas perawat atau waktu perawat dan pandangan pola shift perawat.

Pandangan perawat adalah penggambaran dari daftar tugas dua dimensi. Dengan demikian nmaka variable keputusan dapat di tentukan untuk setiap perawat pada tiap harinya [7]. Dalam membuat model ada beberapa hal yang dimodelkan secara matematis sebagai berikut:

- Variabel Keputusan

Tabel 2.4 Variabel Keputusan Nurse Rostering Problem

- x_{etd} = bernilai 1 jika perawat *e* diberikan tipe *shift t* pada hari *d*, dan bernilai 0 bila tidak ada pemberian kerja pada shift tersebut.
- n_{er} = Angka kecocokan dari *regular expression r* di dalam jadwal kerja dari perawat *e.*
- p_{ew} = Jumlah jam kerja yang dialokasikan untuk perawat *e* selama periode waktu yang didefinisikan dengan batas kerja *w.*
- q_{td} = Jumlah tipe *shift t* yang dialokasikan untuk hari *d.*

- Parameter

Tabel 2.5 Parameter Nurse Rostering Problem

- r_{er}^{max} = Angka maksimal kecocokan antara *regular expression r* di dalam jadwal kerja perawat *e.*
- r_{er}^{min} = Angka minimal kecocokan antara *regular expression r* di dalam jadwal kerja perawat *e.*
- a_{er} = Bobot yang diasosiasikan dengan *regular expression r* untuk perawat *e.*
- v_{ew}^{max} = Angka maksimal dari jam yang dialokasikan untuk perawat *e* selama dalam periode waktu yang didefinisikan oleh batasan waktu kerja.
- v_{ew}^{min} = Angka minimal dari jam yang dialokasikan untuk perawat *e* selama dalam

periode waktu yang didefinisikan oleh Batasan waktu kerja.

- b_{ew}= Bobot yang diasosiasikan dengan Batasan waktu kerja *w* untuk perawat *e*.
- s_{td}^{max} = Angka maksimum dari tipe shift *t* yang dibutuhkan pada hari *d.*
- s_{td}^{min}= Angka minimum dari tipe shift *t* yang dibutuhkan pada hari *d.*
- c_{td} = Bobot yang diasosiasikan dengan kebutuhan minimal tipe *shift t* dalam hari *d.*

2.2.4.2. Batasan

Batasan yang umumnya terjadi dalam *Nurse Rostering Problem* bisa dibagi menjadi dua secara general: *hard constrant,* dan *soft constaraint. Hard constraints* biasanya sudah termasuk dalam cakupan kebutuhan (contoh, permintaan staff perhari, pershift, dan per ketegori skill) dimana *soft constraints* biasanya terlibat dalam kebutuhan waktu pada jadwal personal. Tujuannya adalah untuk selalu menjadwalkan *resources* untuk memenuhi *hard constraint* saat berusaha mencapai hasil dengan kualitas tinggi dan menghargai *soft constraints* [7]. Biasanya C*onstraint* yang ada adalah sebagai berikut:

1. Beban kerja perawat
2. Shift kerja yang sama berturut-turut
3. Shift kerja yang sama berturut-turut perhari
4. Tingkat keterampilan perawat
5. Preferensi atau persyaratan perawat
6. Hari libur perawat
7. Waktu kosong diantara shift kerja
8. Tipe shift penugasan
9. Liburan (dapat diprediksi)

10. Bekerja di akhir pekan
11. Batasan diantara kelompok atau jenis perawat yang tidak diperbolehkan untuk begerja bersamaan
12. Pola shift
13. Riwayat catatan (contoh : Penugasan sebelumnya)
14. Persyaratan lain dalam jangka waktu yang lebih lama selain periode waktu perencanaan, contohnya setiap hari satu shift harus dilaksanakan
15. Batasan antar shift (contoh : shift tidak bisa ditugaskan kepada satu orang di waktu yang sama)
16. Persyaratan berbeda untuk perawat atau permintaan staff pada setiap shift. [7]

Contoh pemodelan matematis terhadap salah satu *Constraint :*

- Perawat hanya bisa dialokasikan ke dalam satu shift per hari

$$\sum_{t \in T} x_{etd} \leq 1, \forall e \in E, d \in D$$

2.2.4.3. Fungsi Objective

Tabel 2.6 Fungsi Objective Nurse Rostering Problem

$$Min\ f(s) = \sum_{e \in E} \sum_{i=1}^{4} f_{e,i}(x) + \sum_{t \in T} \sum_{d \in D} \sum_{i=5}^{6} f_{t,d,i}(x) \qquad (2a)$$

$$where$$

$$f_{e,1}(x) = \sum_{e \epsilon R_e} max\ \{0, (n_{er} - r_{er}^{max}) a_{er}\} \qquad (2b)$$

$$f_{e,2}(x) = \sum_{e \epsilon R_e} max\ \{0, (r_{er}^{min} - n_{er})a_{er}\} \quad (2c)$$

$$f_{e,3}(x) = \sum_{e \epsilon R_e} max\ \{0, (p_{ew} - v_{ew}^{max})b_{ew}\} \quad (2d)$$

$$f_{e,4}(x) = \sum_{e \epsilon R_e} max\ \{0, (v_{ew}^{min} - p_{ew})b_{ew}\} \quad (2e)$$

$$f_{e,5}(x) = max\{0, (s_{td}^{min} - q_{td})c_{td}\} \quad (2f)$$

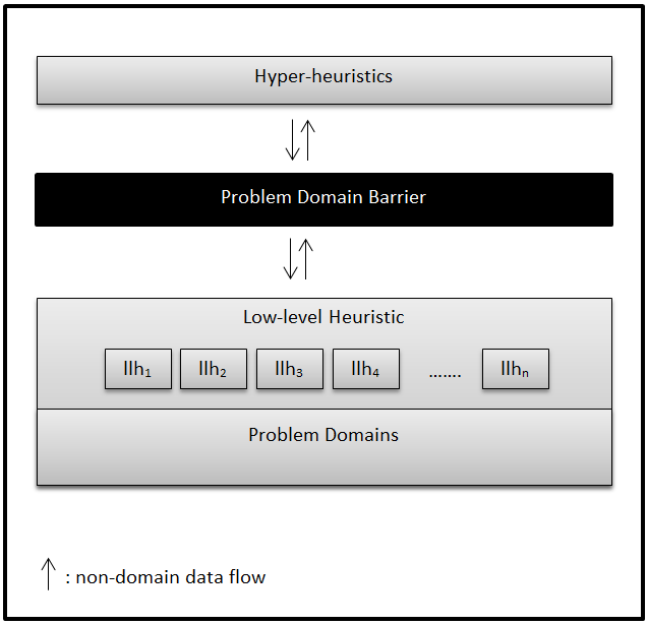
$$f_{e,6}(x) = max\{0, (q_{td} - s_{td}^{max})c_{td}\} \quad (2g)$$

2.2.5. Hyper Heuristic

Pada dasarnya, *Hyper-Heuristic* adalah gagasan untuk mengembangkan algoritma yang lebih generic yang berkebalikan dengan pendekatan *metaheuristic* standar yang biasanya membutuhkan *parameter tuning* yang lebih intensif untuk menyelesaikan *domain problem* yang berbeda. Untuk mengatasi masalah ini, *hyper-heuristic* datang degan ide untuk bekerja di level yang lebih tinggi contohnya menggunakan *low-level heuristic search space* dibandingkan langsung melalui *solution search-space.* Mengangkat *search space* dimaksudkan agar *hyper-heuristic* lebih *generic* dan berkinerja baik lintas masalah domain.

Definisi dari *Hyper-heuristic* adalah pendekatan secara *heuristic* dimana melakukan pencarian tidak lengkap yang tidak menjamin kesuksesan untuk menemukan solusi optimal, sebuah

prosess iterative yang memandu heuristic subordinat dengan menggabungkan konsep secara pintar untuk mengeksploitasi *search space* menggunnakan strategi pembelajaran untuk membangun informasi dengan tujuan untuk menemukan solusi optimal terdekat [8].



Gambar 2.1 Hyper Heuristic

2.2.6. Algoritma Hill Climbing Hyper Heuristic

Algoritma *Hill Climbing* dilakukan dengan cara memilih sebuah "suksesor" dalam pada setiap tahap pencarian, dalam standard *hill climbing, success value* diambil berdasarkan yang lebih baik daripada keadaan saat ini (*current state)* [9]
Dalam penelitian ini, Algoritma Hill Climbing dijadikan algoritma pembanding untuk mendapatkan solusi terbaik dari penjadwalan yang dilakukan

2.2.7. Algoritma Self-Adaptive Learning

Algoritma *Self-adaptive learning* dilakukan berdasarkan *self-adaptive* strategy, yang berarti algoritma tersebut bekerja dengan carasi memilih struktur lingkungan dalam algoritma *artificial bee colony* untuk menyelesaikan *flowshop scheduling* [8] yaitu penjadwalan yang melibatkan perintah yang ketat dalam setiap operasi yang dilakukan dalam pekerjaan, hal tersebut sesuai dengan penjadwalan perawat yang memiliki aturan ketat dalam waktu jam kerja dan operasionalnya.

Dalam penelitian ini algortima *self-adaptive learning* menggunakan pendekatan *hyper heuristic*, berikut ini adalah contoh daftar variable yang digunakan dalam algoritma *self-adaptive learning*, gambar dibawah akan menjelaskan bagaimana algoritma *self-adaptive* bekerja yang berasal dari referensi [8]

Tabel 2.7 List Variable yang digunakan dalam Self-Adaptive Learning Algorithm

Variable	Description
P	Domain dari masalah, contoh: pemeriksaan masalah *timetabling*
St	Kondisi Berhenti
M	Jumlah Standar *low-level heuristic* yang ada didalam P
LH	Suatu *ArrayBlockingQueue* dari sebuah tipe data integer untuk menyimpan indeks sementara dari *low-level heuristics*
S	Ukuran dari LH
WH	Suatu *LinkedLisit* dari tipe data integer untuk menyimpan

Variable	Description
	low-level heuristic dengan performa yang baik
Sbest	Solusi terbaik dari nilai fungsi objektif
L	*Low-level heuristic* yang terpilih
newFunctionVal	Nilai fungsi objektif dari solusi baru yang dihasilkan dengan menerapkan *low-level Heuristic*
Scur	Fungsi solusi objective yang digunakan pada saat iterasi (*current)*
B	*Boundary levl*
α	*Decay rate*
N	Jumlah Iterasi
R	Suatu *ArrayBlockingQueue* dari tipe data *Double* untuk menyimpan nilai dari fungsi solusi objektif yang diterima
L	Ukuran dari R
C	Elemen indeks terpilih dalam R

Tabel 2.8 List Method pada Algoritma Self-Adaptive Learning

Method	*Description*
P.getNumberOfHeuristics()	Mengambil jumlah dari *low-level heuristic* dari Problem domain P.
GetRand(M)	*Generate* angka random positif kurang dari M

Method	*Description*
P.getFunctionValue(i)	*Generate* solusi awal yang layak dengan algoritma 1 dan menyimpannya ke dalam solusi memori di index i.
LH.poll()	Mendapatkan elemen dari LH dengan Index 0 dan menghapusnya dari LH
P.applyLLH(l,i,j)	Menerapkan standar *low-level heuristic* dalam problem domain P dengan index 1 dengan solusi yang disimpan di *solution memory i* dan menyimpan solusi baru di *solution memory j.* Mengembalikan nilai fungsi objektif ke solusi yang baru
P.copySolution(i,j)	Menyalin solusi ke *solution memory index i* ke *solution memory index j.*
WH.add(l)	menambahkan *low-level heuristic index l* ke *list index low-level heuristic*

Method	*Description*
	yang memiliki performa baik, WH.
R.add(x)	Menambahkan nilai *x* ke *ArrayBlockingQueue R*
R.get(x)	Mengambil nilai dari *ArrayBlockingQueue R* dalam index *x.*
R.poll()	Menghapus secara permanen elemen dalam *ArrayBlockingQueue R* yang memiliki index 0
R.put(a)	Memasukkan kedalam *ArrayBlockingQueue R.*

Tabel 2.9 Pseudocode Algoritma Self Adaptive Learning

Algoritma Self-Adaptive – Hill Climbing Based Hyper Heuristic.

```
1:  procedure SOLVE (ProblemDomain P)
2:  Set stopping condition St
3:  Set M ← P.getNumberOfHeuristics()
4:  Set ArrayBlockingQueue dari Integer dengan size S untuk
    menyimpan index dari low-level heuristic, LH
5:  Set LinkedList dari Integer untuk menyimpan index dengan
    low-level heuristic yang memiliki performa baik.
6:  for i=0 to S-1 do
7:      LH.add(getRand(M))
8:  end for
9:  P.initiateSolution(0)
```

```
10: Sbest ← P.getFunctionValue(0)
11: While is not met do
12:     if LH is not empty then
13:       1← LH.poll()
14:     else
15:       isi 75% elemen dari LH dengan elemen dari WH
16:       isi 25% elemen dari LH dengan index dari standar low
      level heuristics yang ada didalam P yang dipilih secara
      random
17:       clear elemen dari WH
18:       1 ← LH.poll()
19:     end if
20:     newFunctionVal ← P.applyLLH(1,0,1)
21:     if newFunctionalVal ≤ Sbest then
22:       //solusi baru diterima
23:       P.copySolution(1,0)
24:       Sbest ← newFunctionVal
25:       WH.add(1)
26:     end if
27:   end while
28:   return while
29:   return Sbest
30: end procedure
```

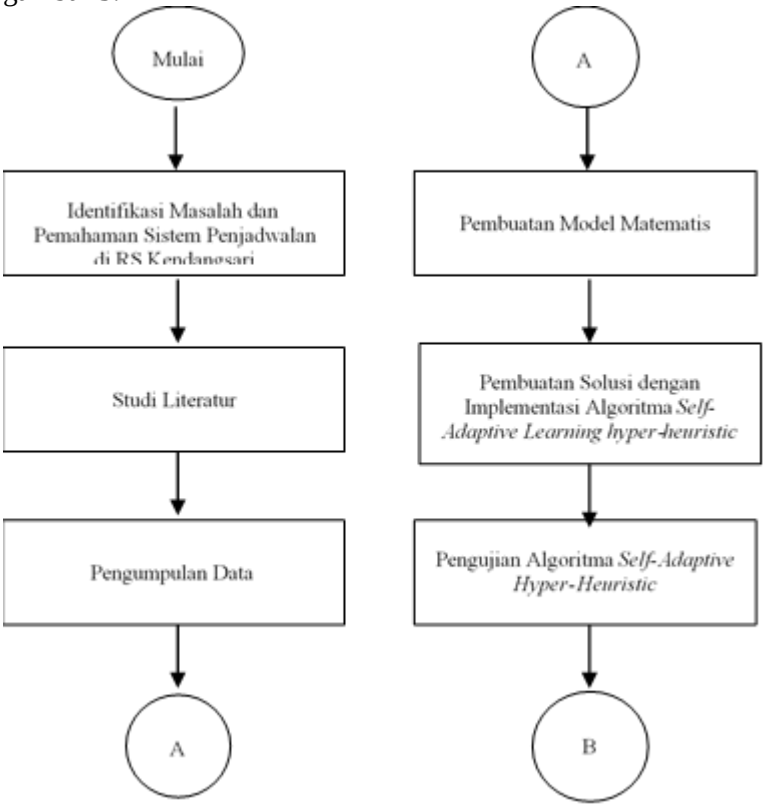
(Halaman ini sengaja dikosongkan)

BAB III
METODOLOGI PENELITIAN

Pada bab ini menjelaskan terkait metodologi yang akan digunakan sebagai panduan untuk menyelesaikan penelitian tugas akhir ini.

3.1. Metodologi Penelitian

Diagram Metodologi dari Tugas Akhir ini dapat dilihat pada gambar 3.1



Gambar 3.1 Metodologi Penelitian (1)

Gambar 3.2 Metodologi Penelitian (2)

3.2. Uraian Metodologi

Dibawah ini adalah penjelasan dari setiap proses alur metodologi berdasarkan diagram alur metodologi pada sub bab sebelumnya

3.2.1. Identifikasi Permasalahan

Pada proses ini dilakukan identifikasi permasalahan dengan melakukan analisa proses bisnis. Proses

identifikasi permasalahan dilakukan dengan cara melakukan wawancara dengan pihak Bulog Divre Jawa Timur terkait proses bisnis Bulog terutama dalam pelaksanaan Program RASKIN. Permasalahan yang ditemukan di Bulog Divre Jatim berkaitan dengan proses distribusi beras RASKIN. Proses distribusi RASKIN di Jawa Timur dilakukan untuk 38 wilayah kabupaten atau kota dengan 59 gudang.

3.2.2. Studi Literatur

Studi Literatur termasuk tahap persiapan dari pelaksanaan penelitian ini, dimana pada tahap ini kita sudah menememukan masalah yang ada lalu mencari referensi yang berasal dari buku atau penelitian sebelumnya yang memiliki keterkaitan dengan masalah yang ada. Setelah mengetahui permasalahan, faktor-faktor yang mempengaruhi permasalahan dan juga batasannya, maka langkah berikutnya yaitu menentukan metode penyelesaian permasalahan. Untuk memilih metode yang sesuai dengan permasalahan, maka diperlukan analisis terhadap metode-metode yang telah ada yang berasal dari laporan penelitian dengan permasalahan yang sama.

3.2.3. Pengumpulan Data dan Wawancara

Pada tahapan ini, dilakukan setelah didapatkan permasalahan dan diketahui mengenai konsep tentang masalah yang akan dihadapi adalah melakukan pengumpulan data, data yang akan digunakan adalah jadwal atau shift perawat untuk 1 bulan yang akan dilakukan optimasi..

3.2.4. Pemodelan Permasalahan

Setelah mendapatkan seluruh data yang diperlukan untuk menyelesaikan permasalahan, maka diperlukan pembuatan model matematika dari data-data yang sudah diperoleh, pemodelan mengacu pada sub bab

Tujuan dari pemodelan ini adalah untuk menentukan fungsi tujuan,variabel keputusan dan juga batasan dari permasalahan yang didapatkan, diantaranya :

A. Fungsi Tujuan

Fungsi Tujuan merupakan tujuan yang akan dicapai dari tugas akhir ini. Fungsi Tujuan pada tugas akhir ini adalah memaksimalkan nilai dari *Jain Fairness Index* dari jadwal tiap-tiap divisi pada RSIA Kendangsari.

B. Variabel Keputusan

Set of Nurse yang bekerja pada shift perhari

C. Fungsi Batasan

Batasan merupakan kumpulan dari beberapa fungsi yang digunakan untuk memberikan batasan dalam variable keputusan dalam mencapai fungsi tujuan. Batasan yang digunakan dalam tugas akhir ini dibagi menjadi dua bagian yaitu *Hard Constraint* dan *Soft Constraint*. *Constraint* akan berubah sesuai kebutuhan dari rumah sakit. Berikut merupakan contoh daftar constraint tersebut:

1. *Hard Constraint*

- Jumlah shift yang harus diisi oleh satu perawat
- Setiap shift selalu ada pegawai yang bekerja
- Setiap perawat harus mendapatkan shift pagi, siang, dan malam
- Jumah maksimal hari libur perawat
- Jumlah hari bekerja selama 1 bulan

2. *Soft Constraint*

- Batas minimal dan maksimal jumlah perawat dalam tiap shift nya
- Kebutuhan personal dari perawat

3.2.5. Pembuatan Solusi dengan Implementasi Algoritma Self-Adaptive-Learning Hyper-Heuristic

Setelah pembuatan model selesai, maka berikutya adalah membuat solusi menggunakan *Self-adaptive learning algorithm.* Pada tahap ini dibuat sebuah algoritma *self-adaptive learning* dengan menggunakan bahasa pemrograman *Java* diawali dengan melakukan otomasi untuk melakukan *generate* jadwal secara otomatis dan nantinya akan dioptimasi dengan algoritma *Self-Adaptive Learning Hyper Heuristic* untuk memaksimalkan nilai dari *Jain Fairness Index.*

3.2.6. Pengujian Algoritma Self-Adaptive Hyper Heursistic

Setelah dilakukan pembuatan algoritma *Self-Adaptive Learning Hyper-heuristic* maka akan dilakukan pengujian dengan membandingkan dengan *Jain Fairness Index* dengan jadwal sebelum dilakukan optimasi, apabila sudah lebih baik maka algoritma dikatakan berhasil.

3.2.7. Analisis Hasil dan Penarikan Kesimpulan

Pada tahap ini dilakukan penarikan kesimpulan yang dilakukan dengan melihat hasil yang di dapatkan dari pengujian algoritma lalu dari hasil tersebut akan didapatkan jadwal yang optimal, setelah didapatkan jadwalnya akan dibandingkan dengan jadwal yang telah dibuat secara manual di rumah sakit kendangsari untuk mendapatkan kesimpulan apakah jadwal cocok dengan *Soft Constraint* dan *Hard Constraint* yang ada .Setelah kesimpulan dan solusi permasalahan,

selanjutnya akan muncul pendapat akan saran penelitian apa yang harus dikembangkan agar hasil yang didapatkan lebih optimal dibandingkan dengan penelitian yang sekarang.

3.2.8. Penyusunan Laporan Tugas Akhir

Penyusunan laporan merupakan tahap akhir dari proses-proses yang telah dilakukan sebelumnya. Disini dilakukan dokumentasi terhadap proses-proses yang telah dilakukan dan kesimpulan dari permasalahan yang didapatkan. Seluruh pelaksanaan ataupun pengerjaan tugas akhir ini akan didokumentasikan dengan mengukuti format yang telah ditetapkan oleh laboratorium Rekayasa Data dan Intelegensia Bisnis (RDIB) serta yang berlaku di Jurusan Sistem Informasi ITS.

Di dalam laporan Tugas Akhir akan mencakup :

a. BAB I Pendahuluan

Pada bab ini akan dijelaskan mulai dari latar belakang, rumusan permasalahan, batasan permasalahan, tujuan dan manfaat pengerjaan tugas akhir.

b. BAB II Tinjauan Pustaka

Pada bab ini akan dijelaskan mengenai penelitian-penelitian sebelumnya yang telah dilakukan serta teori-teori yang menunjang permasalahan yang dibahas pada tugas akhir ini

c. BAB III Metodologi

Pada bab ini akan dijelaskan alur proses dari pengerjaan tugas akhir mulai dari identifikasi permasalahan sampai pembuatan laporan tugas akhir.

d. BAB IV Data Masukan dan Pemodelan

Pada bab ini akan dijelaskan proses pengumpulan dan juga deskripsi data-data yang digunakan dalam penyelesaian tugas akhir serta melakukan pemodelan dari data yang didapatkan.

e. BAB V Implementasi

Bab ini berisi tentang implementasi dan penjelasan kode program setiap alur proses yang telah dijabarkan sebelumnya pada metodologi yang digunakan dalam tugas akhir.

f. BAB VI Hasil dan Pembahasan

Bab ini berisi analisis dan pembahasan daam penyelesaian permasalahan yang dibahas pada pengerjaan tugas akhir.

g. BAB VII Kesimpulan dan Saran

Bab ini berisi kesimpulan dan saran yang ditujukan untuk kelengkapan penyempurnaan tugas akhir ini.

(Halaman ini sengaja dikosongkan)

BAB IV
DATA MASUKAN DAN PEMODELAN

Pada bagian ini akan dijelaskan mengenai Batasan-batasan serta fungsi tujuan yang didapatkan berdasarkan pengumpulan data dari RSIA Kendangsari MERR, Surabaya yang akan dilakukan otomasi dan dilakukan otomasi menggunakan java dan di optimasi menggunakan algoritma *Self Adaptive Hyper Heuristic*

4.1. Hasil Pengumpulan Data

Pada Rumah Sakit Ibu dan Anak Kendangsari MERR, Surabaya terdapat 6 (enam) divisi yang akan dilakukan otomasi dan optimasi penjadwalan yaitu Divisi Farmasi, Poli/IGD (Instalasi Gawat Darurat), Bayi&NICU (Neonatal Intensive Care Unit), Gizi, dan OK (Operatie Kamer). Jadwal dibuat berdasarkan beberapa hal yaitu instruksi dari Menteri kesehatan, regulasi yang berlaku di rumah sakti dan keadaan personal dari tiap perawat. Berdasarkan jumlah pasien yang ditangani oleh rumah sakit pada tiap divisi memiliki jumlah perawat/staff yang berbeda-beda, untuk waktu kerja tiap perawat adalah 7 jam dalam satu hari yang berarti setiap perawat harus mengisi satu shift pada tiap harinya serta mendapatkan 1 hari jatah libur dalam satu minggu serta 1 kali jatah cuti dalam satu bulan.

Lalu dengan melihat sistem regulasi dari RSIA Kendangsari MERR, Surabaya memiliki 5 jenis shift diantaranya yaitu Shift Pagi yang dimulai pukul 07.00 sampai 14.00 lalu Shift Sore dari pukul 14.00 sampai 21.00, Shift Malam dari pukul 21.00 sampai jam 7.00, Shift Lepas Malam dimana shift ini merupakan shift khusus yang ada setelah 2(dua) kali Shift Malam, dan Middle Shift yang dimulai pukul 10.00-17.00 selain itu untuk memperhitungkan kesehatan perawat setelah menjaga 2 (dua) kali Shift Malam maka akan mendapatkan jatah shift Lepas Malam yaitu kosong pada hari tersebut dan mendapatkan hari libur untuk hari selanjutnya.

Total Jumlah Pegawai yang ada di Rumah Sakit Kendangsari MERR, Surabaya terdapat 126 orang termasuk security dan *cleaning services.*

Berdasarkan peraturan yang ada, penjadwalan dilakukan oleh masing-masing kepala unit dan dijadwalakan berdasarkan peraturan masing-masing divisi dan ada beberapa divisi yang memiliki shift yang berbeda diantaranya divisi Gizi yang membagi shift pagi menjadi 2 yaitu ada pagi yang dimulai dari jam 5 dan yang dimulai pukul 7, serta divisi IGD yang memiliki dua tipe shift middle ada yang dimulai dari pukul 10 pagi dan ada yang dimulai pukul 12 siang.

Dalam melakukan penjadwalan pada tiap divisi menggunakan pola M-M-LM-L dimana pola ini berarti untuk menentukan libur masing-masing perawat untuk mendapatkan jatah libur melewati 2(dua) kali shift malam terlebih dahulu lalu shift lepas malam dimana shift lepas malam adalah bagian dari shift malam.

Tetapi dengan kondisi penjadwalan saat ini dibuat berdasarkan dengan kebijkana kepala perawat masing-masing unit tanpa melakukan pertimbangan keinginan perawat sehingga jadwal kemungkinan mempengaruhi etos kerja dari perawat.

4.1.1. Kebijakan dan Regulasi Rumah Sakit

Kebijakan dalam penjadwalan rumah sakit ditetapkan oleh rumah sakit dan kepala perawat masing-masing divisi melalui berbagai pertimbangan yang nantinya hasil dari pertimbangan tersebut akan dimasukkan kedalam jadwal, Untuk shift lepas malam karena untuk mendapatkan shift tersebut diperlukan 2 kali shift malam sehingga jika dihitung keseluruhan shift malam 2 (dua) kali menjadi 10 x 2 = 20 jam, tiap shift malam ada jam berlebih sebesar 3 jam sehingga 3 x 2 = 6, durasi untuk Lepas Malam adalah 6 jam kerja kabijakan dari masing-masing divisi adalah sebagai berikut:

Kebijakan untuk semua :

- Setiap Perawat dalam satu hari hanya boleh mengisi 1 shift per hari
- Perawat bekerja minimal 42 jam/minggu
- Tiap perawat mendapatkan shift pagi, sore dan malam

Kebijakan masing-masing divisi

1. Divisi Farmasi
   - Shift Terdiri dari 5 jenis yaitu
     - P (Pagi) pukul 7.00 – 14.00 (7 Jam),
     - S (Sore) pukul 14.00 – 21.00 (7 Jam),
     - MS (Middle Shift) pukul 10.00 – 17.00,
     - M (Malam) pukul 21.00 – 7.00(10 Jam) ,
     - LM (Lepas Malam).
   - Untuk kepala perawat dan bagian Gudang apotek memiliki jam kerja kantor (pukul 7.00 – 16.00)

2. Divisi IGD (Instalasi Gawat Darurat)
   - Shift Terdiri dari 6 jenis yaitu
     - P (Pagi) pukul 7.00 – 14.00 (7 Jam),
     - S (Sore) pukul 14.00 – 21.00 (7 Jam),
     - MS (Middle Shift 1) pukul 10.00 – 17.00,
     - MD (Middle Shift 2) pukul 12.00 – 19.00,
     - M (Malam) pukul 21.00 – 7.00(10 Jam) ,
     - LM (Lepas Malam).

3. Divisi NICU (Neonatal Intensive Care Unit) & Bayi
   - Shift Terdiri dari 6 jenis yaitu
     - P (Pagi) pukul 7.00 – 14.00 (7 Jam),
     - S (Sore) pukul 14.00 – 21.00 (7 Jam),
     - MS (Middle Shift 1) pukul 10.00 – 17.00,
     - MD (Middle Shift 2) pukul 12.00 – 19.00,
     - M (Malam) pukul 21.00 – 7.00(10 Jam) ,
     - LM (Lepas Malam).

4. Divisi Gizi
   - Shift Terdiri dari 6 jenis yaitu

- P1 (Pagi) pukul 5.00 – 12.00 (7 Jam)
- P2 (Pagi) pukul 7.00 – 14.00 (7 Jam),
- S (Sore) pukul 14.00 – 21.00 (7 Jam),
- MS (Middle Shift 1) pukul 10.00 – 17.00,
- MD (Middle Shift 2) pukul 12.00 – 19.00,
- M (Malam) pukul 21.00 – 7.00(10 Jam) ,
- LM (Lepas Malam).

5. Divisi OK (Operation Kamer)
   - Shift Terdiri dari 6 jenis yaitu
     - P1 (Pagi) pukul 7.00 – 14.00 (7 Jam)
     - S (Siang) pukul 14.00 – 21.00 (7 Jam),
     - On Call.

6. Divisi SIM RM (Sistem Informasi Manajemen & Rekam Medis)
   - Shift Terdiri dari 6 jenis yaitu
     - P (Pagi) pukul 7.00 – 14.00 (7 Jam),
     - S (Sore) pukul 14.00 – 21.00 (7 Jam),
     - MS (Middle Shift 1) pukul 10.00 – 17.00,
     - MD (Middle Shift 2) pukul 12.00 – 19.00,
     - M (Malam) pukul 21.00 – 7.00(10 Jam) ,
     - LM (Lepas Malam).

4.2. Pembuatan Model untuk Masalah Penjadwalan

Masalah penjadwalan perawat pada divisi yang ada di RSIA Kendangsari MERR, Surabaya adalah menjadwalkan perawat dalam waktu 1 bulan yang memenuhi waktu kerja dan batasan yang lainnya. Lalu dalam permaslaahan ini diusahakan untuk memenuhi tujuan yang telah ditentukan.

4.2.1. Tipe Shift Masing-Masing Unit

Tabel 4.1 Tipe Shift Masing-Masing Staff

Tipe Shift	Notasi	No. ID	Jam Kerja
Pagi	P	1	07.00 – 14.00 WIB
Pagi 1	P1	2	04.30 – 11.30 WIB
Pagi 2	P2	3	05.00 – 12.00 WIB
Pagi-Sore	PS	4	06.00 – 20.00 WIB
Siang	S	5	14.00 – 21.00 WIB
Malam	M	6	21.00 – 07.00 WIB
Lepas Malam	LM	7	-
Middle Shift	MS	8	10.00 – 17.00 WIB

Tipe Shift	Notasi	No. ID	Jam Kerja
Middle Shift 2	MD	9	12.00 – 19.00 WIB
Libur	L	10	-

Tipe Skill

1. Instalasi Farmasi

Tabel 4.2 Tips Skill Instalasi Farmasi

No	Kode Staff	Skill	Tipe Skill
1	101	Apoteker	1
2	102	Staff Senior	2
3	103	Anggota	4
4	104	Anggota	4
5	105	Anggota	4
6	106	Anggota	4
7	107	Anggota	4

No	Kode Staff	Skill	Tipe Skill
8	108	Petugas Gudang Obat	3

2. Ruang Bayi dan NICU

Tabel 4.3 Tipe Skill Ruang Bayi dan NICU

No	Kode Staff	Skill	Tipe Skill
1	201	Kepala Unit	1
2	202	Anggota	4
3	203	Anggota	4
4	204	Anggota	4
5	205	Anggota	4
6	206	Anggota	4
7	207	Anggota	4
8	208	Anggota	4

No	Kode Staff	Skill	Tipe Skill
9	209	Anggota	4
10	210	Anggota	4
11	211	Anggota	4
12	212	Anggota	4

3. SIM Rekam Medis dan Registrasi

Tabel 4.4 Tipe Skill SIM & RM

No	Kode Staff	Skill	Tipe Skill
1	301	Anggota	4
2	302	Anggota	4
3	303	Anggota	4
4	304	Anggota	4
5	305	Anggota	4
6	306	Anggota	4

4. Instalasi Gizi

Tabel 4.5 Tipe Skill Gizi

No	Kode Staff	Skill	Tipe Skill
1	401	Ahli Gizi	1
2	402	Chef	5
3	403	Chef	5
4	404	Chef	5
5	405	Chef	5
6	406	Helper	6
7	407	Helper	6
8	408	Helper	6
9	409	Helper	6
10	410	Penyaji	4
11	411	Penyaji	4

No	Kode Staff	Skill	Tipe Skill
12	412	Penyaji	4
13	413	Penyaji	4
14	414	Penyaji	4
15	415	Penyaji	4
16	416	Café	7
17	417	Café	7
18	418	Driver	8
19	419	Driver	8

5. Instalasi Rawat Jalan dan IGD

Tabel 4.6 Tipe Skill IGD

No	Kode Staff	Skill	Tipe Skill
1	501	Anggota	4

No	Kode Staff	Skill	Tipe Skill
2	502	Anggota	4
3	503	Anggota	4
4	504	Anggota	4
5	505	Anggota	4
6	506	Anggota	4
7	507	Anggota	4
8	508	Anggota	4

6. Kamar Operasi

Tabel 4.7 Tipe Skill OK

No	Kode Staff	Skill	Tipe Skill
1	601	Anggota	4
2	602	Anggota	4

No	Kode Staff	Skill	Tipe Skill
3	603	Anggota	4
4	604	Anggota	4
5	605	Anggota	4

4.2.2. Batasan

Penyelesaian permasalahan penjadwalan perawat menggunakan *Algoritma Self-Adaptive Hyper Heuristic* memiliki beberapa Batasan sesuai dengan aturan dan regulasi yang berlaku. Batasan yang berlaku dibagi menjadi dua bagian yaitu *Hard Constraint* yaitu Batasan yang tidak dapat dilanggar dan *Soft Constraint* yaitu Batasan yang dapat dilanggar.

- *Hard Constraint*
  1. Divisi Farmasi :
     1) Jumlah staff pada shift pagi tiap harinya harus sama dengan 3
     2) Jumlah staff pada shift siang tiap harinya harus sama dengan 2
     3) Jumlah staff pada shift malam tiap harinya harus sama dengan 1
  2. Divisi Poli & IGD :
     1) Jumlah staff pada shift pagi di setiap hari adalah lebih dari sama dengan satu atau kurang dari sama dengan dua
     2) Jumlah staff pada shift siang pada satu hari lebih dari atau sama dengan dua dan kurang dari sama dengan empat

3) Jumlah staff pada Shift Malam dalam satu hari adalah lebih dari sama dengan 1 atau kurang dari sama dengan 2

3. Divisi NICU & Bayi :
   1) Jumlah staff pada shift pagi setiap hari harus lebih dari sama dengan dua dan kurang dari sama dengan empat.
   2) Jumlah staff pada shift siang setiap hari harus lebih dari sama dengan satu dan kurang dari sama dengan dua.
   3) Jumlah shift malam setiap hari harus lebih dari sama dengan dua dan kurang dari sama dengan tiga
4. Divisi Gizi :
   1) Jumlah staff pada shift pagi tiap harinya harus sama dengan 3
   2) Jumlah staff pada shift siang tiap harinya harus sama dengan 2
   3) Jumlah staff pada shift malam tiap harinya harus sama dengan 1
5. Divisi OK :
   1) Jumlah staff pada shift pagi setiap hari harus lebih dari sama dengan tiga dan kurang dari sama dengan empat
   2) Jumlah staff pada shift siang setiap hari harus sama dengan dua
   3) Jumlah staff pada shift pagi pada hari Minggu harus terdiri dari empat
6. Divisi SIM & RM :
   1) Jumlah staff pada shift pagi setiap hari harus lebih dari sama dengan satu dan kurang dari sama dengan dua
   2) Jumlah staff pada shift siang setiap hari harus sama dengan dua.
   3) Jumlah staff pada shift malam pada tiap hari harus terdiri dari satu

- *Soft Constraint*

1) Ukuran Optimalitas berdasarkan *Jain Fairness Index* tiap staff di masing-masing divisi.
2) Jumlah libur tiap staf dalam 1 bulan sama dengan jumlah hari minggu pada 1 bulan.

4.2.3. Variabel Keputusan

Variabel keputusan dari penelitian ini adalah

$\mathbf{X_{idt}}$ **=** bernilai 1 jika staf **i** ditugaskan pada tipe shift **t** pada hari **d**, bernilai 0 jika tidak.

Dimana

i = 1,2,3,…n; i adalah staff ke-, n adalah jumlah staff

d= 1,2,3,…k; k adalah jumlah hari periode penjadwalan, d adalah hari ke-k.

t= 1,2,3,…z; z adalah tipe shift, t adalah tipe shift ke-t **(lihat tabel 4.1**)

4.2.4. Notasi dan Asumsi

Tabel 4.8 Notasi dan Asumsi

Notasi	Deskripsi
K	Jumlah Total Karyawan
S	Tipa skill staf, s ∈ S
C	Jumlah hari minggu
i	Staff ke-i
j	Hari ke-j
M	Minggu
Sb	Sabtu
W	Weekdays

4.2.5. Pemodelan Hard Constraint

Pada pemodelan ini akan dibagi berdasarkan divisi, dikarenakan adanya perbedaan *Hard Constraint* pada tiap divisi yang ada di RSIA Kendangsari MERR Surabaya

4.2.5.1. Divisi Farmasi

Staff pada divisi farmasi dibagi atas 4(empat) jenis yaitu apoteker, staff senior, anggota, dan petugas obat dimana setiap jenis staff memiliki jenis penjadwalan yang berbeda-beda, berikut ini adalah Batasan yang digunakan **hanya** untuk anggota dikarenakan apoteker, staff senior dan petugas obat sudah memiliki jadwal yang fix dan tidak bisa diubah..

Hard Constraint 1

Jumlah staff pada shift pagi tiap harinya harus sama dengan 3. Dirumuskan untuk setiap nilai d

$$\sum_{i=1}^{K} X_{id1} = 3 \qquad \textbf{(4.1)}$$

Hard Constraint 2

Jumlah staff pada shift siang tiap harinya harus sama dengan 2. Dirumuskan untuk setiap nilai d

$$\sum_{i=1}^{K} X_{id2} = 2 \qquad \textbf{(4.2)}$$

Hard Constraint 3

Jumlah staff pada shift malam tiap harinya harus sama dengan 1. Dirumuskan untuk setiap nilai d

$$\sum_{i=1}^{K} X_{id3} = 1 \qquad \textbf{(4.3)}$$

Hard Constraint 4

Jumlah staff pada shift pagi pada hari Minggu harus terdiri dari satu. Dirumuskan untuk setiap nilai d jika nilai d adalah hari minggu.

$$\sum_{i=1}^{K} X_{id1} = 1 \qquad \textbf{(4.4)}$$

Hard Constraint 5

Dalam pembuatan jadwal ada peraturan untuk setiap pembuatan shift harus memenuhi pola shift Malam-Malam-Lepas Malam-Libur.

4.2.5.2. Divisi IGD

Staff pada divisi IGD untuk setiap staff atau pegawai yang ada tidak ada perbedaan jenis dimana semua staff dan perawat memiliki jadwal shift yang sejenis, berikut adalah Batasan untuk anggota divisi IGD

Hard Constraint 1

Jumlah staff pada shift pagi di setiap hari adalah lebih dari sama dengan satu atau kurang dari sama dengan dua. Dirumuskan untuk setiap nilai d

$$\sum_{i=1}^{K} X_{id1} \geq 1 \qquad \textbf{(4.5)}$$

$$\sum_{i=1}^{K} X_{id1} \leq 2 \qquad \textbf{(4.6)}$$

Hard Constraint 2

Jumlah staff pada shift siang pada satu hari lebih dari atau sama dengan dua dan kurang dari sama dengan empat. Dirumuskan untuk setiap nilai d

$$\sum_{i=1}^{K} X_{id2} \geq 2 \qquad \textbf{(4.7)}$$

$$\sum_{i=1}^{K} X_{id2} \leq 4 \qquad \textbf{(4.8)}$$

Hard Constraint 3

Jumlah staff pada Shift Malam dalam satu hari adalah lebih dari sama dengan 1 atau kurang dari sama dengan 2. Dirumuskan untuk setiap nilai d

$$\sum_{i=1}^{K} X_{id3} \geq 1 \qquad \textbf{(4.9)}$$

$$\sum_{i=1}^{K} X_{id3} \leq 2 \qquad \textbf{(4.10)}$$

Hard Constraint 4

Dalam pembuatan jadwal ada peraturan untuk setiap pembuatan shift harus memenuhi pola shift Malam-Malam-Lepas Malam-Libur.

4.2.5.3. Divisi NICU & BAYI

Staff pada divisi NICU & Bayi tidak terlalu berbeda secara umum hanya pada divisi NICU & Bayi untuk kepala perawat yang memiliki jadwal yang berbeda.

Hard Constraint 1

Jumlah staff pada shift pagi setiap hari harus lebih dari sama dengan dua dan kurang dari sama dengan empat . Dirumuskan untuk setiap nilai d

$$\sum_{i=1}^{K} X_{id1} \geq 2 \qquad \textbf{(4.11)}$$

$$\sum_{i=1}^{K} X_{id2} \leq 4 \qquad \textbf{(4.12)}$$

Hard Constraint 2

Jumlah staff pada shift siang setiap hari harus lebih dari sama dengan satu dan kurang dari sama dengan dua. Dirumuskan untuk setiap nilai d

$$\sum_{i=1}^{K} X_{id2} \geq 1 \qquad \textbf{(4.13)}$$

$$\sum_{i=1}^{K} X_{id2} \leq 2 \qquad \textbf{(4.14)}$$

Hard Constraint 3

Jumlah staff pada shift malam setiap hari harus lebih dari sama dengan dua dan kurang dari sama dengan tiga. Dirumuskan untuk setiap nilai d

$$\sum_{i=1}^{K} X_{id3} \geq 2 \quad \textbf{(4.15)}$$

$$\sum_{i=1}^{K} X_{id3} \leq 3 \quad \textbf{(4.16)}$$

Hard Constraint 4

Dalam pembuatan jadwal ada peraturan untuk setiap pembuatan shift harus memenuhi pola shift Malam-Malam-Lepas Malam-Libur.

4.2.5.4. Divisi Gizi

Staff pada divisi Gizi memiliki banyak tipe anggota, untuk itu Batasan berikut adalah Batasan secara umum untuk tipe skill penyaji divisi Gizi.

Hard Constraint 1

Jumlah staff pada shift pagi tiap harinya harus sama dengan 3. Dirumuskan untuk setiap nilai d

$$\sum_{i=1}^{K} X_{id1} = 3 \quad \textbf{(4.17)}$$

Hard Constraint 2

Jumlah staff pada shift siang tiap harinya harus sama dengan 2

$$\sum_{i=1}^{K} X_{id2} = 2 \quad \textbf{(4.18)}$$

Hard Constraint 3

Jumlah staff pada shift malam tiap harinya harus sama dengan 1

$$\sum_{i=1}^{K} X_{id3} = 1 \quad \textbf{(4.19)}$$

Hard Constraint 4

Dalam pembuatan jadwal ada peraturan untuk setiap pembuatan shift harus memenuhi pola shift Malam-Malam-Lepas Malam-Libur.

4.2.5.5. Divisi Operatie Kamer (OK)

Hard Constraint 1

Jumlah staff pada shift pagi setiap hari harus lebih dari sama dengan tiga dan kurang dari sama dengan empat. Dirumuskan untuk setiap nilai d

$$\sum_{i=1}^{K} X_{id1} \geq 3 \qquad \textbf{(4.20)}$$

$$\sum_{i=1}^{K} X_{id1} \leq 4 \qquad \textbf{(4.21)}$$

Hard Constraint 2

Jumlah staff pada shift siang setiap hari harus sama dengan dua. Dirumuskan untuk setiap nilai d

$$\sum_{i=1}^{K} X_{id2} = 2 \qquad \textbf{(4.22)}$$

Hard Constraint 3

Jumlah staff pada shift pagi pada hari Minggu harus terdiri dari empat. Dirumuskan untuk setiap nilai d jika nilai d adalah hari minggu.

$$\sum_{i=1}^{K} X_{id1} = 4 \qquad \textbf{(4.23)}$$

4.2.5.6. Divisi SIM & RM

Hard Constraint 1

Jumlah staff pada shift pagi setiap hari harus lebih dari sama dengan satu dan kurang dari sama dengan dua. Dirumuskan untuk setiap nilai d

$$\sum_{i=1}^{K} X_{id1} \geq 1 \quad \textbf{(4.24)}$$

$$\sum_{i=1}^{K} X_{id1} \leq 2 \quad \textbf{(4.25)}$$

Hard Constraint 2

Jumlah staff pada shift siang setiap hari harus sama dengan dua. Dirumuskan untuk setiap nilai d

$$\sum_{i=1}^{K} X_{id2} \geq 1 \quad \textbf{(4.26)}$$

$$\sum_{i=1}^{K} X_{id2} \leq 2 \quad \textbf{(4.27)}$$

Hard Constraint 3

Jumlah staff pada shift malam pada tiap hari harus terdiri dari satu. Dirumuskan untuk setiap nilai d

$$\sum_{i=1}^{K} X_{id3} = 1 \quad \textbf{(4.28)}$$

Hard Constraint 5

Dalam pembuatan jadwal ada peraturan untuk setiap pembuatan shift harus memenuhi pola shift Malam-Malam-Lepas Malam-Libur.

4.2.6. Pemodelan Soft Constraint

Soft Constraint 1

Nilai Optimality dari *Jain Fairness Index (JFI)*

$$JFI = \frac{\left(\sum_{i=1}^{n} f_i\right)^2}{\left(n * \sum (f_i)^2\right)} \quad \textbf{(4.29)}$$

$$f_i = \sum_{d=1}^{k} \sum_{t=1}^{z} |X_{idt} - 1|(4M_d + 2Sb_d + W_d) \quad \textbf{(4.30)}$$

Dimana

$$M_d \begin{cases} 1, \text{jika d pada hari Minggu} \\ 0, \text{jika tidak} \end{cases}$$

$$Sb_d \begin{cases} 1, \text{jika d pada hari Sabtu} \\ 0, \text{jika tidak} \end{cases}$$

$$W_d \begin{cases} 1, \text{jika d pada hari selain sabtu dan minggu} \\ 0, \text{jika tidak} \end{cases}$$

Soft Constraint 2

Jumlah libur tiap staf dalam 1 bulan sama dengan jumlah hari minggu pada 1 bulan

$$\sum_{i=1}^{K} X_{id10} = C \qquad \textbf{(4.31)}$$

$$C = \sum \text{hari Minggu dalam 1 bulan}$$

$$X_{id10} \begin{cases} 1, \text{jika staff } i \text{ pada hari ke} - \text{j mendapat shift "L"} \\ 0, \text{jika tidak} \end{cases}$$

4.2.7. Fungsi Tujuan

Fungsi Tujuan dari model yang akan dibuat adalah untuk memaksimalkan nilai dari dari *Jain Fairness Index* pada tiap divisi

$$MAX\ JFI = \frac{(\sum_{i=1}^{n} f_i)^2}{(n * \sum (f_i)^2)} \qquad \textbf{(4.32)}$$

$$f_i = \sum_{d=1}^{k} \sum_{t=1}^{z} |X_{idt} - 1|(4M_d + 2Sb_d + W_d) \qquad \textbf{(4.33)}$$

4.3. Penghitungan Jain Fairness Index

Untuk melakukan penghitungan *Jain Fairness Index* pada penjadwalan perawat di Rumah Sakit Ibu dan Anak Kendangsari MERR Surabaya merupakan sebuah metode yang digunakan untuk menanganipermasalahan keadilan yag ada di dalam pembagian shift kerja, pada studi kasus ini JFI (*Jain Fairness Index)* dibagi per divisi, jadi pada satu divisi memiliki JFI masing-masing. Untuk menghitung *Jain Fairness Index*

pada masing-masing divisi pertama akan diberikan bobot berdasarkan hari libur, karena berdasarkan asumsi jika para perawat akan lebih senang apabila libur pada akhir pekan maka jika perawat mendapatkan libur pada hari minggu akan mendapatkan bobot sebesar 4, jika libur pada hari sabtu maka akan mendapatkan bobot 2 dan jika libur pada hari kerja akan mendapatkan bobot 1. Sehingga dari ketentuan bobot tersebut akan ada fungsi tujuan dari JFI yaitu seperti pada *equation* nomor (**4.29)**

Setelah menentukan bobot yang ada maka setelah itu akan dimasukkan kedalam rumus berikut

$$JFI = \frac{\left(\sum_{i=1}^{n} f_i\right)^2}{\left(N * \sum (f_i)^2\right)} \qquad \textbf{(4.33)}$$

N = Jumlah Pegawai

Rumus diatas jika dihitung memiliki langkah-langkah sebagai berikut :

1) Kuadratkan jumlah hari libur * bobot selama satu bulan untuk semua staf $\left(\sum_{i=1}^{1} f_i\right)^2$
2) Setelah itu jumlahkan kuadrat dari masing-masing staff untuk hari libur * bobot lalu hasilnya dikalikan dengan jumlah pegawai $(N * \sum (fi)^2)$

4.4. Pemodelan Algoritma Optimasi *Self Adaptive Hyper Heuristic*

Pemodelan Algoritma Self Adaptive yang digunakan mengacu pada pseudocode pada penelitian oleh A. Muklason, "Hyper-Heuristics And Fairness In Examination Timetabling Problems," 2017 di **tabel 2.9** Dalam mengimplementasikan algoritma Self adaptive learning hyper heuristic dalam studi kasus ini adalah menetpakan fungsi tujuan jika dalam studi kasus ini fungsi tujuannya adalah memaksimalkan nilai dari *Jain Fairness Index* sehingga pada nantinya algoritma ini akan diterapkan pada masing-masing divisi. Setelah menetapkan

fungsi tujuan, yang selanjutnya dilakukan adalah menentukan *Low Level Heuristic*, dalam kasus ini yaitu :

H.1 : Mengganti shift "Middle Shift" menjadi libur pada salah satu karyawan dalam satu bulan (Move)

H.2 : Menukar shift karyawan yang memiliki shift libur dengan shift karyawan lain dalam 1(satu) hari (Swap)

Dalam memilih low level heuristic, performa baik ditentukan dari nilai JFI yang lebih besar.

Tahap-tahap algoritma :

1) Pertama dalam membuat algoritma ini tentukan dahulu jumlah iterasi yang akan dilakukan yaitu stopping condition (ex. 100)
2) Membuat array Queue dengan size tertentu untuk menyimpan low level heuristic (swap/move) yang akan diantrikan untuk dieksekusi
3) Membuat Linked List untuk menyimpan index dari *low level heuristic (swap/move)* yang memiliki performa baik
4) Deklarasikan variable untuk menyimpan hasil paling baik (bestSolution)
5) Buat iterasi untuk mendapatkan nilai random untuk memilih low level heuristic dan disimpan kedalam array
6) Lakukan iterasi untuk mengeksekusi setiap *low level heuristic* pada feasible solution dan hasil tersebut di simpan kedalam variable CurrentSolution lalu bandingkan dengan BestSolution.
7) Jika memiliki performa baik maka *low level heuristic* tersebut akan kembail masuk kedalam *Array Blocking Queue* untuk dieksekusi kembali, jika tidak memiliki performa baik maka tidak akan dimasukkan kembali
8) Jika queue telah habis maka akan diisi kembali secara random sampai jumlah iterasi tercukup

9) Setelah iterasi berakhir, ambil nilai terbaik dari seluruh iterasi yang telah dijalankan yaitu nilai yang di simpan di variable bestSolution

4.5. Pemodelan Algoritma Optimasi *Hill Climbing*

Algoritma Optimasi Hill Climbing diimplementasikan pada penelitian ini untuk dijadikan pembanding hasil dari algoritma *Self Adaptive Learning Hyper Heuristic*. Secara umum, algoritma hyper heuristic yang lain adalah pengembangan dari hill climbing, biasanya membuat seperangkat modifikasi pada candidat solusi yang diberikan dan modifikasi diterima apabila ada perbaikan atau memiliki nilai lebih baik daripada solusi sebelumnya [10]. Algoritma ini sama-sama memilih *Low Level Heuristic (LLH)* yang berperforma baik, tahap-tahap dari algoritma ini adalah :

1) Pertama dalam membuat algoritma ini tentukan dahulu jumlah iterasi yang akan dilakukan yaitu stopping condition (ex. 100)
2) Deklarasikan variable untuk menyimpan hasil paling baik (bestJFI) untuk inisiasi bestJFI adalah JFI yang didapat dari generate jadwal yang belum di optimasi.
3) Buat iterasi untuk mendapatkan nilai random untuk memilih low level heuristic dan langsung dieksekusi pada tiap iterasi
4) Setelah iterasi berakhir, ambil nilai terbaik dari seluruh iterasi yang telah dijalankan yaitu nilai yang di simpan di variable bestJFI.

(*halaman ini sengaja dikosongkan)*

BAB V
IMPLEMENTASI

Bab ini menjelaskan mengenai implementasi program untuk melakukan otomasi dan optimasi penjadwalan di RSIA Kendangsari Surabaya

5.1. Lingkungan Uji Coba

Uji coba untuk melakukan Optimasi menggunakan algoritma *Self Adaptive Learning Hyper Heuristic* untuk kasus penjadwalan perawat di Rumah Sakit Ibu dan Anak Kendangsari MERR Surabaya dilakukan menggunakan Bahasa pemrograman java dengan bantuan IDE Netbeans. Komputer yang digunakan adalah sebuah laptop dengan prosesor intel core i5 dengan kapasitas RAM sebesar 8 GB dan kapasitas Hard Disk sebesar 500 GB. Perangkat lunak yang digunakan dalam pengerjaan penelitian ini adalah system operasi Windows 10, dengan aplikasi IDE Netbeans, dan media penyimpanan data Microsoft Excel. Pada table 5.1 menampilkan perangkat keras dan lunak yang digunakan dalam pengerjaan penelitian tugas akhir ini.

Tabel 5.1 Lingkungan Uji Coba

Perangkat Keras	
Komputer	Laptop
Prosesor	Intel Core i5
RAM	8 GB
Hard Disk	500 GB
Perangkat Lunak	
Microsoft Windows 10	Operating System
Netbeans IDE	Aplikasi untuk pengembangan aplikasi dengan Bahasa pemrograman java
Microsoft Excel 2016	Aplikasi untuk melakukan penyimpanan data masukan dan rekap data mentah.

5.2. Otomasi Penjadwalan menggunakan Java

Setelah menentukan Batasan-batasan yang digunakan sebelumnya maka selanjutnya adalah melakukan otomasi untuk *generate* jadwal secara otomatis untuk bulan sebelumnya berdasarkan data yang ada, untuk data yang didapatkan kali ini adalah data per divisi pada bulan November 2017 sehingga pada otomasi kali ini akan dilakukan generate untuk bulan desember 2017 masing-masing per divisi. Pada sub bab ini akan dijelaskan tentang implementasi pemrograman java dalam melakukan otomasi dalam melakukan *generate* jadwal untuk bulan selanjutnya. Sebelum masuk kedalam masing-masing divisi, dibuat dahulu class "parentOtomasi" untuk dijadikan kelas *parent* dari inheritance dimana *inheritance* pada studi kasus ini adalah kelas dari masing-masing divisi. Berikut adalah list method yang ada di dalam kelas *parent* :

1) Method readcsv()
   Dalam method ini berisi kode untuk membaca file .csv yang dapat dijadikan acuan dalam membuat jadwal pada bulan selanjutnya, file .csv dibagi berdasarkan divisi.
2) Method tanggalhari()
   Dalam method ini berisi kode untuk mencetak array jadwal berupa hari dan tanggal
3) Method hitungJFI()
   Dalam method ini berisi kode untuk melakukan penghitungan *Jain Fairness Index*, masukan dari method ini adalah *array* dari jadwal yang telah dilakukan generate.
4) Method move()
   Dalam method ini berisi kode untuk melakukan salah satu *low level heuristic* yaitu move atau mengganti MS menjadi L secara random pada tiap-tiap pegawai
5) Method swap()
   Dalam method ini berisi kode untuk menjalankan salah satu *low level heuristic* yaitu swap atau menukar hari libur secara random antar karyawan dalam satu hari.

6) Method HillClimbing()
   Dalam method ini berisi kode untuk menjalankan algoritma hill climbing untuk melakukan move dan swap sampai mendapatkan nilai *Jain Fairness Index* yang terbaik, yaitu dengan melakukan iterasi sebanyak n kali dengan memasukkan secara random move dan swap jika hasil lebih baik maka akan dipilih hasil tersebut dan berlanjut sampai iterasi selesai. input dari method ini adalah jumlah iterasi yang diinginkan.
7) Method SelfAdaptive()
   Dalam method ini berisi kode untuk menjalankan algoritma Self Adaptive Learning untuk move dan swap sampai mendapatkan nilai *Jain Fairness Index* yang terbaik, yaitu dengan melakukan iterasi sebanyak n kali lalu menggunakan *Array Blocking Queue* untuk menampung antrian eksekusi dengan jumlah tertentu lalu akan dibandingkan, apabila lebih baik daripada current solution maka akan masuk kembali kedalam *queue* jika tidak lebih baik maka tidak dimasukkan kedalam queue untuk dieksekusi.

5.2.1. Otomasi Divisi Farmasi

Cara penjadwalan pada divisi farmasi :

- Kepala Unit dan Petugas Gudang selalu memiliki shift pagi dari hari Senin-Sabtu dan libur setiap hari Minggu.
- Untuk staf no 2 memiliki pola Shift Siang,
- Lihat pola M-M-LM-L yang belum lengkap pada bulan sebelumnya
- Isikan pola M-M-LM-L melanjutkan pola bulan sebelumnya untuk Staff 3-7
- Setiap hari harus memiliki paling tidak 1 orang shift malam sehingga jika salah satu staf sudah shift LM maka harus ada staf yang lain shift M
- Untuk staf no 3 sampai 7 memiliki pola sehabis shift libur adalah P-P-S-S

- Untuk staf no 3 sampai 7 sebelum mendapatkan shift malam harus mendapatkan shift siang
- Dalam satu hari shift pagi harus terdiri dari 3, shift siang terdiri dari 2
- Khusus hari minggu P=1 S=2 M=1 tidak ada middle shift
- Jika jumlah Libur, Cuti, dan Lepas malam kurang dari sama dengan satu maka shift sisa merupakan shift middle.

Setelah mengetahui pola dan cara untuk membuat jadwal secara manual, setelah itu baru akan dilakukan *coding* untuk melakukan penjadwalan secara otomatis, berikut ini adalah source code dan penjelasan serta hasil dari *coding*.

```
1.    public GenerateFarmasi1(int NumberOfEmployee, String  csvFile) {
2.      super(NumberOfEmployee, csvFile);
3.    }
4.    public String[][] initiate() {
5.    //STEP 1 bikin baris pertama dan terakhir P
6.    for (int i = NumberOfDays; i <NumberOfDays+ LengthOfMonth[newMonths] ; i++) {
7.      roster[0][i] = "P";
8.      roster[NumberOfEmployee-1][i] = "P";
9.      if((i+FirstDay)%7 == 0){
10.     roster[0][i]= "L";
11.     roster[NumberOfEmployee-1][i] = "L";
12.   }
```

Kode Program 5.1 Otomasi Divisi Farmasi 1

1) Kode program diatas adalah step pertama untuk menjadwalkan staff dengan indeks 0, yaitu kepala unit dengan menjadwalkan akan masuk setiap hari di shift pagi dan libur pada hari minggu

```
1.  //STEP 2 bikin staff no2 spesialis siang
2.    for (int i = NumberOfDays; i <NumberOfDays +LengthOfMonth[newMonths] ; i++) {
3.      roster[1][i] = "S";
4.      if((i+FirstDay)%7 == 0){
5.       roster[1][i]= "L";
6.      }
7.    }
```

Kode Program 5.2 Otomasi Divisi Farmasi 2

2) Kode Program diatas untuk menjadwalkan staff nomor 2 atau pada indeks no.1 untuk mendapatkan shift siang setiap hari dan libur default hari minggu.

```
1.  //STEP 3 memasukkan pola M M LM L P P S S untuk employee no 3
2.          String Pattern []={"M", "M", "LM","L","P","P","S","S","-1", "-1"};
3.          for (int i = 0; i < NumbE; i++) {
4.              for (int j = NumberOfDays; j < NumberOfDays+LengthOfMonth[newMonths]; j++)
    {
5.                      roster[i+2][j]= Pattern[(j-StartingPoint)%10];
6.
7.              }
8.              StartingPoint +=2;
9.          }
```

Kode Program 5.3 Otomasi Divisi Farmasi 3

3) Kode Program diatas adalah kode yang berfungsi untuk mengisi pola M-M-LM-L serta mengisi P-P-S-S untuk 5 perawat

```
1.  // STEP 4 : menetukan midle shift
2.          String NamaShift[]={"P","S","M","LM","L","CT","MS"};
3.              for (int j = NumberOfDays; j <NumberOfDays+LengthOfMonth[newMonths]; j++) {

4.                  int JmlShift [] ={0,0,0,0,0,0,0};
5.                  for (int i = 0 ; i <NumberOfEmployee; i++) {
6.                      for (int k = 0; k < JmlShift.length; k++) {
7.                          if(roster[i][j].equalsIgnoreCase(NamaShift[k])) {
8.                              JmlShift[k] ++;
9.                          }
10.                     }
11.                     if(roster[i][j]== "-1"){
12.                         if((JmlShift[3]+JmlShift[4]+JmlShift[5])<= 1){
13.                             roster[i][j]="MS";
14.                         }else{
15.                             roster[i][j]="S";
16.                         }
17.                     }
18.                 }
19.             }
20.         return this.roster;
21.     }
```

Kode Program 5.4 Otomasi Divisi Farmasi 4

4) Pada tahap yang ke-enam adalah kode yang berfungsi untuk mengisi middle shift dimana pada setiap hari jika LM + CT + L <=1 maka akan diisi middle shift pada hari tersebut.

5.2.2. Otomasi Divisi IGD

Cara penjadwalan pada divisi IGD :

- Masukkan pola M M LM L berdasarkan pola bulan sebelumnya
- Setiap minggu tiap perawat dapat 1 kali shift pagi, 3 kali shift sore, 1 Libur, sisanya middle shift
- Masukkan pola “M,M,LM,L,P,P,S,S,S,S,L,MS,MD,S,S,S”

Setelah mengetahui pola dan cara untuk membuat jadwal secara manual, setelah itu baru akan dilakukan *coding* untuk melakukan penjadwalan secara otomatis, berikut ini adalah source code dan penjelasan serta hasil dari *coding*

Penjelasan Kode Program:

```
1.  public class GenerateIGD extends parentOtomasi{
2.          public GenerateIGD(int NumberOfEmployee, String csvFile){
3.              super(NumberOfEmployee, csvFile);
4.          }
5.
6.      public String[][] initiateIGD() {
7.          int StartingPoint=0;
8.          int NumbE= 8;
9.          String Pattern []={"M", "M", "LM","L","P","P","S","S","S","S","S","L","MD","MS"
    ,"S","S","S"};
10.         for (int i = 0; i < NumbE; i++) {
11.             for (int j = NumberOfDays; j < NumberOfDays+LengthOfMonth[newMonths]; j++)
    {
12.                 roster[i][j]= Pattern[(j-StartingPoint)%15];
13.             }
14.             StartingPoint +=2;
15.         }
16.         return this.roster;
17.     }
```

Kode Program 5.5 Otomasi Divisi IGD

1) Pada divisi IGD yang berbeda dari farmasi adalah pola yang digunakan untuk memenuhi constraint yang ada sehingga menggunakan pola seperti diatas yang diulang ke semua anggota perawat yang ada di IGD.

5.2.3. Otomasi Divisi NICU dan Bayi

Cara penjadwalan pada divisi NICU dan Bayi :

- Isikan Pola M-M-LM-L pada setiap anggotanya, kecuali pada jadwal kepala anggota, karena kepala anggota yang hanya memiliki pola P dan L dalam satu minggunya, setiap minggunya kepala anggota libur pada hari minggu dan memiliki shift pagi dari hari senin sampai sabtu
- Selanjutnya untuk anggota kedua sampai anggota ke tiga belas isikan pola {“P”,“P”,“S”,“S”,“Ms”,“Ms”} setelah mengisi pola M-M-LM-L
- Isi juga secara menurun geser dua kotak kekanan secara berulang

Setelah mengetahui pola dan cara untuk membuat jadwal secara manual, setelah itu baru akan dilakukan coding untuk melakukan penjadwalan secara otomatis, berikut ini adalah source code dan penjelasan serta hasil dari coding.

Penjelasan Kode Program:

```
1.  public class GenerateBayiNICU extends parentOtomasi {
2.
3.      public GenerateBayiNICU(int NumberOfEmployee, String csvFile){
4.           super(NumberOfEmployee, csvFile);
5.          }
6.
7.       public String[][] Generate() {
8.          //STEP 1 bikin baris pertama P
9.          for (int i = NumberOfDays; i <NumberOfDays+LengthOfMonth[newMonths] ; i++) {
10.              roster[0][i] = "P";
11.              if((i+FirstDay)%7 == 0){
12.                  roster[0][i]= "L";
13.              }
14.             }
```

Kode Program 5.6 Otomasi Divisi NICU & Bayi

1) Untuk bagian pertama melakukan perulangan untuk mengisi baris pertama yaitu kepala perawat sehingga akan mendapatkan shift pagi dan mendapatkan libur pada hari minggu

```
1.  //STEP 2 memasukkan pola M M LM L P P S S untuk employee no 3
2.          int StartingPoint=0;
3.          int NumbE= 12;
4.          String Pattern []={"M", "M", "LM","L","P","P","S","S","MS", "MS"};
5.          for (int i = 0; i < NumbE; i++) {
6.              for (int j = NumberOfDays; j < NumberOfDays+LengthOfMonth[newMonths]; j++) {

7.                  roster[i+1][j]= Pattern[(j-StartingPoint)%10];
8.              }
9.              StartingPoint +=2;
10.         }
11.
12.             return this.roster;
13.     }
```

Kode Program 5.7 Otomasi Divisi Bayi NICU

2) Untuk bagian kedua yaitu melakukan perulangan untuk memasukkan pola M-M-LM-L dengan menambahkan P P S S MS MS sehingga jika dilakukan perulangan sampai ke staff selanjutnya akan memenuhi constraint.

5.2.4. Otomasi Divisi Gizi

Cara penjadwalan pada divisi Gizi :

- Ahli Gizi selalu memiliki shift pagi dari hari Senin-Sabtu dan libur setiap hari Minggu.
- Untuk Staf 2-5 (Chef), siskan pola P1,P,S,P1,P,L,S,S,MS,MS. Cara mengisi jarak 10 kotak dengan startingpoint +2
- Untuk Staf 6,8,9 (HELPER),siskan pola P2,P,S,S,L,P,S,P2,P,MS. Cara mengisi jarak 10 kotak dengan startingpoint +2
- Untuk staf nomer 7 isikn pola, S,S,S,P1,P1,P1,L dengan jarak antar pola 7
- Staff 10 nomor 5 S,S,P,P,MD,MD,L dengan jarak 7
- Lihat pola M-M-LM-L yang belum lengkap pada bulan sebelumnya untuk Staff 11-15
- Isikan pola M-M-LM-L melanjutkan pola bulan sebelumnya untuk Staff 11-15
- Setiap hari harus memiliki paling tidak 1 orang shift malam sehingga jika salah satu staf sudah shift LM maka harus ada staf yang lain shift M
- Untuk staf no 11-15 memiliki pola sehabis shift libur adalah P-P-S-S
- Untuk staf 10.-15 jika array= null, jika kolom terdapat MD maka isi P, dan jika tidak ada MD maka isi MD
- Alokasikan request Libur dan Cuti,
- Jika jumlah Libur, Cuti, dan Lepas malam kurang dari sama dengan satu maka shift sisa merupakan shift middle.
- Untuk staff 16 dan 19 memiliki pola "PS","S","S","P","P","P","PS","L","P","P","S","S","S", "L"
- Untuk staff 17 dan 18 memiliki pola "L","P","P","S","S","S","L","PS","S","S","P","P","P", "PS"

Setelah mengetahui pola dan cara untuk membuat jadwal secara manual, setelah itu baru akan dilakukan coding untuk melakukan penjadwalan secara otomatis, berikut ini adalah source code dan penjelasan serta hasil dari coding.

Penjelasan Kode Program:

```
1.  public class GenerateGizi extends parentOtomasi {
2.          public GenerateGizi(int NumberOfEmployee, String csvFile){
3.              super(NumberOfEmployee, csvFile);
4.          }
5.      public String[][] initiate() {
6.          //STEP 1 bikin employee pertama P
7.          for (int i = NumberOfDays; i <NumberOfDays+LengthOfMonth[newMonths] ; i++) {
8.              roster[0][i] = "P";
9.              roster[NumberOfEmployee-1][i] = "P";
10.             if((i+FirstDay)%7 == 0){   //minggu
11.                 roster[0][i]= "L";     //maka libur
12.                 roster[NumberOfEmployee-1][i] = "L";
13.             }
14.            }
```

Kode Program 5.8 Otomasi Divisi Gizi 1

1) Pada tahap yang pertama, kode program tersebut digunakan untuk menjadwalkan kepala unit yang berada di baris ke satu dengan shift pagi dan libur i=di hari minggu

```
1.  //STEP 2 memasukkan pola P1,P,S,P1,P,L,S,S,MS,MS untuk employee no 2-5
2.  int StartingPointChef=0;
3.  int NumbChef= 4;
4.  String PatternChef [] = {"P1","P","S","P1","P","L","S","S","MD","MD"};
5.  for (int i = 0; i < NumbChef; i++) {
6.      for (int j = NumberOfDays; j < NumberOfDays+LengthOfMonth[newMonths]; j++) {
7.          roster[i+1][j]= PatternChef[(j-StartingPointChef)%10];
8.      }
9.      StartingPointChef += 2;
10. }
```

Kode Program 5.9 Otomasi Divisi Gizi 2

2) Pada tahap yang kedua, kode program tersebut digunakan untuk melakukan penjadwalan karyawan no 2-5

```
1.  //STEP 3 memasukkan pola L P P S S S P P untuk employee no 6 (Riza Johan)
2.          String Pattern7 [] = {"L","P1","P1","S","S","S","P1","P1"};
3.              for (int j = NumberOfDays; j < NumberOfDays+LengthOfMonth[newMonths]; j++)
    {
4.                  roster[5][j]= Pattern7[(j+5)%8];
5.              }
```

Kode Program 5.10 Otomasi Divisi Gizi 3

3) Pada tahap yang ketiga, kode program tersebut digunakan untuk melakukan penjadwalan dengan pola tersebut untuk pegawai nomor 6

```
1.  //STEP 4 memasukkan pola untuk employee no 7-9
2.          int StartingPointHelper=0;
3.          int NumbHelper = 3;
4.          String PatternHelper [] = {"P2","P","S","S","L","P","S","P2","P","MS"};
5.          for (int i = 0; i < NumbHelper; i++) {
6.              for (int j = NumberOfDays; j < NumberOfDays+LengthOfMonth[newMonths]; j++)
    {
7.                  roster[i+6][j]= PatternHelper[(j-StartingPointHelper)%10];
8.              }
9.              StartingPointHelper += 2;
10.         }
```

Kode Program 5.11 Otomasi Divisi Gizi 4

4) Pada tahap yang keempat, kode program tersebut digunakan untuk melakukan penjadwalan dengan pola tersebut untuk pegawai nomor 7-9

```
1.  //STEP 5 memasukkan pola L P P S S S P P untuk employee no 10
2.              String Pattern10 [] = {"S","S","P","P","MD","MD","L"};
3.                  for (int j = NumberOfDays; j < NumberOfDays+LengthOfMonth[newMonths]; j+
    +) {
4.                      roster[9][j]= Pattern10[(j+5)%7];
5.                  }
```

Kode Program 5.12 Otomasi Divisi Gizi 5

5) Pada tahap yang kelima, kode program tersebut digunakan untuk melakukan penjadwalan dengan pola tersebut untuk pegawai nomor 10

```
//STEP 6 memasukkan pola M M LM L P P S S untuk employee no 11-15
        int StartingPointPenyaji=0;
        int NumbE= 5;
        String Pattern [] = {"M","M","LM","L","P","P","S","S","-1","-1"};
        for (int i = 0; i < NumbE; i++) {
            for (int j = NumberOfDays; j < NumberOfDays+LengthOfMonth[newMonths]; j++)
    {
                roster[i+10][j]= Pattern[(j-StartingPointPenyaji)%10];
            }
            StartingPointPenyaji += 2;
        }
```

Kode Program 5.13 Otomasi Divisi Gizi 6

6) Pada tahap yang keenam, kode program tersebut digunakan untuk melakukan penjadwalan dengan pola tersebut untuk pegawai nomor 10

```
//STEP 7 BIKIN JADWAL STAFF 16 DAN 19
        for (int i = NumberOfDays; i <NumberOfDays+LengthOfMonth[newMonths] ; i++) {
            String Pattern1619 [] = {"PS","S","S","P","P","P","PS","L","P","P","S","S",
    "S","L"};
            for (int j = NumberOfDays; j < NumberOfDays+LengthOfMonth[newMonths]; j++)
    {
                roster[15][j]= Pattern1619[(j+9)%14];
                roster[18][j]= Pattern1619[(j+9)%14];
            }
        }
```

Kode Program 5.14 Otomasi Divisi Gizi 7

7) Pada tahap yang ketujuh, kode program tersebut digunakan untuk melakukan penjadwalan dengan pola tersebut untuk pegawai nomor 16 dan 19

```
//STEP 8 BIKIN JADWAL STAFF 17 DAN 18
        for (int i = NumberOfDays; i <NumberOfDays+LengthOfMonth[newMonths] ; i++) {
            String Pattern1718 [] = {"L","P","P","S","S","S","L","PS","S","S","P","P","P
    ","PS"};
            for (int j = NumberOfDays; j < NumberOfDays+LengthOfMonth[newMonths]; j++) {

                roster[16][j]= Pattern1718[(j+9)%14];
                roster[17][j]= Pattern1718[(j+9)%14];
            }
        }
```

Kode Program 5.15 Otomasi Divisi Gizi 8

8) Pada tahap yang kedelapan, kode program tersebut digunakan untuk melakukan penjadwalan dengan pola tersebut untuk pegawai nomor 17 dan 18

```
// STEP 9 : menentukan midle shift
          String NamaShift[]={"P","S","M","LM","L","CT","MD"};
            for (int j = NumberOfDays; j <NumberOfDays+LengthOfMonth[newMonths]; j++) {

                int JmlShift [] ={0,0,0,0,0,0,0};
                for (int i = 0 ; i <NumberOfEmployee; i++) {
                    for (int k = 0; k < JmlShift.length; k++) {
                        if(roster[i][j].equalsIgnoreCase(NamaShift[k])) {
                            JmlShift[k] ++;
                        }
                    }
                    if(roster[i][j] == "-1"){
                        if(JmlShift[6] == 0){
                            roster[i][j]="MD";
                        }else{
                            roster[i][j]="P";
                        }
                    }
                }
            }

        return this.roster;
    }
```

Kode Program 5.16 Otomasi Divisi Gizi 9

9) Pada tahap yang ke-sembilan adalah kode yang berfungsi untuk mengisi middle shift dimana pada setiap hari jika CT = 0 maka akan diisi middle shift pada hari tersebut, jika tidak maka akan diisi P.

5.2.5. Otomasi Divisi OK

Cara Penjadwalan pada divisi OK :

- Isikan pola M-M-LM-L mengikuti pola sebelumnya pada setiap anggota.
- Pengisian pola M-M-LM-Lmengikuti aturan setiap shift malam hanya berisi 1 staff. Bila hari tersebut sudah ada orang yang mendapat shift malam, maka staff lain tidak boleh mendapat shift malam.
- Pengisian pola M-M-LM-L diikuti dengan pola P-P-P-L-S-S-S-(-1) di bagian belakangnya.
- Mengisi pola yang kosong dengan ketentuan apabila dalam satu hari jumlah P, S. dan M-nya kurang dari sama dengan 3 maka diisi dengan “M”. Apabila lebih dari 3 maka diisi P

Setelah mengetahui pola dan cara untuk membuat jadwal secara manual, setelah itu baru akan dilakukan coding untuk melakukan penjadwalan secara otomatis, berikut ini adalah source code dan penjelasan serta hasil dari coding.

Penjelasan Kode Program:

```
1.  //STEP 1 mengisi Pola berdderet
2.          int StartingPoint=0;
3.          String Pattern []={"P","P","S","P","S","L","P","P","S","-1"};
4.          for (int i = 0; i < NumberOfEmployee; i++) {
5.              for (int j = NumberOfDays; j < NumberOfDays+LengthOfMonth[newMonths]; j++)
    {
6.                  roster[i][j]= Pattern[(j-StartingPoint)%10];
7.              }
8.              StartingPoint ++;
9.          }
10.
```

Kode Program 5.17 Otomasi Divisi OK 1

1) Pada tahap yang pertama adalah kode program yang berfungsi untuk mengisi pola berderet dengan melakukan perulangan dengan pola P-P-S-P-S-L-P-P-S-(-1)/null perulangan dilakukan untuk staff selanjutnya bergeser 1 kolom sehingga starting point +1

```
1.  //Step 2 Mengisi Kolom kosong
2.      String NamaShift[]={"S","P","L","CT"};
3.          for (int j = NumberOfDays; j <NumberOfDays+LengthOfMonth[newMonths]; j++) {
4.              int JmlShift [] ={0,0,0,0};
5.              for (int i = 0 ; i <NumberOfEmployee; i++) {
6.                  for (int k = 0; k < JmlShift.length; k++) {
7.                      if(roster[i][j].equalsIgnoreCase(NamaShift[k])) {
8.                          JmlShift[k] ++;
9.                      }
10.                 }
11.                 if(roster[i][j]== "-1"){
12.                     if(JmlShift[0]<1){
13.                         roster[i][j]= "S";
14.                     }else{
15.                         roster[i][j]="P";
16.                     }
17.                 }
18.             }
19.         }
```

Kode Program 5.18 Otomasi Divisi OK 2

2) Pada tahap yang kedua adalah kode untuk mengisi yang masih kosong/null dimana memilki syarat pada satu hari jika dalam satu hari memiliki P < 1 maka akan diisi P, jika tidak maka akan diisi S.

```
1.  //STEP 3.A MENGISI P SEBANYAK 4
2.        int StartingPointLibur=0;
3.        String pola2 []={"L","L","P","P","P","P"};
4.        for (int j = NumberOfDays; j < NumberOfDays+LengthOfMonth[newMonths]; j++) {
5.          for (int k =0; k <pola2.length; k++) {
6.              if((j+FirstDay)%7 == 0 ){
7.                  roster[k][j]=pola2[(k+StartingPointLibur+2)%6];
8.              }
9.          }StartingPointLibur +=2;
10.       }
11.     return this.roster;
```

Kode Program 5.19 Otomasi Divisi OK 2

3) Pada tahap yang ketiga adalah melihat apabila pada hari indeks ke 6 yaitu hari minggu akan diisi dengan pola kolom L-L-P-P-P .

5.2.6. Otomasi Divisi SIM & RM

Cara Penjadwalan pada divisi SIM dan RM :

- Isikan pola M-M-LM-L mengikuti pola sebelumnya pada setiap anggota.
- Pengisian pola M-M-LM-Lmengikuti aturan setiap shift malam hanya berisi 1 staff. Bila hari tersebut sudah ada orang yang mendapat shift malam, maka staff lain tidak boleh mendapat shift malam.
- Pengisian pola M-M-LM-L diikuti dengan pola P-P-P-L-S-S-S-(-1) di bagian belakangnya.
- Mengisi pola yang kosong dengan ketentuan apabila dalam satu hari jumlah P, S. dan M-nya kurang dari sama dengan 3 maka diisi dengan “M”. Apabila lebih dari 3 maka diisi P

Setelah mengetahui pola dan cara untuk membuat jadwal secara manual, setelah itu baru akan dilakukan coding untuk melakukan penjadwalan secara otomatis, berikut ini adalah source code dan penjelasan serta hasil dari coding.
Penjelasan Kode Program:

```
1.  public class GenerateSIMRM extends parentOtomasi {
2.       public GenerateSIMRM(int NumberOfEmployee, String csvFile) {
3.          super(NumberOfEmployee, csvFile);
4.      }
5.
6.      public String[][] GenerateSIMRM () {
7.
8.          //STEP 1 memasukkan pola M M LM L P P P L S S S untuk semua employee
9.          int StartingPoint=0;
10.         String Pattern []={"M","M","LM","L","P","P","P","L","S","S","-1","-1"};
11.         for (int i = 0; i < NumberOfEmployee; i++) {
12.             for (int j = NumberOfDays; j < NumberOfDays+LengthOfMonth[newMonths]; j++)
    {
13.                 roster[i][j]= Pattern[(j-StartingPoint)%12];
14.             }
15.             StartingPoint +=2;
16.         }
```

Kode Program 5.20 Otomasi Divisi SIM & RM 1

1) Pada tahap yang pertama adalah kode program yang berfungsi untuk melakukan perulangan untuk memasukkan pola M-M-LM-L-P-P-P-S-S-S-(-1)/null dan akan berulang pada staff berikutnya bergeser 2 kolom untuk memenuhi hard constraint dalam setiap hari setidaknya ada 1 orang shift malam.

```
1.  // STEP 2 : menentukan midle shift
2.       String NamaShift[]={"P","S","M","LM","L","CT","MS"};
3.          for (int j = NumberOfDays; j <NumberOfDays+LengthOfMonth[newMonths]; j++) {

4.              int JmlShift [] = {0,0,0,0,0,0,0};
5.              for (int i = 0 ; i < NumberOfEmployee; i++) {
6.                  for (int k = 0; k < JmlShift.length; k++) {
7.                      if(roster[i][j].equalsIgnoreCase(NamaShift[k])) {
8.                          JmlShift[k] ++;
9.                      }
10.                 }
11.                 if(roster[i][j]== "-1"){
12.                     if((JmlShift[0]+JmlShift[1]+JmlShift[2]) <= 3){
13.                         roster[i][j]="MS";
14.                     }else {
15.                         roster[i][j]="P";
16.                        // roster[i][j]="S";
17.                     }
18.                 }
19.             }
20.
21.         }
22.
23.     return this.roster;
24. }
```

Kode Program 5.21 Otomasi Divisi SIM & RM 2

2) Pada tahap yang kedua adalah kode yang berfungsi untuk mengisi middle shift dimana pada setiap hari jika jumlah P+ S + M kurang dari atau samadengan 3 maka

maka akan diisi middle shift pada hari tersebut, jika tidak maka akan diisi P.

5.2.7. Otomasi Perhitungan Jain Fairness Index

Setelah diketahui cara menghitung *Jain Fairness Index* (JFI) secara manual, maka dilakukan *coding* untuk menghitung JFI secara otomatis berdasarkan jadwal yang di *generate* secara otomatis.

Penjelasan Kode Program :

```
1.    public double hitungJFI(String [][] roster){
2.       double jfi= 0;
3.       double sum=0;
4.       double JmlhJfiStafkudrat=0;
5.       double JmlSelStaf=0;
6.       double JmlSelStaf_kuadrat=0;
7.        for (int i = 0 ; i <NumberOfEmployee; i++) {
8.         int a=0;
9.         int JmlLibur[]= {0,0,0};
10.        int Bobot[]={4,2,1};
11.        int liburXbobot= 0;
12.        int Jml_liburXbobot=0;
13.        double JFIperStaf_Kuadrat= 0;
14.       for (int j = NumberOfDays; j <NumberOfDays+ LengthOfMonth[newMonths]; j++) {
15.        if(roster[i][j]. equalsIgnoreCase("L")){
16.          if((j+FirstDay)%7 == 0){
17.          JmlLibur[0]++;
18.          }else if ((j+FirstDay)%7 == 6){
19.          JmlLibur[1]++;
20.          }else{
21.          JmlLibur[2]++;
22.          }
23.         }
24.       }
```

Kode Program 5.22 Otomasi Penghitungan Jain Fairness Index 1

1) Pada tahap pertama adalah inisiasi tipe data yang akan digunakan dalam penghitungan dan Melakukan pembuatan array untuk menyimpan jumlah bobot yang akan digunakan dimana bobot untuk libur hari minggu = 4, Sabtu = 2, Hari Kerja = 1

```
1.          for (int k = 0; k < JmlLibur.length; k++) {
2.         int bb= JmlLibur[k];
3.         liburXbobot =JmlLibur[k]*Bobot[k];
4.         //jumlah dengan pembobotan
5.        Jml_liburXbobot +=liburXbobot; // hitung per staf
6.         //hitung masing masing staf
7.       }
8.       JFIperStaf_Kuadrat =  Math.pow(Jml_libur Xbobot, 2);//benar kuadrat masing masing s
   taf
9.      JmlhJfiStafkudrat +=JFIperStaf_Kuadrat;
10.    JmlSelStaf +=Jml_liburXbobot;
11.     }
```

Kode Program 5.23 Otomasi Penghitungan Jain Fairness Index 2

2) Melakukan perulangan untuk menghitung jumlah * bobot pada tiap staff dan dikuadrat lalu dijumlahkan

```
1.    JmlSelStaf_kuadrat = Math.pow(JmlSelStaf, 2);
2.    jfi=(JmlSelStaf_kuadrat /(NumberOfEmployee* JmlhJfiStafkudrat));
3.     return jfi; 4:
4.  }
```

Kode Program 5.24 Otomasi Penghitungan Jain Fairness Index 3

3) Melakukan kuadrat untuk penghitungan no.3 lalu dibagi dengan jumlah staff kuadrat * jumlah staf

5.3. Otomasi Pengecekan Hard Constraint

Setelah melakukan otomasi untuk generate jadwal secara otomatis, langkah selanjutnya adalah melakukan otomasi untuk mengecek apakah hard constraint yang ada telah terpenuhi, dikarenakan masing-masing divisi memiliki hard constraint yang berbeda-beda sehingga untuk otomasi dilakukan di masing-masing class divisi

5.3.1. Otomasi Pengecekan Constraint Divisi Farmasi

```
1.  public boolean checkConstraintFarmasi(String[][] roster){
2.        boolean value1= false;
3.        boolean value2= false;
4.
5.        for (int j = NumberOfDays; j <NumberOfDays+LengthOfMonth[newMonths]; j++) {
6.            String [] Shift ={"P","S","M"};
7.            int [] Jml_Shift = {0,0,0};
8.            for (int i = 0; i < NumberOfEmployee; i++) {
9.                if((j+FirstDay)%7 == 0){
10.                   if(roster[i][j].equalsIgnoreCase("P")){
11.                         Jml_Shift[0]++;
12.                     } else if(roster[i][j].equalsIgnoreCase("S")){
13.                         Jml_Shift[1]++;
14.                     } else if(roster[i][j].equalsIgnoreCase("M")){
15.                         Jml_Shift[2]++;
16.                     }
17.
18.                   if ((Jml_Shift[0] == 1 && Jml_Shift[1]== 2 )&& Jml_Shift[2]== 1){
19.                       value1 = true;
20.                   }else{
21.                       value1 = false;
22.                   }
23.               } else{
24.                    if(roster[i][j].equalsIgnoreCase("P")){
25.                         Jml_Shift[0]++;
26.                     } else if(roster[i][j].equalsIgnoreCase("S")){
27.                         Jml_Shift[1]++;
28.                     } else if(roster[i][j].equalsIgnoreCase("M")){
29.                         Jml_Shift[2]++;
30.                     }
31.                   if (Jml_Shift[0]== 3 && Jml_Shift[1]== 2 && Jml_Shift[2]== 1){
32.                       value2 = true;
33.                   } else{
34.                       value2 = false;
35.                   }
36.               }
37.            }
38.       }
39.
40.       System.out.println("Cek Constraint");
41.       System.out.println("Nilai value1 : " + value1);
42.       System.out.println("Nilai value2 : " + value2);
43.
44.       if (value1=value2) {
45.           value1=true;
46.       } else {
47.           value1=false;
48.       }
49.
50.    return value1;
51.    }
```

Kode Program 5.25 Otomasi Pengecekan Constraint Divisi Farmasi

Dalam pengecekan constraint pada divisi farmasi dijadikan kedalam satu method, karena didalamnya hanya variable jumlah hari yang menjadi Batasan sehingga bisa dijalankan dengan mengambil nilai berapa jumlah P, S, dan M, dalam satu

hari, lalu jika kondisi memenuhi maka akan mengeluarkan value true, dan apabila tidak maka akan mengeluarkan false, dan karena ada 2 jenis Batasan, yaitu di hari kerja dan di hari minggu maka dibedakan kebutuhannya, sehingga akan dikeluarkan dua nilai balikan yaitu value1 dan value2.

5.3.2. Otomasi Pengecekan Constraint Divisi IGD

```
1.  public boolean checkConstraintIGD(String[][] roster){
2.          boolean value1= false;
3.          boolean value2= false;
4.
5.          for (int j = NumberOfDays; j <NumberOfDays+LengthOfMonth[newMonths]; j++) {
6.              String [] Shift ={"P","S","M"};
7.              int [] Jml_Shift = {0,0,0};
8.              for (int i = 0; i < NumberOfEmployee; i++) {
9.                  if((j+FirstDay)%7 == 0){
10.                     if(roster[i][j].equalsIgnoreCase("P")){
11.                         Jml_Shift[0]++;
12.                       } else if(roster[i][j].equalsIgnoreCase("S")){
13.                         Jml_Shift[1]++;
14.                       } else if(roster[i][j].equalsIgnoreCase("M")){
15.                         Jml_Shift[2]++;
16.                       }
17.                     if ((((Jml_Shift[0] >= 1)||(Jml_Shift[0] <= 2)) &&
18.                             ((Jml_Shift[1] >= 2) || (Jml_Shift[1] <= 4)) &&
19.                             ((Jml_Shift[2] >= 1)||(Jml_Shift[2] <= 2)))){
20.                         value1 = true;
21.                     }else{
22.                         value1 = false;
23.                     }
24.                 }
25.             }
26.          }
27.         System.out.println("Cek Constraint");
28.         System.out.println("Nilai value1 : " + value1);
29.
30.         System.out.println(value1);
31.         return value1;
32.     }
```

Kode Program 5.26 Pengecekan Constraint Divisi IGD

Dalam pengecekan constraint pada divisi IGD dijadikan kedalam satu method, karena didalamnya hanya variable jumlah hari yang menjadi batasan sehingga bisa dijalankan dengan mengambil nilai berapa jumlah P, S, dan M, dalam satu hari, lalu jika kondisi memenuhi maka akan mengeluarkan value true, dan apabila tidak maka akan mengeluarkan false, karena divisi IG D hanya memiliki 1 jenis constraint pada hari biasa yaitu P $\leq$2, 4$\leq$ S $\geq$2, dan M $\leq$ 2 maka hanya ada 1 nilai balikan yaitu value1.

5.3.3. Otomasi Pengecekan Constraint Divisi NICU & Bayi

```
1.  public boolean checkConstraintBayi(String [][] roster){
2.          boolean value1 = false;
3.          boolean value2 = false;
4.          for (int j = NumberOfDays; j <NumberOfDays+LengthOfMonth[newMonths]; j++) {
5.              String [] Shift ={"P","S", "M"};
6.              int [] Jml_Shift = {0,0,0};
7.              for (int i = 0; i < NumberOfEmployee; i++) {
8.                      if(roster[i][j].equalsIgnoreCase("P")){
9.                            Jml_Shift[0]++;
10.                       } else if(roster[i][j].equalsIgnoreCase("S")){
11.                           Jml_Shift[1]++;
12.                       } else if(roster[i][j].equalsIgnoreCase("M")){
13.                           Jml_Shift[2]++;
14.
15.                     if (((Jml_Shift[0] >= 2) || (Jml_Shift[0] <= 4)) &&
16.                             ((Jml_Shift[1] >= 1) || (Jml_Shift[1] <= 2))
17.                             &&((Jml_Shift[2] >= 2) || (Jml_Shift[2] <= 3))){
18.                         value1 = true;
19.                     }else{
20.                         value1 = false;
21.                     }
22.                 }
23.              }
24.         }
25.         System.out.println("Cek Constraint");
26.         System.out.println("Nilai value1 : " + value1);
27.     return value1;
28.    }
```

Kode Program 5.27 Otomasi Pengecekan Constraint Divisi NICU & Bayi

Dalam pengecekan constraint pada divisi nicu dijadikan kedalam satu method, karena didalamnya hanya variable jumlah hari yang menjadi Batasan sehingga bisa dijalankan dengan mengambil nilai berapa jumlah P, S, dan M, dalam satu hari, lalu jika kondisi memenuhi maka akan mengeluarkan value true, dan apabila tidak maka akan mengeluarkan false, 1 jenis constraint pada hari biasa yaitu P $\leq$2, 1$\leq$ S $\geq$2, dan 2 $\leq$ M $\geq$3 maka hanya ada 1 nilai balikan yaitu value1.

5.3.4. Otomasi Pengecekan Constraint Divisi Gizi

```
1.  public boolean checkConstraintGizi(String [][] roster){
2.        boolean value1= false;
3.        boolean value2= false;
4.
5.        for (int j = NumberOfDays; j <NumberOfDays+LengthOfMonth[newMonths]; j++) {
6.            String [] Shift ={"PS","M"};
7.            int [] Jml_Shift = {0,0,0};
8.            for (int i = 0; i < NumberOfEmployee; i++) {
9.                if((j+FirstDay)%7 <= 1){
10.                    if(roster[i][j].equalsIgnoreCase("PS")){
11.                          Jml_Shift[0]++;
12.                     } else if(roster[i][j].equalsIgnoreCase("M")){
13.                          Jml_Shift[1]++;
14.                     }
15.
16.                   if (Jml_Shift[0] == 2 && Jml_Shift[1]== 1){
17.                       value1 = true;
18.                   }else{
19.                       value1 = false;
20.                   }
21.               } else{
22.                    if(roster[i][j].equalsIgnoreCase("PS")){
23.                          Jml_Shift[0]++;
24.                     } else if(roster[i][j].equalsIgnoreCase("M")){
25.                          Jml_Shift[1]++;
26.                     }
27.                   if (Jml_Shift[1]== 1){
28.                       value2 = true;
29.                   } else{
30.                       value2 = false;
31.                   }
32.               }
33.            }
34.       }
35.
36.       System.out.println("Cek Constraint");
37.       System.out.println("Nilai value1 : " + value1);
38.       System.out.println("Nilai value2 : " + value2);
39.
40.       if (value1=value2) {
41.           value1=true;
42.       } else {
43.           value1=false;
44.       }
45.
46.    return value1;
47.   }
```

Kode Program 5.28 Otomasi Pengecekan Constraint Divisi Gizi

Dalam pengecekan constraint pada divisi nicu dijadikan kedalam satu method, karena didalamnya hanya variable jumlah hari yang menjadi Batasan sehingga bisa dijalankan dengan mengambil nilai berapa jumlah P, S, dan M, dalam satu hari, lalu jika kondisi memenuhi maka akan mengeluarkan value true, dan apabila tidak maka akan mengeluarkan false, 2 jenis Batasan, yaitu di hari kerja dan di hari minggu maka dibedakan kebutuhannya, sehingga akan dikeluarkan dua nilai balikan yaitu value1 dan value2.

5.3.5. Otomasi Pengecekan Constraint Divisi OK

```
1.  public boolean checkConstraintOK(String [][] roster){
2.          boolean value1 = false;
3.          boolean value2 = false;
4.
5.          for (int j = NumberOfDays; j <NumberOfDays+LengthOfMonth[newMonths]; j++) {
6.              String [] Shift ={"P","S"};
7.              int [] Jml_Shift = {0,0,0};
8.              for (int i = 0; i < NumberOfEmployee; i++) {
9.                  if((j+FirstDay)%7 == 0){
10.                     if(roster[i][j].equalsIgnoreCase("P")){
11.                         Jml_Shift[0]++;
12.                     } else if(roster[i][j].equalsIgnoreCase("S")){
13.                         Jml_Shift[1]++;
14.                     }
15.
16.                     if (Jml_Shift[0] == 4 && Jml_Shift[1]== 0 ){
17.                         value1 = true;
18.                     } else{
19.                         value1 = false;
20.                     }
21.
22.                 } else{
23.                      if(roster[i][j].equalsIgnoreCase("P")){
24.                          Jml_Shift[0]++;
25.                      } else if(roster[i][j].equalsIgnoreCase("S")){
26.                          Jml_Shift[1]++;
27.                      }
28.
29.                     if (((Jml_Shift[0] >= 2) || (Jml_Shift[0] <= 4)) &&
30.                             ((Jml_Shift[1] >= 2) || (Jml_Shift[1] <= 3))){
31.                         value2 = true;
32.                     }else{
33.                         value2 = false;
34.                     }
35.                 }
36.              }
37.         }
38.         System.out.println("Cek Constraint");
39.         System.out.println("Nilai value1 : " + value1);
40.         System.out.println("Nilai value2 : " + value2);
41.
42.         if (value1=value2) {
43.             value1=true;
44.         } else {
45.             value1=false;
46.         }
47.     return value1;
48.    }
```

Kode Program 5.29 Otomasi Pengecekan Constraint Divisi OK

Dalam pengecekan constraint pada divisi nicu dijadikan kedalam satu method, karena didalamnya hanya variable jumlah hari yang menjadi Batasan sehingga bisa dijalankan dengan mengambil nilai berapa jumlah P, S, dan M, dalam satu hari, lalu jika kondisi memenuhi maka akan mengeluarkan value true, dan apabila tidak maka akan mengeluarkan false, 2 jenis Batasan, yaitu di hari kerja dan di hari minggu maka dibedakan kebutuhannya, sehingga akan dikeluarkan dua nilai balikan yaitu value1 dan value2.

5.3.6. Otomasi Pengecekan Constraint Divisi SIM & RM

```
1.  public boolean checkConstraintSIMRM(String [][] roster){
2.          boolean value1 = false;
3.          boolean value2 = false;
4.
5.          for (int j = NumberOfDays; j <NumberOfDays+LengthOfMonth[newMonths]; j++) {
6.              String [] Shift ={"P","S","M"};
7.              int [] Jml_Shift = {0,0,0};
8.              for (int i = 0; i < NumberOfEmployee; i++) {
9.                       if(roster[i][j].equalsIgnoreCase("P")){
10.                           Jml_Shift[0]++;
11.                       } else if(roster[i][j].equalsIgnoreCase("S")){
12.                           Jml_Shift[1]++;
13.                       }else if(roster[i][j].equalsIgnoreCase("M")){
14.                           Jml_Shift[2]++;
15.                       }
16.
17.                      if (((Jml_Shift[0] >= 1) || (Jml_Shift[0] <= 2)) &&
18.                              ((Jml_Shift[1] >= 1) || (Jml_Shift[1] <= 2))
19.                              && (Jml_Shift[2] == 1)){
20.                          value1 = true;
21.                      }else{
22.                          value1 = false;
23.                      }
24.              }
25.          }
26.          System.out.println("Cek Constraint");
27.          System.out.println("Nilai value1 : " + value1);
28.
29.      return value1;
30.     }
```

Kode Program 5.30 Otomasi Pengecekan Constraint Divisi SIM & RM

Dalam pengecekan constraint pada divisi nicu dijadikan kedalam satu method, karena didalamnya hanya variable jumlah hari yang menjadi Batasan sehingga bisa dijalankan dengan mengambil nilai berapa jumlah P, S, dan M, dalam satu hari, lalu jika kondisi memenuhi maka akan mengeluarkan value true, dan apabila tidak maka akan mengeluarkan false, 1 jenis constraint pada hari biasa yaitu $1 \leq P \geq 2$, $1 \leq S \geq 2$, dan $2 \leq M = 1$ maka hanya ada 1 nilai balikan yaitu value1.
Bh

5.4. Optimasi dengan implementasi Algoritma Self Adaptive Learning Hyper Heuristic

Dalam melakukan optimasi dengan menggunakan metode Hyper heuristic yang pertama dilakukan adalah melakukan *coding* untuk membuat *low level heuristic.* dalam *low level heuristic (LLH)* pada studi kasus ini akan menggunakan 2 buah

LLH yaitu *move()* dan *swap()*. Setelah itu baru dilakukan implementasi algoritma *Self Adaptive Learning*

5.4.1. Pembuatan Low Level Heuristic (LLH)

Berikut adalah kode program untuk membuat LLH pada studi kasus penjadwalan perawat di RSIA kendangsari, Surabaya.

1) *Low Level Heuristic 1, Move*

```
1.  public void move(){
2.        int  JmlLibur;
3.        int selisih;
4.        int JumlahLibur_seharusnya = (LengthOfMonth[newMonths]/7);
5.         if(((NumberOfDays + (FirstDay - 1)) % 7 ) >=4){
6.             JumlahLibur_seharusnya ++;
7.         }
8.        for (int i = 0 ; i <NumberOfEmployee; i++) {
9.            JmlLibur=0;
10.           selisih=0;
11.           List<Integer>  posisiMS = new ArrayList<>();
12.             for (int j = NumberOfDays; j <NumberOfDays+LengthOfMonth[newMonths]; j++) {

13.                 if(roster[i][j].equalsIgnoreCase("L")){
14.                         JmlLibur++;
15.                     }
16.                 if (roster [i][j].equalsIgnoreCase("MS")){
17.                     posisiMS.add(new Integer(j));
18.                  }
19.             }
20.             selisih = JumlahLibur_seharusnya-JmlLibur;
21.             if (selisih >0 && posisiMS.size()>0){
22.             for (int k = 0; k < selisih; k++) {
23.                 int IndexMSygDiganti = rand.nextInt(posisiMS.size());
24.                    roster[i][posisiMS.get(IndexMSygDiganti)]= "L";
25.                    }
26.             }
27.       }
28.    }
```

Kode Program 5.31 Pembuatan Low Leve Heuristic 1, Move

Dalam pembuatan method ini yang dilakukan adalah melakukan deklarasi variable yaitu selisih, JmlLibur, dan JumlahLibur dan JumlahLibur_seharusnya yang diambil dari jumlah hari minggu yang ada pada bulan tersebut,. Setelah itu dilakukan perulangan untuk mengambil jumlah libur yang telah didapatkan oleh setiap staf lalu dideklarasikan juga *ArrayList* untuk menyimpan jumlah

MS yang ada pada setiap staf. Setelah mendapatkan jumlah dari hari libur dan jumlah shift MS maka akan dicocokkan dengan jumlah libur yang harus didapatkan jika memiliki kekurangan jumlah libur maka akan dilakukan random Shift Middle mana yang akan diganti dengan shift libur.

2) *Low Level Heuristic 2, Swap*

```
1.  public void Swap(){
2.         List<Integer>  posisiLHari = new ArrayList<>();
3.          List<Integer>  posisiLEmployee = new ArrayList<>();
4.           for (int j = NumberOfDays; j <NumberOfDays+LengthOfMonth[newMonths]; j++) {
5.                for (int i = 0 ; i <NumberOfEmployee; i++) {
6.                   if(roster[i][j].equalsIgnoreCase("L")){
7.                       posisiLHari.add(new Integer(j));
8.                       posisiLEmployee.add(new Integer(i));
9.                    }
10.              }
11.          }
12.           int memilihHari = rand.nextInt(posisiLHari.size());
13.           String temp = roster[posisiLEmployee.get(memilihHari)][posisiLHari.get(memilih
    Hari)];
14.           int memilihEmployee;
15.
16.           do {
17.               memilihEmployee = rand.nextInt(NumberOfEmployee);
18.           }while(memilihEmployee == posisiLEmployee.get(memilihHari));
19.           roster[posisiLEmployee.get(memilihHari)][posisiLHari.get(memilihHari)]
20.                   = roster[posisiLEmployee.get(memilihEmployee)][posisiLHari.get(memilih
    Hari)];
21.           roster[posisiLEmployee.get(memilihEmployee)][posisiLHari.get(memilihHari)]= te
    mp;
22.      }
```

Kode Program 5.32 Pembuatan Low Level Heuristic 2, Swap

Untuk melakukan swap pertama-tama lakukan deklarasi *ArrayList* untuk menyimpan posisi dari hari dan karyawan yang akan dilakukan pertukaran, lalu tahap selanjutnya yaitu melakukan perulangan pada setiap hari yang berarti membaca tiap kolom yang ada pada jadwal (array roster) lalu jika menemukan “L” yaitu Libur maka indeks dari tanggal dan staff yang mendapatkan akan disimpan kedalam *ArrayList* setelah itu maka akan dilakukan penukaran pada hari tersebut untuk hari libur, sehingga akan menambahkan libur salah satu karyawan secara random namun tidak melanggar *Hard Constraint* per hari.

5.4.2. Implementasi Algoritma Self Adaptive Learning

```
1.  public  void SelfAdaptive(int iterationself, int IterationQueue) throws InterruptedExce
    ption {
2.          int [] BlockingQueue = new int [IterationQueue];
3.          bestJFIcost = curJFIcost = hitungJFI(roster);
4.          String move = "M";
5.          String swap = "S";
6.          String selectedLLH = null;
7.          String [][] Solpertama = roster.clone();
8.
9.          ArrayBlockingQueue<String> queue = new ArrayBlockingQueue<String>(IterationQueu
    e);
10.
11.         for (int j = 0; j < IterationQueue; j++) {
12.             if (Math.random() < 0.5) {
13.                 queue.put(move);
14.             } else {
15.                 queue.put(swap);
16.             }
17.         }
18.         for (int i = 0; i < iterationself; i++) {
19.             System.out.printf("%2d, %5f, %5f \n", i,curJFIcost, bestJFIcost );
20.             if (queue.isEmpty()) {
21.                 for (int j = 0; j < IterationQueue; j++) {
22.                     if (Math.random() < 0.5) {
23.                         queue.put(move);
24.                     } else {
25.                         queue.put(swap);
26.                     }
27.                 }
28.             } else {
29.                 if (queue.peek().equalsIgnoreCase("M")){
30.                     move();
31.                     queue.poll();
32.                     selectedLLH = "M";
33.                     //System.out.println("pilih move");
34.                 }else {
35.                     Swap();
36.                     queue.poll();
37.                     selectedLLH = "S";
38.                     //1System.out.println("pilih swap");
39.                 }
40.             }
41.             curJFIcost= hitungJFI(roster);
42.             if (curJFIcost > bestJFIcost) {
43.                 bestJFIcost = curJFIcost;
44.                 queue.put(selectedLLH);
45.                 roster = Solpertama.clone();
46.             }
47.         }
48.         System.out.println("Best JFI : "+ bestJFIcost );
49.     }
```

Kode Program 5.33 Implementasi Algoritma Self Adaptive Learning

Tahap tahap dalam melakukan implementasi algoritma Self Adaptive Learning adalah sebagai berikut :

1) Deklarasikan *Array Blocking Queue* untuk menyimpan antrian untuk eksekusi swap dan move. *Array Blocking Queue* adalah suatu array yang memiliki Panjang antrian yang dapat ditentukan , dalam array blocking queue memiliki prinsip FIFO (*First in, First out)* sehingga yang memiliki urutan paling depan adalah yang masuk paling awal dan dieksekusi paling awal [11]. *Array Blocking Queue* pada algoritma ini digunakan untuk menampung antrian dari string yang melambangkan satu method jika yang keluar adalah String "M" maka yang akan dijalankan adalah method Move, jika yang keluar adalah String "S", maka yang akan dijalankan adalah Swap.
2) Setelah deklarasi *Array Blocking Queue* selanjutnya adalah menetapkan jumlah iterasi dan jumlah kapasitas dari *Array Blocking Queue*
3) Lalu lakukan pengisian queue dengan melakukan perulangan dan mengisi secara random sebanyak kapasitas dari queue
4) Lakukan perulangan sebanyak yang diinginkan (n kali) lalu dalam perulangan tersebut buat kondisi :
    a. Jika yang ada di dalam queue yang pertama adalah String "M" maka jalankan method move(), jika String "S" maka jalankan method Swap().
    b. Jika hasil dari curJFI atau hasil dari eksekusi method lebih baik daripada bestJFI yang sebelumnya telah diinisiasikan maka LLH tersebut akan masuk kembali kedalam queue (put()), jika tidak maka akan dihapus (poll()).
5) Jika telah mendapatkan hasil terbaik (bestJFI) dari iterasi maka akan dilakukan *clone* untuk diterapkan kedalam jadwal.

BAB VI
HASIL DAN PEMBAHASAN

Bab ini akan menjelaskan mengenai hasil dari otomasi dan optimasi dari penjadwalan perawat di RSIA Kendangsari Surabaya

6.1. Hasil Otomasi Penjadwalan per Divisi

Pada sub bab kali ini akan ditampilkan hasil dari otomasi penjadwalan yang ada pada setiap divisi berdasarkan kode program serta ditampilkan juga untuk nilai *Jain Fairness Index* dari masing-masing divisi. Berikut adalah tabel keterangan dari jadwal.

Tabel 6.1 Keterangan Jadwal

Shift	Kode	Jam
Pagi	P	07.00 – 14.00 WIB
Pagi 1	P1	04.30 – 11.30 WIB
Pagi 2	P2	05.00 – 12.00 WIB
Pagi-Sore	PS	06.00 – 20.00 WIB
Siang	S	14.00 – 21.00 WIB
Malam	M	21.00 – 07.00 WIB
Lepas Malam	LM	-
Middle Shift	MS	10.00 – 17.00 WIB
Middle Shift 2	MD	12.00 – 19.00 WIB
Libur	L	-

1) Hasil Otomasi Penjadwalan Divisi Farmasi

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
J	Sb	M	Sn	S1	R	K	J	Sb	M	Sn	S1	R	K	J	Sb
P	P	L	P	P	P	P	P	P	L	P	P	P	P	P	P
S	S	L	S	MS	MS	MS	S	S	L	MS	S	S	S	MS	MS
M	M	LM	L	P	P	S	S	MS	S	M	M	LM	L	P	P
MS	MS	M	M	LM	L	P	P	S	S	MS	MS	M	M	LM	L
S	S	S	MS	M	M	LM	L	P	P	S	S	MS	MS	M	M
P	P	S	S	MS	MS	M	M	LM	L	P	P	S	S	MS	MS
LM	L	P	P	S	S	MS	MS	M	M	LM	L	P	P	S	S
P	P	L	P	P	P	P	P	P	L	P	P	P	P	P	P

17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
M	Sn	S1	R	K	J	Sb	M	Sn	S1	R	K	J	Sb	M
L	P	P	P	P	P	P	L	P	P	P	P	P	P	L
L	MS	MS	S	MS	MS	S	L	S	S	S	S	MS	MS	L
S	S	MS	MS	M	M	LM	L	P	P	S	S	MS	MS	M
P	P	S	S	MS	MS	M	M	LM	L	P	P	S	S	S
LM	L	P	P	S	S	MS	S	M	M	LM	L	P	P	S
M	M	LM	L	P	P	S	S	MS	MS	M	M	LM	L	P
S	MS	M	M	LM	L	P	P	S	S	MS	MS	M	M	LM
L	P	P	P	P	P	P	L	P	P	P	P	P	P	L

```
    Hari Minggu:  500     Hari Sabtu:    0     Hari Kerja:    0 Sum :  500
    Hari Minggu:  500     Hari Sabtu:    0     Hari Kerja:    0 Sum :  500
    Hari Minggu:  100     Hari Sabtu:    0     Hari Kerja:   50 Sum :  150
    Hari Minggu:    0     Hari Sabtu:   50     Hari Kerja:   50 Sum :  100
    Hari Minggu:    0     Hari Sabtu:    0     Hari Kerja:   75 Sum :  75
    Hari Minggu:  100     Hari Sabtu:   50     Hari Kerja:   25 Sum :  175
    Hari Minggu:    0     Hari Sabtu:   50     Hari Kerja:   50 Sum :  100
    Hari Minggu:  500     Hari Sabtu:    0     Hari Kerja:    0 Sum :  500
Jfi: 0.665158371040724
```

Gambar 6.1 Hasil Otomasi Penjadwalan Farmasi

2) Hasil Otomasi Penjadwalan Divisi IGD

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
J	Sb	M	Sn	S1	R	K	J	Sb	M	Sn	S1	R	K	J	Sb
M	M	LM	L	P	P	S	S	S	S	S	L	MD	MS	S	M
MS	S	M	M	LM	L	P	P	S	S	S	S	S	L	MD	MS
L	MD	MS	S	M	M	LM	L	P	P	S	S	S	S	S	L
S	S	L	MD	MS	S	M	M	LM	L	P	P	S	S	S	S
S	S	S	S	L	MD	MS	S	M	M	LM	L	P	P	S	S
P	S	S	S	S	S	L	MD	MS	S	M	M	LM	L	P	P
L	P	P	S	S	S	S	S	L	MD	MS	S	M	M	LM	L
M	LM	L	P	P	S	S	S	S	S	L	MD	MS	S	M	M

```
17   18   19   20   21   22   23   24   25   26   27   28   29   30   31
 M   Sn   Sl    R    K    J   Sb    M   Sn   Sl    R    K    J   Sb    M
 M   LM    L    P    P    S    S    S    S    S    L   MD   MS    S    M
 S    M    M   LM    L    P    P    S    S    S    S    S    L   MD   MS
MD   MS    S    M    M   LM    L    P    P    S    S    S    S    S    L
 S    L   MD   MS    S    M    M   LM    L    P    P    S    S    S    S
 S    S    S    L   MD   MS    S    M    M   LM    L    P    P    S    S
 S    S    S    S    S    L   MD   MS    S    M    M   LM    L    P    P
 P    P    S    S    S    S    S    L   MD   MS    S    M    M   LM    L
LM    L    P    P    S    S    S    S    S    L   MD   MS    S    M    M
```

```
    Hari Minggu:   0     Hari Sabtu:   0     Hari Kerja:  100 Sum :  100
    Hari Minggu:   0     Hari Sabtu:   0     Hari Kerja:  100 Sum :  100
    Hari Minggu: 100     Hari Sabtu: 100     Hari Kerja:   50 Sum :  250
    Hari Minggu: 200     Hari Sabtu:   0     Hari Kerja:   50 Sum :  250
    Hari Minggu:   0     Hari Sabtu:   0     Hari Kerja:  100 Sum :  100
    Hari Minggu:   0     Hari Sabtu:   0     Hari Kerja:  100 Sum :  100
    Hari Minggu: 200     Hari Sabtu: 100     Hari Kerja:   25 Sum :  325
    Hari Minggu: 100     Hari Sabtu:   0     Hari Kerja:   75 Sum :  175
Jfi: 0.8132780082987552
```

Gambar 6.2 Hasil Otomasi Penjadwalan Divisi IGD

3) Hasil Otomasi Penjadwalan Divisi Bayi & NICU

```
 1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16
 J  Sb   M  Sn  Sl   R   K   J  Sb   M  Sn  Sl   R   K   J  Sb
 P   P   L   P   P   P   P   P   P   L   P   P   P   P   P   P
 M   M  LM   L   P   P   S   S  MS  MS   M   M  LM   L   P   P
MS  MS   M   M  LM   L   P   P   S   S  MS  MS   M   M  LM   L
 S   S  MS  MS   M   M  LM   L   P   P   S   S  MS  MS   M   M
 P   P   S   S  MS  MS   M   M  LM   L   P   P   S   S  MS  MS
LM   L   P   P   S   S  MS  MS   M   M  LM   L   P   P   S   S
 M   M  LM   L   P   P   S   S  MS  MS   M   M  LM   L   P   P
MS  MS   M   M  LM   L   P   P   S   S  MS  MS   M   M  LM   L
 S   S  MS  MS   M   M  LM   L   P   P   S   S  MS  MS   M   M
 P   P   S   S  MS  MS   M   M  LM   L   P   P   S   S  MS  MS
LM   L   P   P   S   S  MS  MS   M   M  LM   L   P   P   S   S
 M   M  LM   L   P   P   S   S  MS  MS   M   M  LM   L   P   P
MS  MS   M   M  LM   L   P   P   S   S  MS  MS   M   M  LM   L

18  19  20  21  22  23  24  25  26  27  28  29  30  31
Sn  Sl   R   K   J  Sb   M  Sn  Sl   R   K   J  Sb   M
 P   P   P   P   P   P   L   P   P   P   P   P   P   L
 S  MS  MS   M   M  LM   L   P   P   S   S  MS  MS   M
 P   S   S  MS  MS   M   M  LM   L   P   P   S   S  MS
 L   P   P   S   S  MS  MS   M   M  LM   L   P   P   S
 M  LM   L   P   P   S   S  MS  MS   M   M  LM   L   P
MS   M   M  LM   L   P   P   S   S  MS  MS   M   M  LM
 S  MS  MS   M   M  LM   L   P   P   S   S  MS  MS   M
 P   S   S  MS  MS   M   M  LM   L   P   P   S   S  MS
 L   P   P   S   S  MS  MS   M   M  LM   L   P   P   S
 M  LM   L   P   P   S   S  MS  MS   M   M  LM   L   P
MS   M   M  LM   L   P   P   S   S  MS  MS   M   M  LM
 S  MS  MS   M   M  LM   L   P   P   S   S  MS  MS   M
 P   S   S  MS  MS   M   M  LM   L   P   P   S   S  MS
```

```
    Hari Minggu:  500     Hari Sabtu:    0     Hari Kerja:    0 Sum :  500
    Hari Minggu:  100     Hari Sabtu:    0     Hari Kerja:   50 Sum :  150
    Hari Minggu:    0     Hari Sabtu:   50     Hari Kerja:   50 Sum :  100
    Hari Minggu:    0     Hari Sabtu:    0     Hari Kerja:   75 Sum :  75
    Hari Minggu:  100     Hari Sabtu:   50     Hari Kerja:   25 Sum :  175
    Hari Minggu:    0     Hari Sabtu:   50     Hari Kerja:   50 Sum :  100
    Hari Minggu:  100     Hari Sabtu:    0     Hari Kerja:   50 Sum :  150
    Hari Minggu:    0     Hari Sabtu:   50     Hari Kerja:   50 Sum :  100
    Hari Minggu:    0     Hari Sabtu:    0     Hari Kerja:   75 Sum :  75
    Hari Minggu:  100     Hari Sabtu:   50     Hari Kerja:   25 Sum :  175
    Hari Minggu:    0     Hari Sabtu:   50     Hari Kerja:   50 Sum :  100
    Hari Minggu:  100     Hari Sabtu:    0     Hari Kerja:   50 Sum :  150
    Hari Minggu:    0     Hari Sabtu:   50     Hari Kerja:   50 Sum :  100
Jfi: 0.6647727272727273
```

Gambar 6.3 Hasil Otomasi Penjadwalan Divisi Bayi & NICU

4) Hasil Otomasi Penjadwalan Divisi Gizi

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
R	K	J	Sb	M	Sn	Sl	R	K	J	Sb	M	Sn	Sl	R	K
P	P	P	P	L	P	P	P	P	P	P	L	P	P	P	P
P	L	S	P1	P	P1	S	S	P	CT	L	L	P1	P1	S	S
L	MD	MD	MD	P1	MD	CT	MD	S	P1	P1	L	L	S	MD	S
MD	P2	P1	L	S	S	P	CT	CT	L	MD	S	S	MD	P1	MD
S	S	CT	S	MD	L	MD	P	S	S	S	P1	MD	L	CT	P
P1	P1	S	CT	L	S	P1	P1	P1	MD	S	MD	L	S	S	P1
P2	S	S	S	S	L	S	S	S	S	S	L	S	P	P	P2
P2	L	P2	CT	L	P2	P2	P2	L	P	P	S	P2	P2	P2	L
S	S	P	P	P2	L	S	S	S	P2	P2	P2	L	L	CT	S
P	P	MD	MD	P	L	S	P	P	P	MD	MD	L	S	P	MD
L	CT	S	S	MD	M	M	M	LM	L	P	P	M	M	LM	L
S	S	M	M	M	LM	L	MD	S	S	S	L	CT	CT	M	M
MD	MD	P	P	L	P	P	S	M	M	LM	L	S	S	P	P
M	M	LM	CT	S	MD	S	P	P	P	M	M	LM	CT	S	S
P	P	MD	MD	P	S	L	M	M	LM	L	S	P	P	MD	L
P	P	P	S	L	PS	P	S	S	S	P	L	PS	S	P	P
S	S	S	P	PS	L	S	P	P	P	S	PS	L	P	S	S
S	S	S	P	PS	L	P	S	S	P	P	L	PS	S	S	P
CT	CT	CT	S	L	PS	S	CT	CT	S	S	PS	L	CT	CT	S

17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
M	Sn	Sl	R	K	J	Sb	M	Sn	Sl	R	K	J	Sb	M
L	P	P	P	P	P	P	L	P	P	P	P	P	P	L
S	S	MD	MD	P1	P	S	P1	P	L	S	S	MD	MD	P1
P	L	S	S	MD	MD	P1	P	S	P1	P	L	S	S	MD
S	P1	P	L	S	S	MD	MD	P1	P	S	P1	P	L	S
P1	P	S	P1	P	L	S	S	MD	MD	P1	P	S	P1	P
S	S	S	P1	P1	L	P1	P1	S	S	S	P1	P1	L	P1
S	P2	P	MS	P2	P	S	S	L	P	S	P2	P	MS	P2
L	P	S	P2	P	MS	P2	P	S	S	L	P	S	P2	P
S	S	L	P	S	P2	P	MS	P2	P	S	S	L	P	S
P	P	MD	MD	L	S	S	P	P	MD	MD	L	S	S	P
S	S	P	P	M	M	LM	L	P	P	S	S	P	P	M
P	P	S	S	P	P	M	M	LM	L	P	P	S	S	P
LM	L	P	P	S	S	P	P	M	M	LM	L	P	P	S
M	M	LM	L	P	P	S	S	P	P	M	M	LM	L	P
MD	MD	M	M	LM	L	P	P	S	S	P	MD	M	M	LM
L	PS	S	S	P	P	P	PS	L	P	P	S	S	S	L
PS	L	P	P	S	S	S	L	PS	S	S	P	P	P	PS
PS	L	P	P	S	S	S	L	PS	S	S	P	P	P	PS
L	PS	S	S	P	P	P	PS	L	P	P	S	S	S	L

```
Hari Minggu:  500     Hari Sabtu:    0     Hari Kerja:    0 Sum :  500
Hari Minggu:    0     Hari Sabtu:   50     Hari Kerja:   50 Sum :  100
Hari Minggu:    0     Hari Sabtu:    0     Hari Kerja:   75 Sum :  75
Hari Minggu:  100     Hari Sabtu:   50     Hari Kerja:   25 Sum :  175
Hari Minggu:    0     Hari Sabtu:   50     Hari Kerja:   50 Sum :  100
Hari Minggu:    0     Hari Sabtu:   50     Hari Kerja:   75 Sum :  125
Hari Minggu:    0     Hari Sabtu:    0     Hari Kerja:   75 Sum :  75
Hari Minggu:  100     Hari Sabtu:    0     Hari Kerja:   50 Sum :  150
Hari Minggu:    0     Hari Sabtu:   50     Hari Kerja:   50 Sum :  100
Hari Minggu:    0     Hari Sabtu:    0     Hari Kerja:  100 Sum :  100
Hari Minggu:  100     Hari Sabtu:    0     Hari Kerja:   50 Sum :  150
Hari Minggu:    0     Hari Sabtu:   50     Hari Kerja:   50 Sum :  100
Hari Minggu:    0     Hari Sabtu:    0     Hari Kerja:   75 Sum :  75
Hari Minggu:  100     Hari Sabtu:   50     Hari Kerja:   25 Sum :  175
Hari Minggu:    0     Hari Sabtu:   50     Hari Kerja:   50 Sum :  100
Hari Minggu:  300     Hari Sabtu:    0     Hari Kerja:   50 Sum :  350
Hari Minggu:  200     Hari Sabtu:    0     Hari Kerja:   50 Sum :  250
Hari Minggu:  200     Hari Sabtu:    0     Hari Kerja:   50 Sum :  250
Hari Minggu:  300     Hari Sabtu:    0     Hari Kerja:   50 Sum :  350
Jfi: 0.7000401767778224
```

Gambar 6.4 Hasil Otomasi Penjadwalan Divisi Bayi & NICU

5) Hasil Otomasi Penjadwalan Divisi OK

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
J	Sb	M	Sn	Sl	R	K	J	Sb	M	Sn	Sl	R	K	J	Sb
P	P	L	P	S	L	P	P	S	P	P	P	S	P	S	L
S	P	L	S	P	S	L	P	P	P	S	P	P	S	P	S
S	S	P	P	S	P	S	L	P	P	S	S	P	P	S	P
P	S	P	P	P	S	P	S	L	P	P	S	P	P	P	S
P	P	P	P	P	P	S	P	S	L	P	P	S	P	P	P
L	P	P	S	P	P	P	S	P	L	L	P	P	S	P	P

17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
M	Sn	Sl	R	K	J	Sb	M	Sn	Sl	R	K	J	Sb	M
P	P	S	S	P	P	S	L	S	L	P	P	S	S	P
P	P	P	S	S	P	P	L	P	S	L	P	P	S	P
L	L	P	P	S	S	P	P	S	P	S	L	P	P	P
L	S	L	P	P	S	P	P	P	S	P	S	L	P	P
P	P	S	L	P	P	S	P	P	P	S	P	S	L	L
P	S	P	S	L	P	P	P	P	P	P	S	P	S	L

```
    Hari Minggu:    8     Hari Sabtu:    2     Hari Kerja:    2 Sum :  12
    Hari Minggu:    8     Hari Sabtu:    0     Hari Kerja:    2 Sum :  10
    Hari Minggu:    4     Hari Sabtu:    0     Hari Kerja:    3 Sum :  7
    Hari Minggu:    4     Hari Sabtu:    2     Hari Kerja:    2 Sum :  8
    Hari Minggu:    8     Hari Sabtu:    2     Hari Kerja:    1 Sum :  11
    Hari Minggu:    8     Hari Sabtu:    0     Hari Kerja:    3 Sum :  11
Jfi: 0.9685587089593768
```

Gambar 6.5 Hasil Otomasi Penjadwalan Divisi OK

6) Hasil Otomasi Penjadwalan Divisi SIM & RM

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
J	Sb	M	Sn	Sl	R	K	J	Sb	M	Sn	Sl	R	K	J	Sb
P	L	S	S	MS	MS	M	M	LM	L	P	P	P	L	S	S
P	P	P	L	S	S	MS	MS	M	M	LM	L	P	P	P	L
LM	L	P	P	P	L	S	S	MS	MS	M	M	LM	L	P	P
M	M	LM	L	P	P	P	L	S	S	MS	MS	M	M	LM	L
MS	MS	M	M	LM	L	P	P	P	L	S	S	MS	MS	M	M
S	S	P	MS	M	M	LM	L	P	P	P	L	S	S	P	MS

18	19	20	21	22	23	24	25	26	27	28	29	30	
Sn	Sl	R	K	J	Sb	M	Sn	Sl	R	K	J	Sb	M
MS	M	M	LM	L	P	P	P	L	S	S	MS	MS	M
S	MS	MS	M	M	LM	L	P	P	P	L	S	S	MS
L	S	S	MS	MS	M	M	LM	L	P	P	P	L	S
P	P	L	S	S	MS	MS	M	M	LM	L	P	P	P
L	P	P	P	L	S	S	MS	MS	M	M	LM	L	P
M	LM	L	P	P	P	L	S	S	P	MS	M	M	LM

```
    Hari Minggu:  100     Hari Sabtu:   50     Hari Kerja:   75 Sum :  225
    Hari Minggu:  100     Hari Sabtu:   50     Hari Kerja:   75 Sum :  225
    Hari Minggu:    0     Hari Sabtu:  100     Hari Kerja:  100 Sum :  200
    Hari Minggu:    0     Hari Sabtu:   50     Hari Kerja:  100 Sum :  150
    Hari Minggu:  100     Hari Sabtu:   50     Hari Kerja:   75 Sum :  225
    Hari Minggu:  100     Hari Sabtu:    0     Hari Kerja:   75 Sum :  175
Jfi: 0.9795918367346939
```

Gambar 6.6 Hasil Otomasi Penjadwalan Divisi SIM & RM

6.2. Hasil Optimasi Penjadwalan per Divisi

Pada sub bab kali ini akan ditampilkan hasil dari otomasi penjadwalan yang ada pada setiap divisi berdasarkan kode program serta ditampilkan juga untuk nilai Jumlah karyawan pada masing-masing shift dari masing-masing divisi.

1) Hasil Optimasi Penjadwalan Divisi Farmasi

Tabel 6.3 Hasil Optimasi Divisi Farmasi

ID Staff	Skill/Pekerjaan	ID Skill	Tanggal																																
			J	Sb	M	Sn	Sl	Rb	K	J	Sb	M	Sn	Sl	Rb	K	J	Sb	M	Sn	Sl	Rb	K	J	Sb	M	Sn	Sl	Rb	K	J	Sb	M		
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31		
401			P	P	L	P	P	P	P	P	P	L	P	M	P	P	P	P	L	P	P	P	P	S	P	P	P	P	P	P	P	P	L		
402			S	S	L	S	S	S	S	S	S	S	S	L	S	S	L	S	L	S	S	S	S	L	S	L	S	S	S	S	S	S	M		
403			M	M	LM	L	P	P	S	S	MS	S	M	S	LM	L	P	P	S	S	L	MS	M	M	LM	L	L	P	S	L	L	MS	L		
404			L	MS	M	M	LM	L	P	P	S	L	MS	L	M	M	LM	MS	P	P	S	S	MS	P	M	M	LM	L	P	P	S	S	S		
405			S	S	S	L	M	M	LM	M	P	P	S	S	L	L	M	M	LM	L	P	P	S	S	P	S	M	M	LM	S	P	P	S		
406			P	P	S	S	L	L	M	L	LM	L	P	P	S	S	S	L	M	M	LM	L	P	P	S	S	S	L	M	M	LM	M	P		
407			LM	L	P	P	S	S	MS	MS	M	M	LM	P	P	P	S	S	S	L	M	M	LM	MS	L	L	P	S	MS	MS	M	L	LM		
408			P	P	L	P	P	P	P	P	P	L	P	P	P	P	P	P	L	P	P	P	P	P	P	L	P	P	P	P	P	P	L		

Tabel 6.2 Jumlah Staf per Shift

P	S	M	MS	LM	L	CT
25	1	1	0	0	4	0
0	24	1	0	0	6	0
5	7	5	3	3	8	0
7	6	6	4	3	5	0
7	10	7	0	3	4	0
6	8	6	0	3	7	0
6	6	5	5	4	5	0
26	0	0	0	0	5	0

2) Hasil Optimasi Penjadwalan Divisi IGD

Tabel 6.4 Hasil Optimasi Divisi IGD

ID Staff	Skill/ Pekerjaan	ID Skill	Tanggal																													
			J	Sb	M	Sn	Sl	Rb	K	J	Sb	M	Sn	Sl	Rb	K	J	Sb	M	Sn	Sl	Rb	K	J	Sb	M	Sn	Sl	Rb	K	J	Sb	M
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
401			M	M	LM	L	P	L	S	S	S	L	L	S	MD	L	S	M	M	LM	P	P	P	S	S	S	S	S	MD	MD	MS	S	M
402			L	S	M	M	LM	P	P	P	S	S	S	L	S	P	MD	MS	S	M	M	LM	L	P	P	S	S	S	S	S	L	MD	L
403			L	MD	L	S	M	M	LM	M	P	P	S	S	S	S	S	L	MD	MS	S	M	M	LM	L	P	P	L	S	S	S	S	MS
404			S	S	L	MD	L	S	M	L	LM	S	P	P	S	S	S	S	S	L	MD	L	S	M	M	LM	L	P	P	S	S	S	S
405			S	S	S	S	L	MD	L	S	M	M	LM	L	P	L	S	S	S	S	S	L	MD	MS	S	M	M	LM	L	P	P	S	S
406			P	S	S	S	S	S	L	MD	L	S	M	M	LM	L	P	P	S	S	S	S	S	L	MD	MS	S	M	M	LM	L	P	P
407			L	P	P	S	S	S	S	S	L	MD	MS	S	M	M	LM	L	P	P	S	S	S	S	S	L	MD	MS	S	M	M	LM	L
408			M	LM	L	P	P	S	S	S	S	S	S	MD	MS	S	M	M	LM	L	L	P	S	S	S	S	S	S	L	L	S	M	M

Tabel 6.5 Jumlah Staf per shift

P	S	M	MS	LM	L	CT
4	11	5	1	2	5	0
6	11	4	1	2	5	0
4	11	5	2	2	5	0
4	14	3	0	2	6	0
3	13	4	1	2	6	0
4	12	4	1	2	5	0
4	12	4	2	2	5	0
3	14	5	1	2	5	0

3) Hasil Optimasi Penjadwalan Divisi Bayi & NICU

Tabel 6.6 Hasil Optimasi Penjadwalan Divisi Bayi & NICU

ID Staff	Skill/Pekerjaan	ID Skill	Tanggal																																	
			J	Sb	M	Sn	Sl	Rb	K	J	Sb	M	Sn	Sl	Rb	K	J	Sb	M	Sn	Sl	Rb	K	J	Sb	M	Sn	Sl	Rb	K	J	Sb	M
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
201	Kepala Unit	1	P	P	L	P	P	P	P	P	P	L	P	P	P	P	P	P	L	P	P	P	P	P	P	L	P	P	P	P	P	P	L
202	Anggota	4	M	L	LM	P	P	P	L	S	L	MS	M	M	LM	L	P	P	S	S	MS	MS	M	M	LM	L	P	P	S	S	L	MS	M
203	Anggota	4	M	M	M	M	LM	M	P	P	S	S	L	MS	M	M	LM	L	P	P	S	S	L	L	M	M	LM	P	P	P	S	S	L
204	Anggota	4	S	S	L	MS	M	L	LM	L	P	P	S	S	MS	L	M	M	LM	S	P	P	S	L	MS	L	M	M	LM	L	P	P	S
205	Anggota	4	P	P	S	S	MS	MS	M	M	LM	L	P	P	S	S	MS	L	M	M	LM	L	P	P	S	S	L	MS	M	M	LM	L	P
207	Anggota	4	LM	L	P	L	S	S	S	MS	M	M	LM	L	P	P	S	L	LM	L	M	M	LM	L	P	P	S	S	P	MS	M	M	LM
208	Anggota	4	M	L	LM	L	P	P	S	S	MS	MS	M	M	LM	L	P	P	S	L	MS	L	M	M	LM	MS	P	L	S	S	L	MS	M
209	Anggota	4	MS	MS	M	M	LM	P	P	P	S	S	MS	L	L	M	LM	S	P	P	S	S	L	MS	M	M	LM	L	P	P	S	S	L
210	Anggota	4	S	S	L	MS	M	M	LM	L	P	P	S	S	M	MS	M	M	L	S	P	P	S	S	MS	L	M	M	LM	L	P	P	S
211	Anggota	4	P	P	S	S	MS	MS	M	M	LM	L	P	P	S	S	MS	L	M	M	LM	L	P	P	S	S	MS	L	M	M	LM	L	P
212	Anggota	4	LM	M	P	P	S	S	L	MS	M	M	LM	L	P	P	S	S	L	L	M	M	LM	S	P	P	S	S	MS	L	M	M	LM
213	Anggota	4	L	M	LM	L	P	L	S	S	MS	MS	M	L	LM	L	P	P	S	L	MS	MS	M	M	LM	L	P	P	S	S	L	S	M
214	Anggota	4	L	MS	M	M	LM	L	P	P	S	S	MS	M	M	M	LM	L	P	P	S	S	MS	MS	M	M	LM	L	L	P	S	L	MS

Tabel 6.7 Jumlah Staff per Shift

P	S	M	MS	LM	L	CT
26	0	0	0	0	5	0
7	5	6	4	3	6	0
7	6	9	1	3	5	0
6	7	5	3	3	7	0
6	6	6	4	3	5	0
6	6	6	2	5	6	0
5	5	6	5	3	7	0
7	7	5	4	3	5	0
6	8	7	3	2	5	0
6	6	6	4	3	5	0
6	7	7	2	4	5	0
5	6	5	4	3	8	0
5	5	7	5	3	6	0

4) Hasil Optimasi Penjadwalan Divisi Gizi

Tabel 6.8 Hasil Optimasi Penjadwalan Divisi Gizi

ID Staff	Skill/Pekerjaan	ID Skill	Tanggal																																					
			J	Sb	M	Sn	Sl	Rb	K	J	Sb	M	Sn	Sl	Rb	K	J	Sb	M	Sn	Sl	Rb	K	J	Sb	M	Sn	Sl	Rb	K	J	Sb	M							
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31							
501	Ahli gizi	1	P	P	L	P	P	P	P	P	P	L	P	P	P	P	P	P	L	L	P	P	P	L	P	M	P	P	P	P	P	P	L							
502	chef	5	P1	P	S	P1	P	S	S	S	MD	MD	P1	P	S	P1	P	L	S	S	MD	MD	P1	P	S	P1	P	L	S	S	MD	MD	P1							
503	chef	5	MD	MD	L	L	S	P1	P	L	S	S	MD	MD	P1	P	S	P1	P	L	S	S	MD	MD	P1	P	S	L	P	L	P	S	MD							
504	chef	5	S	S	MD	MD	P1	P	S	P1	P	L	S	S	MD	MD	P1	P	S	P1	P	M	S	S	MD	MD	P1	P	S	P1	P	L	S							
505	chef	5	P	L	S	S	MD	MD	P1	P	S	L	P	L	S	S	MD	MD	P1	P	S	L	P	L	S	S	MD	MD	P1	P	S	P1	P							
506	Helper	6	S	S	S	P1	P1	L	P1	P1	S	S	S	P1	P1	L	P1	P1	S	S	S	P1	P1	L	P1	L	S	S	S	P1	P1	S	P1							
507	Helper	6	P2	P	S	S	P	P	S	P2	P	L	P2	P	S	S	L	P	S	P2	P	L	P2	P	S	S	L	P	S	P2	P	L	P2							
508	Helper	6	P	L	P2	P	S	S	S	P	S	P2	P	MD	P2	P	S	S	L	P	S	P2	P	P	P2	P	S	S	L	P	S	P2	P							
509	Helper	6	S	P2	P	P	P2	P	LM	S	L	P	S	P2	P	M	P2	P	S	L	L	P	S	P2	P	L	P2	P	S	S	S	P	S							
510	Penyaji	7	S	S	P	P	MD	MD	L	S	S	P	L	L	MD	P	S	S	P	P	MD	MD	L	S	S	P	P	MD	MD	L	S	S	P							
511	Penyaji	7	M	M	LM	L	P	P	S	S	P	P	M	M	LM	L	P	P	S	P	P	P	M	M	LM	P1	P	P	S	S	P	P	M							
512	Penyaji	7	P	P	M	M	LM	L	P	P	S	S	P	P	M	L	LM	L	P	P	S	S	P	P	M	L	LM	P1	P	P	S	L	P							
513	Penyaji	7	S	S	P	P	M	M	L	L	P	P	S	S	P	P	M	M	LM	S	P	P	S	S	P	P	M	M	LM	L	P	P	S							
514	Penyaji	7	P	P	S	S	P	P	M	M	LM	L	P	P	S	S	P	P	M	M	LM	P1	P	P	S	S	P	P	M	M	LM	L	P							
515	Penyaji	7	LM	L	P1	P	S	S	MD	MD	M	M	LM	L	P	L	S	S	MD	MD	M	L	LM	L	P	P	S	S	P	MD	M	M	L							
516	Cafe	8	S	S	P	PS	S	S	P	P	P	PS	L	P	P	S	S	S	L	PS	S	S	P	P	P	PS	L	P	P	S	S	S	L							
517	Cafe	8	P	P	PS	L	P	P	S	S	S	P1	PS	S	S	P	P	P	PS	S	P	P	S	S	S	L	PS	S	S	P	L	P	PS							
518	Driver	9	P	P	PS	L	L	P	S	S	S	L	P	S	S	P	P	P	PS	L	P	P	S	S	S	L	PS	S	S	P	P	P	PS							
519	Driver	9	S	S	L	PS	S	L	P	P	P	PS	PS	P	P	S	S	S	L	PS	S	S	P	P	P	PS	L	P	P	S	S	S	LM							

Tabel 6.9 Jumlah Staff Per Shift

P	S	M	MS	LM	L	CT
24	0	1	0	0	6	0
6	10	0	0	0	2	0
6	8	0	0	0	6	0
6	10	1	0	0	2	0
6	9	0	0	0	5	0
0	13	0	0	0	4	0
10	9	0	0	0	5	0
10	10	0	0	0	3	0
10	9	1	0	1	4	0
8	10	0	0	0	5	0
13	5	7	0	3	2	0
12	5	4	0	3	5	0
12	8	6	0	2	3	0
12	6	6	0	3	2	0
5	6	5	0	3	6	0
11	12	0	0	0	4	0
11	11	0	0	0	3	0
12	10	0	0	0	5	0
10	11	0	0	1	4	0

5) `Hasil Optimasi Penjadwalan Divisi OK

Tabel 6.10 Hasil Optimasi Penjadwalan Divisi OK

ID Staff	Skill/Pekerjaan	ID Skill	Tanggal																																
			J	Sb	M	Sn	Sl	Rb	K	J	Sb	M	Sn	Sl	Rb	K	J	Sb	M	Sn	Sl	Rb	K	J	Sb	M	Sn	Sl	Rb	K	J	Sb	M		
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31		
601	Staf	4	P	P	L	P	S	L	P	P	S	P	P	P	S	P	S	L	P	P	S	S	P	P	S	L	S	S	P	P	S	S	P		
602	Staf	4	S	P	L	S	P	S	L	P	P	P	S	P	P	S	P	S	P	P	P	S	S	P	P	L	P	L	L	P	P	S	P		
603	Staf	4	S	S	P	P	S	P	S	L	P	P	S	S	P	P	S	P	L	L	P	P	S	S	P	P	S	P	S	L	P	P	P		
606	Staf	4	P	S	P	P	P	S	P	S	L	P	P	S	P	P	P	S	L	S	L	P	P	S	P	P	P	S	P	S	L	P	P		
604	Staf	4	P	P	P	P	P	P	S	P	S	L	P	P	S	P	P	P	P	P	S	L	P	P	S	P	P	P	S	P	S	L	L		
605	Staf	4	L	P	P	S	P	P	P	S	P	L	L	P	P	S	P	P	P	S	P	S	L	P	P	P	P	P	P	S	P	S	L		

Tabel 6.11 Jumlah Staf per Shift

P	S	M	MS	LM	L	CT
16	11	0	0	0	4	0
17	9	0	0	2	5	0
16	11	0	0	3	4	0
18	9	0	0	3	4	0
20	7	0	0	3	4	0
19	7	0	0	3	5	0

6) Hasil Optimasi Penjadwalan Divisi SIM RM

Tabel 6.12 Hasil Optimasi Divisi SIM & RM

ID Staff	Skill/Pekerjaan	ID Skill	Tanggal																																					
			J	Sb	M	Sn	Sl	Rb	K	J	Sb	M	Sn	Sl	Rb	K	J	Sb	M	Sn	Sl	Rb	K	J	Sb	M	Sn	Sl	Rb	K	J	Sb	M							
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31							
301	Staf	4	P	L	S	S	MS	MS	M	M	LM	L	P	P	P	L	S	S	MS	MS	M	M	LM	L	P	P	P	L	S	S	MS	MS	M							
302	Staf	4	P	P	P	L	S	S	MS	MS	M	M	LM	L	P	P	P	L	S	S	MS	MS	M	M	LM	L	P	P	P	L	S	S	MS							
303	Staf	4	LM	L	P	P	P	L	S	S	MS	MS	M	M	LM	L	P	P	P	L	S	S	MS	MS	M	M	LM	M	P	P	P	L	S							
304	Staf	4	M	M	LM	L	P	P	P	L	S	S	MS	MS	M	M	LM	L	P	P	P	L	S	S	MS	MS	M	L	LM	L	P	P	P							
305	Staf	4	MS	MS	M	M	LM	L	P	P	P	L	S	S	MS	MS	M	M	LM	L	P	P	P	L	S	S	MS	MS	M	M	LM	L	P							
306	Staf	4	S	S	P	MS	M	M	LM	L	P	P	P	L	S	S	P	L	M	M	LM	L	P	P	P	L	S	S	P	MS	M	M	LM							

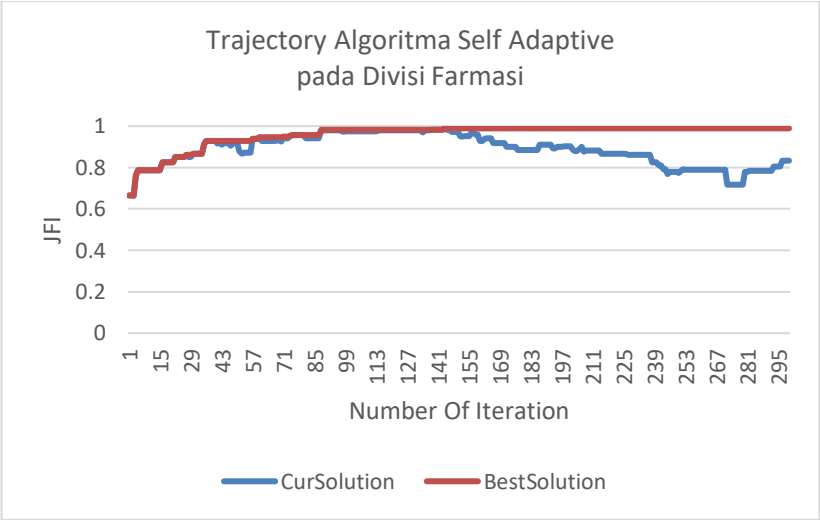
Tabel 6.13 Jumlah Staff per Shift

P	S	M	MS	LM	L	CT
7	6	5	6	2	5	0
9	6	4	5	2	5	0
9	5	5	4	3	5	0
9	4	5	4	3	6	0
7	4	6	6	3	5	0
9	6	6	2	3	5	0

6.2.1. Trajectory Diagram

Trajectory diagram adalah diagram untuk menampilkan perbandingan jumlah dari current solution dan Best Solution pada jumlah iterasi pada algoritma *Self Adaptive Learning Hyper Heuristic*

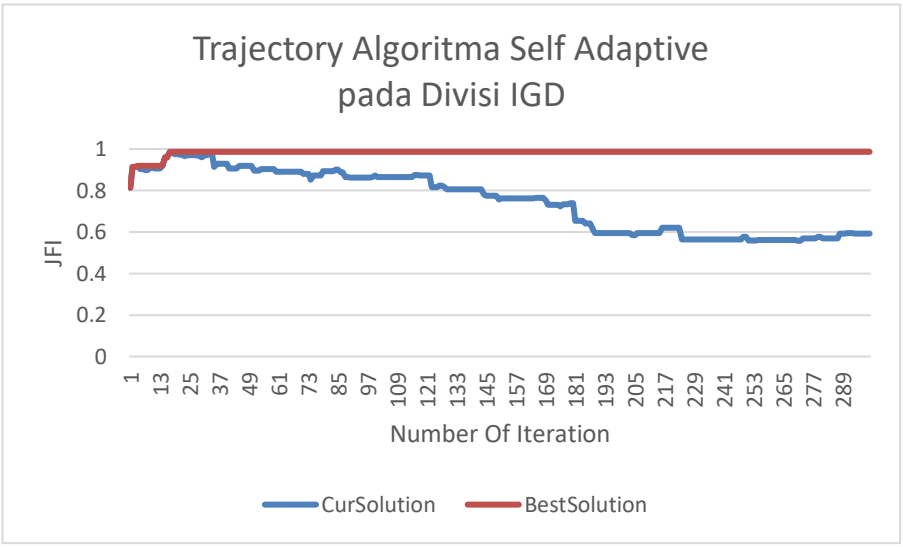
1. Divisi Farmasi



Gambar 6.7 Trajectory Diagram Algoritma Self Adaptive pada Divisi Farmasi

Dapat dilihat diagram trajectory dari Algoritma Self Adaptive dimana nilai bestSolution dicapai pada rentang iterasi ke 85-99 setelah itu baru menurun untuk currentSolution sehingga algoritma ini dapat menemukan solusi terbaik jika dilakukan sebanyak sekitar 100 kali iterasi untuk mendapatkan solusi tersbaik pada divisi Farmasi

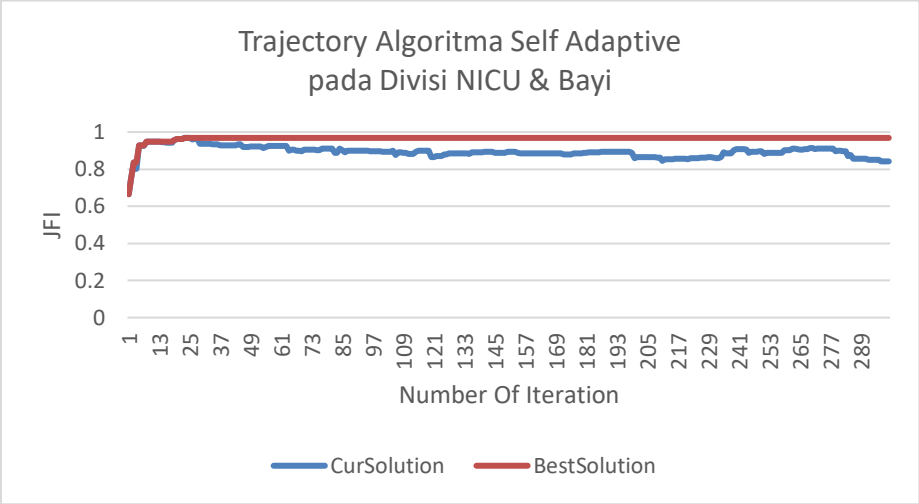
2. Divisi IGD



Gambar 6.8 Trajectory Diagram Algoritma Self Adaptive pada Divisi IGD

Dapat dilihat diagram trajectory dari Algoritma Self Adaptive dimana nilai bestSolution dicapai pada rentang iterasi ke 13-37 setelah itu baru menurun untuk currentSolution sehingga algoritma ini dapat menemukan solusi terbaik jika dilakukan sebanyak sekitar 50 kali iterasi untuk mendapatkan solusi tersbaik pada divisi IGD

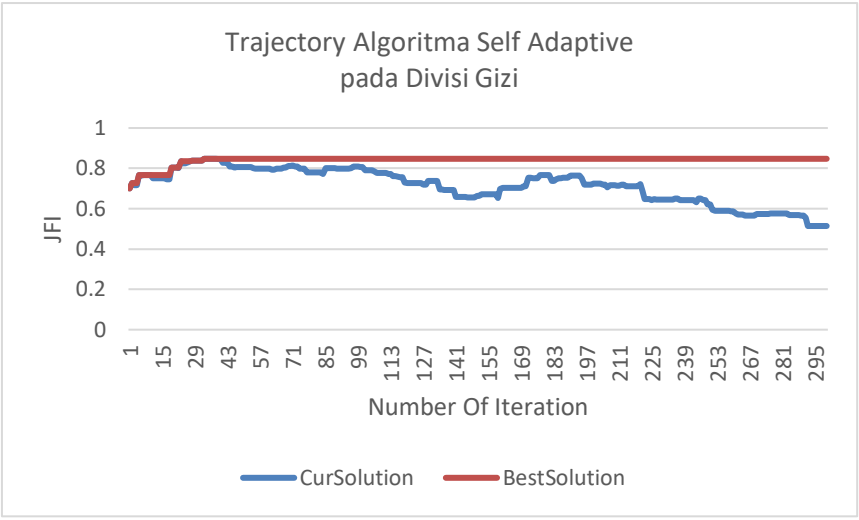
3. Divisi Bayi NICU



Gambar 6.9 Trajectory Diagram Algoritma Self Adaptive divisi NICU Bayi

Dapat dilihat diagram trajectory dari Algoritma Self Adaptive dimana nilai bestSolution dicapai pada rentang iterasi ke 25-37 setelah itu baru menurun untuk currentSolution sehingga algoritma ini dapat menemukan solusi terbaik jika dilakukan sebanyak sekitar 50 kali iterasi untuk mendapatkan solusi tersbaik pada divisi Bayi NICU

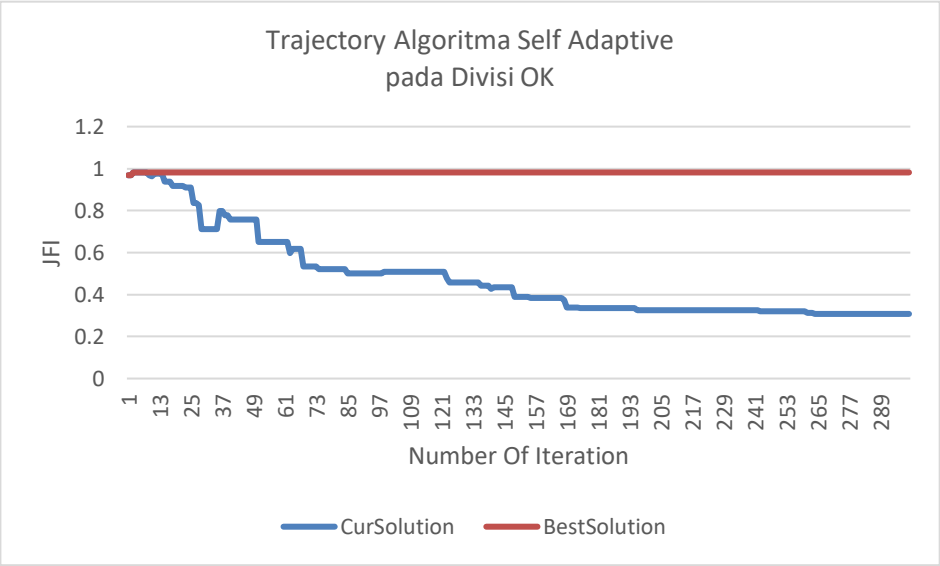
4. Divisi Gizi



Gambar 6.10 Trajectory DIagram Algoritma Self Adaptive pada Divisi Gizi

Dapat dilihat diagram trajectory dari Algoritma Self Adaptive dimana nilai bestSolution dicapai pada rentang iterasi ke 29-43 setelah itu baru menurun untuk currentSolution sehingga algoritma ini dapat menemukan solusi terbaik jika dilakukan sebanyak sekitar 50 kali iterasi untuk mendapatkan solusi tersbaik pada divisi Gizi

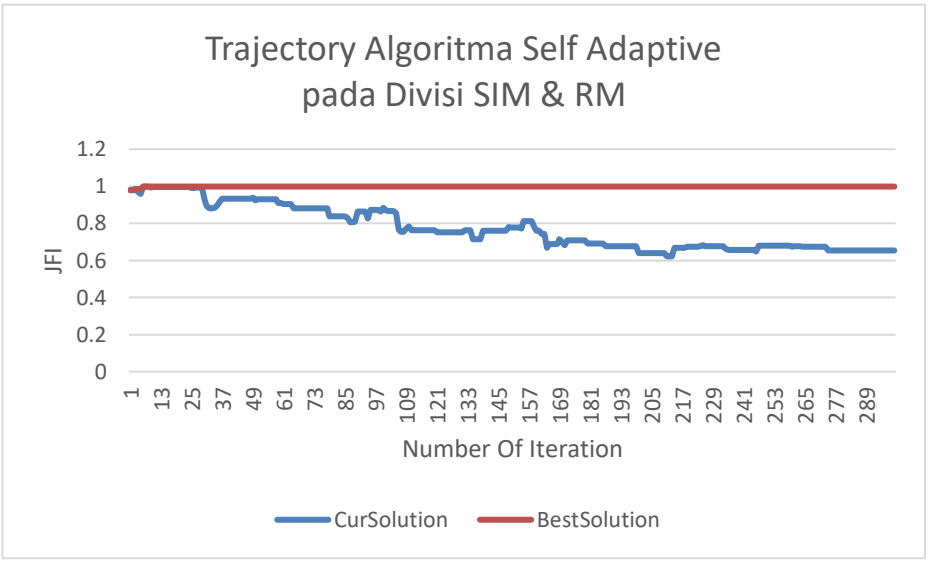
5. Divisi OK



Gambar 6.11 Trajectory Diagram Algoritma Self Adaptive pada Divisi OK

Dapat dilihat diagram trajectory dari Algoritma Self Adaptive dimana nilai bestSolution dicapai pada rentang iterasi awal hal ini diakibatkan karena jadwal yang telah degenerate secara otomatis memiliki nilai JFI yang sudah sangat bagus dan juga dikarenakan tipe shift yang ada di OK tidak ada Middle Shift yang berarti LLH 1 yaitu Move tidak dapat dilakukan dikarenakan tidak bisa mengubah Middle Shift menjadi libur.

6. Divisi SIM & RM



Gambar 6.12 Trajectory Diagram Self Adaptive Divisi SIM & RM

Dapat dilihat diagram trajectory dari Algoritma Self Adaptive dimana nilai bestSolution dicapai mulai iterasi awal, ini diakibatkan karena jadwal yang telah degenerate secara otomatis telah memiliki nilai JFI yang sangat baik sehingga jika dioptimalkan menggunakan algoritma Self Adaptive tidak terlalu berpengaruh

6.3. Analisis Hasil Optimasi

Berdasarkan hasil Optimasi yang didapatkan pada sub bab **6.2** berikut ini adalah hasil pembahasan dari hasil optimasi masing-masing divisi :

1) Divisi Farmasi

Tabel 6.14 Jumlah Hari Libur dan Bobot Farmasi

Bobot	M	Sb	Sn	Sl	Rb	K	J
	4	2	1	1	1	1	1
Staff	Libur						
1	1	1	0	1	0	1	1
2	1	1	1	1	0	1	0
3	0	2	1	0	1	1	0
4	0	1	1	1	1	1	1
5	1	1	1	0	1	0	1
6	1	1	0	1	1	0	1

Berdasarkan data yang ada ada sub bab sebelumnya, tabel diatas digunakan untuk menghitung Jumlah bobot libur yang didapatkan pada tiap staff yang nantinya akan menghasilkan nilai JFI, dari data diatas didapatkan nilai JFI sebesar 0.987361769

2) Divisi IGD

Tabel 6.15 Jumlah Hari Libur dan Bobot IGD

	M	Sb	Sn	Sl	Rb	K	J	Jumlah Libur
Bobot	4	2	1	1	1	1	1	
Staff	Libur							
1	1	0	2	0	1	1	0	5
2	1	0	0	1	0	1	2	5
3	1	2	0	1	0	0	1	5
4	1	0	2	1	1	0	1	6
5	0	0	0	2	2	2	0	6
6	0	1	0	0	0	2	2	5
7	2	2	0	0	0	0	1	5
8	1	0	1	1	1	1	0	5

Berdasarkan data yang ada ada sub bab sebelumnya, tabel diatas digunakan untuk menghitung Jumlah bobot libur yang didapatkan pada tiap staff yang nantinya

akan menghasilkan nilai JFI, dari data diatas didapatkan nilai JFI sebesar 0.941368

3) Divisi Bayi & NICU

Tabel 6.16 Jumlah Hari Libur dan Bobot Divisi Bayi & NICU

	M	Sb	Sn	Sl	Rb	K	J	Jumlah Libur
Bobot	4	2	1	1	1	1	1	
Staff	Libur							
1	5	0	0	0	0	0	0	5
2	1	2	0	0	0	2	1	6
3	1	1	1	0	0	1	1	5
4	2	0	0	0	1	2	2	7
5	1	2	1	0	1	0	0	5
6	0	2	2	1	0	0	1	6
7	0	1	2	1	1	1	1	7
8	1	0	0	2	1	1	0	5
9	3	0	0	0	0	1	1	5
10	1	2	0	1	1	0	0	5
11	1	0	1	1	0	2	0	5
12	1	0	2	1	1	1	2	8
12	0	2	0	1	2	0	1	6

Berdasarkan data yang ada ada sub bab sebelumnya, tabel diatas digunakan untuk menghitung Jumlah bobot libur yang didapatkan pada tiap staff yang nantinya akan menghasilkan nilai JFI, dari data diatas didapatkan nilai JFI sebesar 0.911021814

4) Divisi Gizi

Tabel 6.17 Jumlah Hari Libur dan Bobot Gizi

	M	Sb	Sn	Sl	Rb	K	J	
Bobot	4	2	1	1	1	1	1	Jumlah
Staff	Libur							Libur
1	4	0	1	0	0	0	1	6
2	0	1	0	1	0	0	0	2
3	1	0	2	1	0	1	1	6
4	1	1	0	0	0	0	0	2
5	1	1	0	1	1	0	1	5
6	1	0	0	0	1	1	1	4
7	1	1	1	0	1	0	1	5
8	1	1	0	0	1	0	0	3
9	1	1	1	1	0	0	0	4
10	0	0	1	1	0	3	0	5
11	0	0	1	0	0	1	0	2
12	1	2	0	0	1	1	0	5
13	0	0	0	0	0	2	1	3
14	1	1	0	0	0	0	0	2
15	1	1	0	1	1	1	1	6
16	2	0	2	0	0	0	0	4
17	1	0	1	0	0	0	1	3
18	2	0	2	1	0	0	0	5
19	2	0	1	0	1	0	0	4

Berdasarkan data yang ada ada sub bab sebelumnya, tabel diatas digunakan untuk menghitung Jumlah bobot libur yang didapatkan pada tiap staff yang nantinya akan menghasilkan nilai JFI, dari data diatas didapatkan nilai JFI 0.831653868

5) Divisi OK

Tabel 6.18 Jumlah Hari Libur dan Bobot OK

	M	Sb	Sn	Sl	Rb	K	J	Jumlah Libur
Bobot	4	2	1	1	1	1	1	
Staff	Libur							
1	2	1	0	0	1	0	0	4
2	2	0	0	1	1	1	0	5
3	1	0	1	0	0	1	1	4
4	1	1	0	1	0	0	1	4
5	1	1	0	0	1	0	0	3
6	1	0	1	0	0	1	1	4

Berdasarkan data yang ada ada sub bab sebelumnya, tabel diatas digunakan untuk menghitung Jumlah bobot libur yang didapatkan pada tiap staff yang nantinya akan menghasilkan nilai JFI, dari data diatas didapatkan nilai JFI 0.956953642

6) Divisi SIM & RM

Tabel 6.19 Jumlah Hari Libur dan Bobot SIM & RM

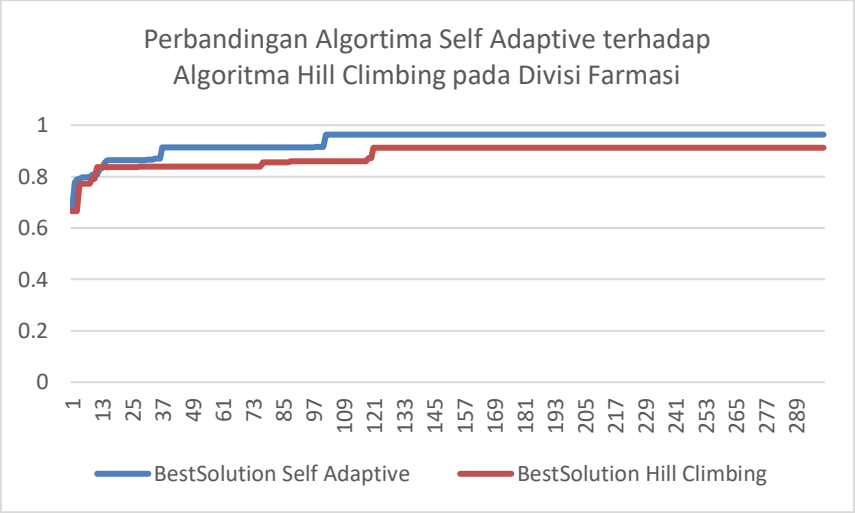
Bobot	M	Sb	Sn	Sl	Rb	K	J	Jumlah Libur
	4	2	1	1	1	1	1	
Staff	Libur							
1	1	1	0	1	0	1	1	5
2	1	1	1	1	0	1	0	5
3	0	2	1	0	1	1	0	5
4	0	1	1	1	1	1	1	6
5	1	1	1	0	1	0	1	5
6	1	1	0	1	1	0	1	5

Berdasarkan data yang ada ada sub bab sebelumnya, tabel diatas digunakan untuk menghitung Jumlah bobot libur yang didapatkan pada tiap staff yang nantinya akan menghasilkan nilai JFI, dari data diatas didapatkan nilai JFI 0.987361769

6.4. Pengujian Algoritma

6.4.1. Perbandingan Algoritma Self Adaptive dengan Algoritma Hill Climbing Hyper Heuristic

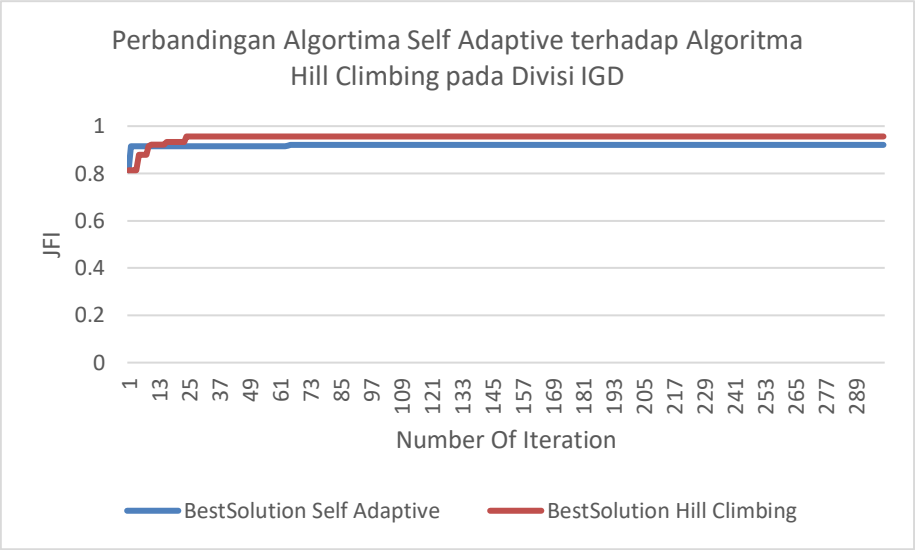
1. Divisi Farmasi



Gambar 6.13 Perbandingan Algoritma Self Adaptive dan Hill Climbing Divisi Farmasi

Dari gambar diatas dapat dilihat perbandingan dari algoritma selfa adaptive dengan hill climbing yang diterapkan di divisi farmasi, berdasarkan grafik untuk best solution dari self adaptive lebih cepat didapatkan daripada di hill climbing serta untuk nilai dari JFI nya juga lebih baik daripada hill climbing.

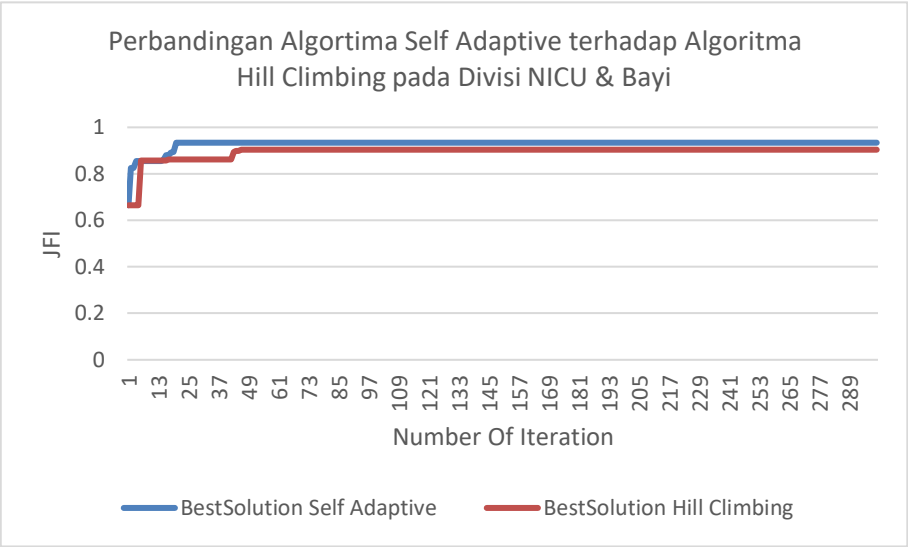
2. Divisi IGD



Gambar 6.14 Perbandingan Algoritma Self Adaptive dan Hill Climbing Divisi IGD

Dari gambar diatas dapat dilihat perbandingan dari algoritma self adaptive dengan hill climbing yang diterapkan di divisi IGD dalam penerapannya di iterasi awal, algoritma self adaptive menemukan nilai JFI yang lebih tinggi, namun setelah iterasi ke 30 algoritma hill climbing memiliki nilai yang lebih baik, sehingga pada jadwal divisi IGD algoritma self adaptive sedikit tidak lebih baik daripada hill climbing.

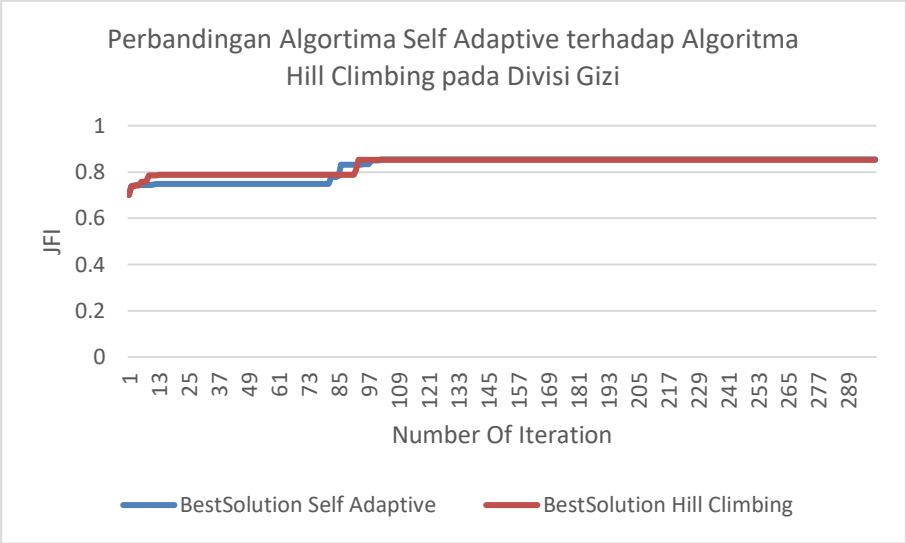
3. Divisi Bayi & NICU



Gambar 6.15 Perbandingan Algoritma Self Adaptive dan Hill Climbing Divisi Bayi & NICU

Dari gambar diatas dapat dilihat perbandingan dari algoritma selfa adaptive dengan hill climbing yang diterapkan di divisi Bayi & NICU, berdasarkan grafik untuk best solution dari self adaptive lebih cepat didapatkan daripada di hill climbing serta untuk nilai dari JFI nya juga lebih baik daripada hill climbing.

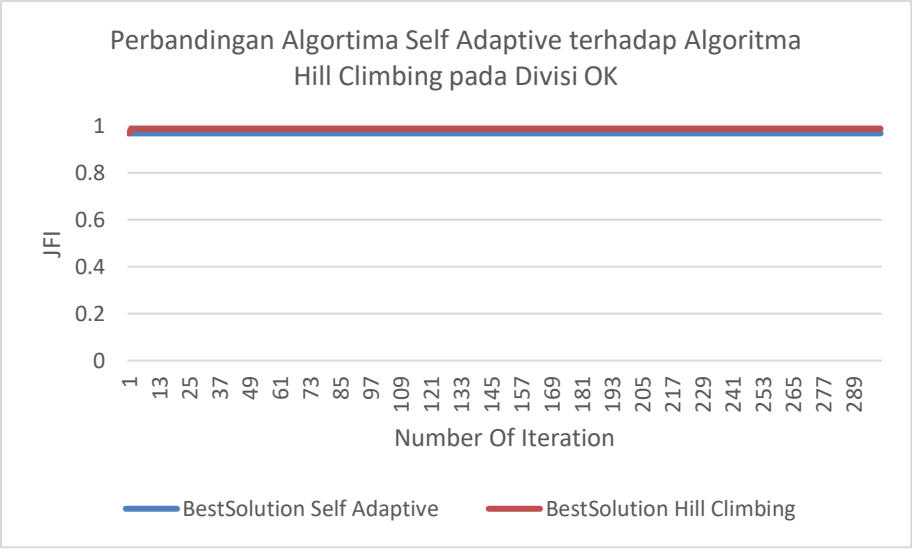
4. Divisi Gizi



Gambar 6.16 Perbandingan Algoritma Self Adaptive dan Hill Climbing Divisi Gizi

Dari gambar diatas dapat dilihat perbandingan dari algoritma self adaptive dengan hill climbing yang diterapkan di divisi Gizi, berdasarkan grafik untuk best solution dari self adaptive lebih cepat didapatkan daripada algoritma hill climbing walaupun akhirnya menemukan nilai JFI yang kurang lebih sama.

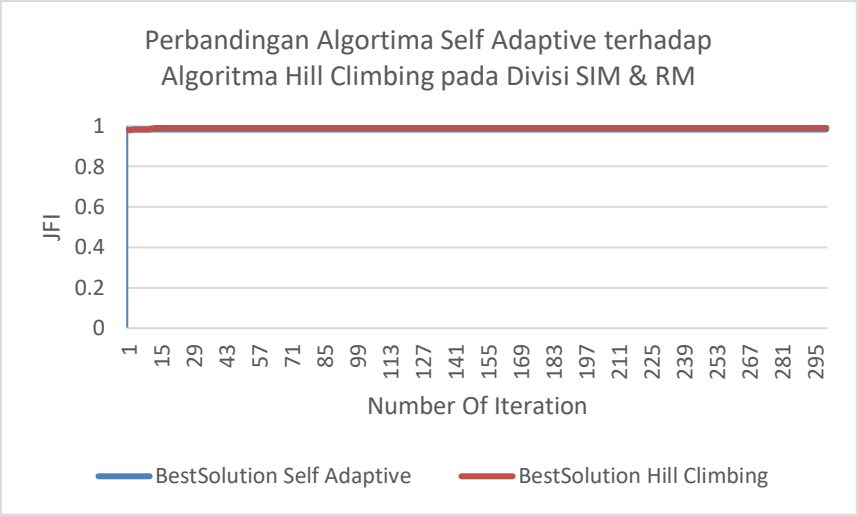
5. Divisi OK



Gambar 6.17 Perbandingan Algoritma Self Adaptive dan Hill Climbing Divisi OK

Dari Gambar diatas dapat dilihat perbandingan dari algoritma self adaptive dengan hill climbing yang diterapkan di divisi OK, dari iterasi awal memang sudah bagus dikarenakan ada low level heuristic yang tidak dapat dieksekusi yaitu LLH 1, move dimana harus ada shift MS dalam jadwal, namun karena tidak ada

6. Divisi SIM RM

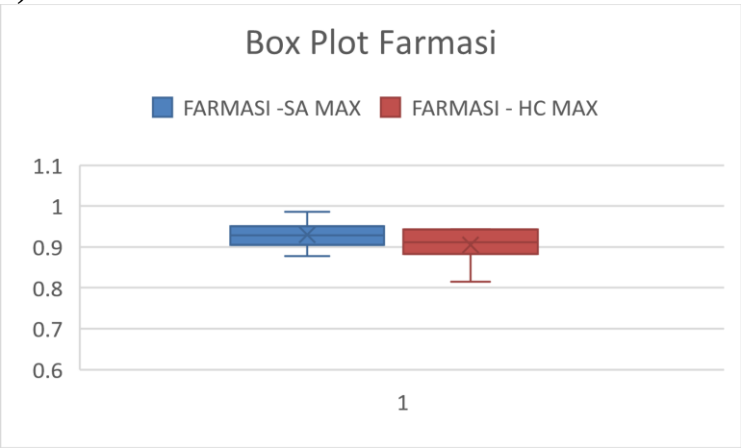


Gambar 6.18 Perbandingan Algoritma Self Adaptive dan Hill Climbing Divisi SIM & RM

Dari Gambar diatas dapat dilihat perbandingan dari algoritma self adaptive dengan hill climbing yang diterapkan di divisi OK, dari iterasi awal memang sudah bagus, hal ini diperkirakan karena hasil dari generate awal sudah memiliki nilai JFI yang bagus sehingga tidak dapat dilakukan optimasi kembali.

6.4.2. Box Plot Diagram

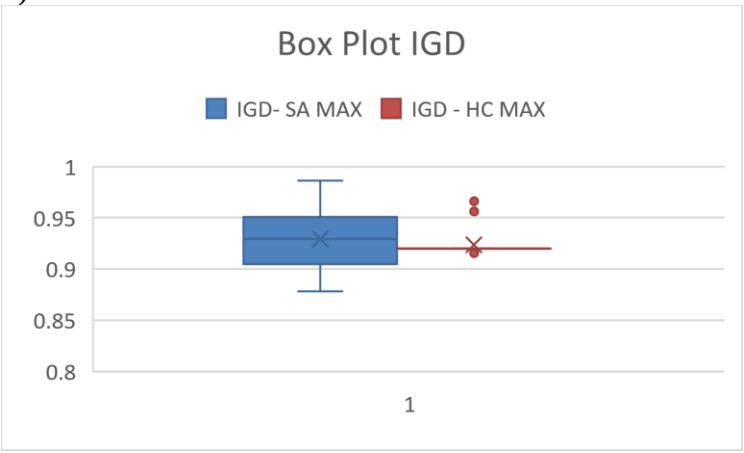
1) Divisi Farmasi



Gambar 6.19 Box Plot Diagram Divisi Farmasi

Gambar diatas adalah diagram box plot untuk melihat persebaran data dari best solution yang telah didapatkan setelah dilakukan percobaan sebanyak 21 iterasi, seperti yang bisa dilihat di gambar yaitu grafik menunjukan persebaran data maksimal dan minimal dari 2(dua) algoritma tidak begitu jauh dimana untuk Self adaptive memiliki persebaran data pada kisaran 0.9 keatas dan algoritma hill climbing memiliki persebaran data diantara 0.8 – 0.9 namun untuk nilai maksimal masih dibawah algoritma self adaptive.

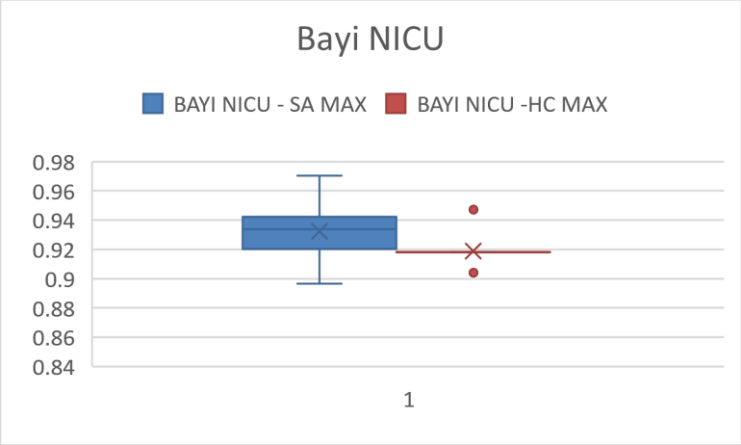
2) Divisi IGD



Gambar 6.20 Box Plot Diagram Divisi IGD

Gambar diatas adalah diagram box plot untuk melihat persebaran data dari best solution yang telah didapatkan setelah dilakukan percobaan sebanyak 21 iterasi, seperti yang bisa dilihat di gambar yaitu grafik menunjukan persebaran data maksimal dan minimal dari 2(dua) algoritma tidak begitu jauh dimana untuk Self adaptive memiliki persebaran data pada kisaran 0.9 - 0.95 dan algoritma hill climbing memiliki persebaran data diantara 0.9 yang berarti memiliki nilai yang sama namun untuk nilai maksimal masih ada diatas algoritma self adaptive namun hanya terdapat 1 data (outlier) dan persebaran dari data tidak merata

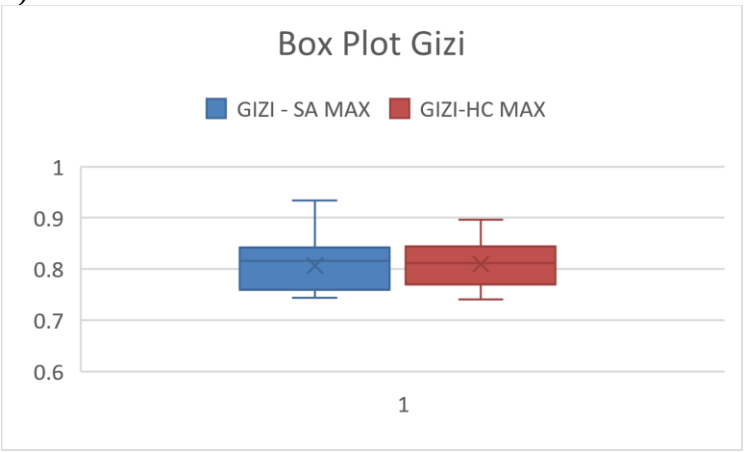
3) Divisi Bayi & NICU



Gambar 6.21 Box Plot Diagram Divisi Bayi & NICU

Gambar diatas adalah diagram box plot untuk melihat persebaran data dari best solution yang telah didapatkan setelah dilakukan percobaan sebanyak 21 iterasi, seperti yang bisa dilihat di gambar yaitu grafik menunjukan persebaran data maksimal dan minimal dari 2(dua) algoritma tidak begitu jauh dimana untuk Self adaptive memiliki persebaran data pada kisaran 0.92 - 0.94 dan algoritma hill climbing memiliki persebaran data diantara 0.92 yang berarti memiliki nilai yang sama untuk semua datanya namun untuk nilai maksimal masih ada diatas algoritma self adaptive namun hanya terdapat 1 data (outlier) dan persebaran dari data tidak merata.

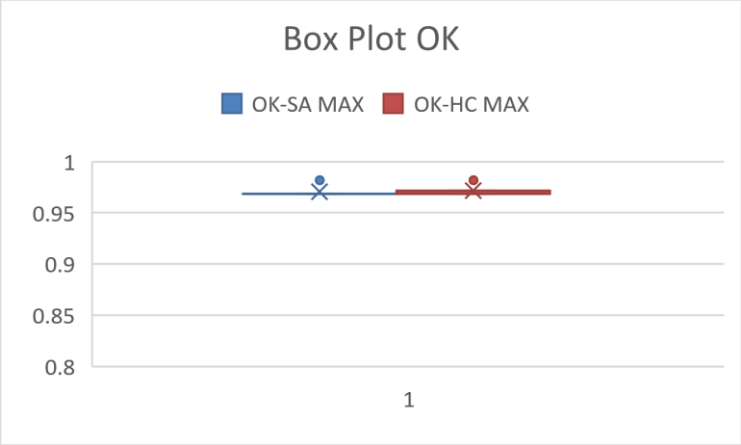
4) Divisi Gizi



Gambar 6.22 Box Plot Diagram Divisi Gizi

Gambar diatas adalah diagram box plot untuk melihat persebaran data dari best solution yang telah didapatkan setelah dilakukan percobaan sebanyak 21 iterasi, seperti yang bisa dilihat di gambar yaitu grafik menunjukan persebaran data maksimal dan minimal dari 2(dua) algoritma tidak begitu jauh dimana untuk Self adaptive memiliki persebaran data pada kisaran 0.7 – 0.8 keatas dan algoritma hill climbing memiliki persebaran data diantara 0.8 – 0.9 namun untuk nilai maksimal masih dibawah algoritma self adaptive.

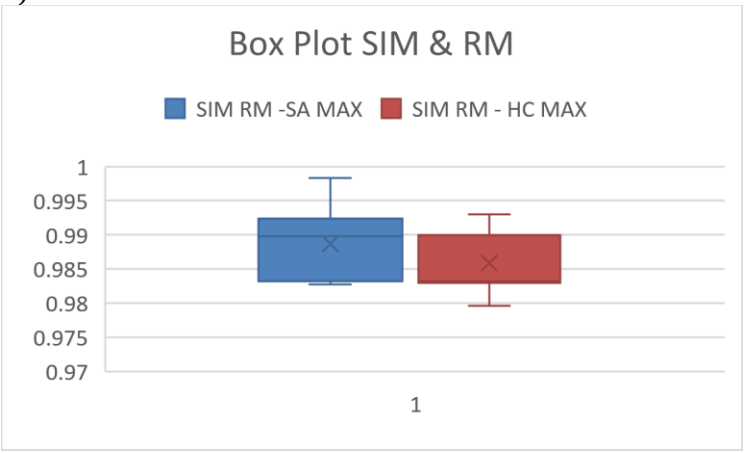
5) Divisi OK



Gambar 6.23 Box Plot Diagram Divisi OK

Gambar diatas adalah diagram box plot untuk melihat persebaran data dari best solution yang telah didapatkan setelah dilakukan percobaan sebanyak 21 iterasi, seperti yang bisa dilihat di gambar yaitu grafik menunjukan persebaran data maksimal dan minimal dari 2(dua) algoritma tidak begitu jauh dimana untuk Self adaptive memiliki persebaran data pada kisaran 0.8 dan hasil dari semua iterasi memiliki nilai yang kebanyakan sama sehingga box plot terbentuk seperti gambar diatas untuk algoritma self adaptive dan algoritma hill climbing memiliki persebaran data diantara 0.8 - 0.9 dan memiliki nilai maksimal yang mendekati nilai maksimal dari algoritma self adaptive.

6) Divisi SIM & RM



Gambar 6.24 Box Plot Diagram Divisi SIM & RM

Gambar diatas adalah diagram box plot untuk melihat persebaran data dari best solution yang telah didapatkan setelah dilakukan percobaan sebanyak 21 iterasi, seperti yang bisa dilihat di gambar yaitu grafik menunjukan persebaran data maksimal dan minimal dari 2(dua) algoritma tidak begitu jauh dimana untuk Self adaptive memiliki persebaran data pada kisaran 0.985 - 0.995 dan algoritma hill climbing memiliki persebaran data diantara 0.98 – 0.99 untuk nilai bestSolution.

6.4.3. Statistik

Statistik yang digunakan pada penelitian ini dilakukan berdasarkan 21 kali *run* dan setiap *run* dilakukan 3 kali iterasi dan megambil nilai dari bestSolution.

1) Divisi Farmasi

Tabel 6.20 Statistik Hill Climbing Divisi Farmasi

MAX	MIN	AVERAGE
0.943129	0.665158	0.893644491

Tabel 6.21 Statistik Self Adaptive Divisi Farmasi

MAX	MIN	AVERAGE
0.989322	0.665158	0.887286

Pada divisi Farmasi dapat dilihat bahwa Algoritma Self Adaptive Learning mendapatkan nilai maksimal yang berbeda dimana algoritma self adaptive learning memiliki nilai maksimal yang lebih tinggi daripada hill climbing serta memiliki rata-rata yang lebih tinggi juga sehingga dapat dikatakan algoritma Self Adaptive berjalan dengan baik dalam mengoptimalkan penjadwalan.

2) Divisi IGD

Tabel 6.22 Statistik Hill Climbing Divisi IGD

MAX	MIN	AVERAGE
0.966088	0.813278	0.921752

Tabel 6.23 Statistik Self Adaptive Divisi IGD

MAX	MIN	AVERAGE
0.986339	0.813278	0.926342

Pada divisi IGD dapat dilihat bahwa Algoritma Self Adaptive memiliki nilai Max dan Average yang lebih besar dibandingkan dengan Hill Climbing maka dari itu

dapat disimpulkan pada divisi IGD algoritma self adaptive berjalan lebih baik dalam mengoptimalkan penjadwalan

3) Divisi Bayi & NICU

Tabel 6.24 Statistik Hill Climbing Divisi Bayi NICU

MAX	MIN	AVERAGE
0.947043	0.664773	0.91540552

Tabel 6.25 Statistik Self Adaptive Divisi Bayi NICU

MAX	MIN	AVERAGE
0.970241	0.664773	0.921799

Pada divisi Bayi NICU dapat dilihat bahwa Algoritma Self Adaptive memiliki nilai Max dan Average yang lebih besar dibandingkan dengan Hill Climbing maka dari itu dapat disimpulkan pada divisi Bayi NICU algoritma self adaptive berjalan lebih baik dalam mengoptimalkan penjadwalan

4) Divisi Gizi

Tabel 6.26 Statistik Hill Climbing Divisi Gizi

MAX	MIN	AVERAGE
0.896015	0.70004	0.803437

Tabel 6.27 Statistik Self Adaptive Divisi Gizi

MAX	MIN	AVERAGE
0.933895	0.664773	0.800893

Pada divisi Gizi dapat dilihat bahwa Algoritma Self Adaptive memiliki nilai Max yang lebih besar namun nilai minimal dan average masih lebih besar Hill Climbing dibandingkan dengan Self Adaptive, mulai dari nilai minimum, maksimum dan rata-rata sehingga

pada divisi Gizi algoritma self adaptive tidak bekerja lebih baik daripada hill climbing.

5) Divisi OK

Tabel 6.28 Statistik Hill Climbing Divisi OK

MAX	MIN	AVERAGE
0.988359	0.968559	0.97151051

Tabel 6.29 Statistik Self Adaptive Divisi IGD

MAX	MIN	AVERAGE
0.981669	0.968559	0.970421

Pada divisi OK dapat dilihat bahwa Algoritma Self Adaptive memiliki nilai Max yang lebih besar namun nilai minimal dan average masih lebih besar Hill Climbing dibandingkan dengan Self Adaptive, mulai dari nilai minimum, maksimum dan rata-rata sehingga pada divisi OK algoritma self adaptive tidak bekerja lebih baik daripada hill climbing.

6) Divisi SIM & RM

Tabel 6.30 Statistik Hill Climbing Divisi SIM & RM

MAX	MIN	AVERAGE
0.992969	0.979592	0.98558319

Tabel 6.31 Statistik Self Adaptive SIM & RM

MAX	MIN	AVERAGE
0.99835	0.979592	0.98826

Pada divisi SIM & RM dapat dilihat bahwa Algoritma Self Adaptive memiliki nilai Max dan Average yang lebih besar dibandingkan dengan Hill Climbing maka dari itu dapat disimpulkan pada divisi SIM & RM algoritma self adaptive berjalan lebih baik dalam mengoptimalkan penjadwalan

6.5. Perbandingan JFI Optimasi dengan JFI Jadwal *Existing*

Berikut ini adalah hasil penghitungan Jain Fairness Index untuk setiap

1) Divisi Farmasi

Tabel 6.32 Perbandingan JFI Divisi Farmasi

Manual (Existing)	Generate Otomatis	Optimasi
0.923664	0.6651	0.9467

2) Divisi IGD

Tabel 6.33 Perbandingan JFI Divisi IGD

Manual (Existing)	Generate Otomatis	Optimasi
0.918403	0.812327	0.943067

3) Divisi Bayi & NICU

Tabel 6.34 Perbandingan JFI Divisi Bayi dan NICU

Manual	Generate Otomatis	Optimasi
0.892292	0.6647	0.9334

4) Divisi Gizi

Tabel 6.35 Perbandingan JFI Divisi Gizi

Manual	Generate Otomatis	Optimasi
0.851644	0.70004	0.83416

5) Divisi OK

Tabel 6.36 Perbandingan JFI Divisi OK

Manual	Generate Otomatis	Optimasi
0.8069	0.9685	0.9685

6) Divisi SIM & RM

Tabel 6.37 Perbandingan JFI Divisi SIM & RM

Manual	Generate Otomatis	Optimasi
0.9009	0.9795	0.9795

Berdasarkan hasil otomasi dan optimasi jika dibandingkan dengan jadwal manual memiliki perbedan pada tiap divisi, dikarenakan karakter dari masing-masing divisi yang berbeda sehingga otomasi dan optimasi memiliki hasil yang berbeda juga, namun rata-rata setelah dilakukan optimasi, nilai dari *Jain Fairness Index* pada tiap divisi dapat meningkat.

(Halaman ini sengaja dikosongkan)

BAB VII
KESIMPULAN DAN SARAN

Bab ini berisikan kesimpulan dari hasil penelitian dan juga saran perbaikan untuk penelitian kedepannya.

7.1. Kesimpulan

Berdasarkan Uji Coba yang telah dilakukan dalam tugas akhir ini, maka kesimpulan yang dapat diambil adalah :

- Dapat membandingkan fairness antara staff rumah sakit dengan melihat hasil Jain fairness index yang di hasilkan oleh penjadwalan manual dan penjadwalan otomatis yang telah di optimasi menggunakan metode hyperheuristic Self Adaptive Learning. Jadwal yang telah optimal tersebut dipilih dari hasil jain fairness Index yang paling optimal dengan menggunakan Hard Constraint yang tidak boleh dilanggar dan Soft Constraint yang ada dilanggar dan ada yang terpenuhi. Pembuatan jadwal optimal menggunakan Bahasa pemrograman java dan tool netbeans 8.02 untuk mengimplementasikan algortima *Self Adaptive Learning Hyper Heuristc*. Model penjadwalan yang di temukan dapat menyelesaikan permalasahan pada studi kasus dengan tercapaiannya tujuan yang sudah di tentukan
- Dari hasil pengujian ditemukan perubahan nilai JFI dari masing-masing unit untuk divisi Farmasi hasil sebelum optimasi dari 0.92 menjadi 0.94 setelah optimasi , divisi IGD sebesar 0.91 menjadi 0.94 , divisi Bayi & NICU dari 0.89 menjadi 0.93, divisi Gizi dari 0.85 menjadi 0.83, divisi OK dari 0.80 menjadi 0.96 ,dan divisi SIM & RM dari 0.90 menjadi 0.97
- Dari hasil pengujian dapat dilihat bahwa rata-rata dari algoritma Self Adaptive memiliki keunggulan dari nilai JFI yang ada di masing masing divisi mulai dari nilai maksimum, minimum dan juga rata-rata dari best

solution yang didapat, namun ada beberapa juga yang tidak dapat dioptimalkan dengan baik oleh algoritma Self Adaptive seperti pada divisi Gizi.

7.2. Saran

Berdasarkan hasil penelitian tugas akhir ini, peneliti memiliki beberapa saran untuk penelitian selanjutnya, yaitu :

1. Pada penelitian ini, jadwal sebelumnya hanya dijadikan acuan dan pencarian pola untuk pembuatan jadwal, jadi belum mengambil data jadwal bulan sebelumnya untuk *generate* jadwal bulan selanjutnya, sehingga sangat disarankan untuk penelitian selanjutnya dapat mengambil data bulan sebelumnya untuk melakukan *generate* jadwal secara otomatis
2. Penggunaan *low level heuristic* yang lebih banyak, dikarenakan dalam penelitian ini LLH yang digunakan hanya 2, yaitu move dan swap sehingga memiliki keragaman yang sedikit, untuk kedepannya diharapkan dapat membuat lebih dari 2 *low level heuristic* sehingga algoritma yang digunakan menjadi lebih baik dan bervariasi.
3. Untuk menghitung keadilan atau *Fairness* hanya menggunakan hari libur, seharusnya bisa menambahkan hal lain seperti jumlah shift malam, atau jumlah shift yang lain yang dijadikan acuan sebagai seberapa adil sebuah jadwal di rumah sakit tersebut.
4. Adanya nilai preferensi yang lebih detail terkait dengan mendukung penentuan kriteria dari *Jain Fairness Index* seperti preferensi perawat yang dapat dilaukan dengan pengumpulan kuesioner, jadi untuk keadilan di dalam rumah sakit bisa menggunakan hal lain selain hari libur yang lebih valid.

DAFTAR PUSTAKA

[1] P. Smet, "Nurse rostering: models and algorithms for theory, practice and integration with other problem," 2015.

[2] M. A. Amir, Optimasi Penjadwalan Perawat Menggunakan Gabungan Integer Linear Programming Dan Variable Neighborhood Search. Studi Kasus Instalasi Gawat Darurat Rumah Sakit Ibnu Sina Makassar., 2017.

[3] T. C. Wong, M. Xu and K. S. Chin, "A two-stage heuristic approach for nurse scheduling problem: A case study in an emergency department. Computers & Operations Research, 51, 99-110.," 2014.

[4] A. Muklason, Solver Penjadwal Ujian Otomatis Dengan Algoritma Maximal Clique dan Hyper-heuristics, Surabaya, 2017.

[5] Rumah Sakit Kendangsari, Tentang Rumah Sakit, [ONLINE]Available: http://rsia.kendangsari.com/tentang-kami/ . [Diakses 27 September 2017].

[6] D.-M. W. C. W. R. H. Rajendra K. Jain, "A Quantitative Measure of Fairness and Discrimination of Resource Allocation in Shared Computer System," 1984.

[7] B. CHEANG, H. LI, A. LIM and B. RODRIGUES, "Nurse Rostering Problems: A Bibliographic Survey," 2003.

[8] A. Muklason, "Hyper-Heuristics And Fairness In Examination Timetabling Problems," 2017.

[9] J. Hoffmann, "A Heuristic for Domain Independent Planning and its," Institute for Computer Science, Albert Ludwigs University,, Freiburg, Germany,.

[10] B. B. E. E. K. Ender Özcan, "Hill Climbers and Mutational Heuristics in Hyperheuristic," Yeditepe University, İstanbul, Turkey.

[11] "Class ArrayBlockingQueue," 2017. [Online]. Available: https://docs.oracle.com/javase/7/docs/api/java/util/concurrent/ArrayBlockingQueue.html. [Accessed 10 January 2018].

[12] A. J. P. B. M. E. O. Ahmad Muklason, "Initial Results on Fairness in Examination Timetabling," *6th Multidisciplinary International Conference on Scheduling : Theory and Applications,* 2013.

[13] B. Bilgin, P. D. Causmaecker, B. Rossie and G. V. Berghe, "Local search neighbourhoods for dealing with a novel nurse rostering model," *Ann Oper Res,* 2012.

[14] B. Bilgin, P. D. Causmaecker and G. V. Berghe, "A Hyperheuristic Approach to Belgian Nurse Rostering," *Multidisciplinary International Conference on Scheduling : Theory and Applications (MISTA),* 2009.

[15] E. K. Burke, P. D. Causmaecker, G. V. Berghe And H. V. Landeghem, "The State Of The Art Of Nurse Rostering," *Journal of Scheduling 7: 441…,* 2004.

[16] E. K. Burke, M. Hyde, G. Kendall, G. Ochoa, E. O¨zcan and J. R. Woodward, "A Classification of Hyper-heuristic Approaches".

[17] A. Hasad, "Algoritma Optimasi dan Aplikasinya," 2010.

BIODATA PENULIS

Penulis lahir di Jakarta pada tanggal 28 Februari 1996. Penulis merupakan anak kedua dari tiga bersaudara. Penulis telah menempuh pendidikan formal di sekolah negeri mulai SD Negeri Jatiasih VIII Bekasi, SMP Negeri 9 Bekasi, SMA Negeri 5 Bekasi. Setelah lulus, penulis melanjutkan ke jenjang perguruan tinggi negeri di Surabaya, yakni Jurusan Sistem Informasi Institut Teknologi Sepuluh Nopember Surabaya. Sebagai mahasiswa, penulis aktif dalam urusan organisasi. Tercatat penulis aktif berkontribusi melalui keanggotaan organisasi mahasiswa di Himpunan Sistem Informasi sejak 2014 hingga 2016 dan BEM Fakultas Teknologi Informasi sejak 2015 hingga 2016. Pernulis juga pernah melakukan kerja praktik di PT. Indonesia Comnets Plus di bidang pengembangan dan Implementasi Aplikasi pada bulan Juni hingga Agustus tahun 2017. Untuk mendapatkan gelar Sarjana Komputer (S.Kom), penulis mengambil laboratorium Rekayasa Data dan Intelejensi Bisnis dengan topik tugas akhir Optimasi Penjadwalan dengan studi kasus staff rumah sakit. Untuk kepentingan penelitian penulis juga dapat dihubungi melalui e-mail: fachrurpra yogo@gmail.com@gmail.com.

LAMPIRAN A

HASIL WAWANCARA dan JADWAL EXISTING

HASIL WAWANCARA

1. Informasi interview

Interviewer : Rizka Pordella, Indriarti Kusumanita, Fachrur Zaffrinda, Fata Hirzi, & Zuli Maulidati
Hari, Tanggal : Senin, 28 Oktober 2017
Pukul : 10.30-12.30
Lokasi : RSIA Kendangsari MERR

2. Informasi narasumber

Nama : Bu Sylvy Medtasya Dzykrzyanka, S. Farm, M. Farm. Klin, APT, MARS
Jabatan : Kepala Bagian
Divisi : Sumber Daya Manusia (SDM) & Hukum
Instansi : RSIA Merr Kendangsari
Lama Bekerja : 3.5 tahun

3. Penjelasan interview

Interview ini bertujuan untuk salah satu sumber data untuk tugas akhir dengan judul "Optimasi Penjadwalan Perawat Dengan Menggunakan Algoritma Tabu Search, Late acceptence, Reinforcement Leraning, Self adaptive learning, simulated anealing Based Hyper-Heuristics (Studi Kasus: Rumah Sakit Ibu Dan Anak Kendangsari)" yang dimaksudkan agar peneliti bisa mendapatkan gambaran mengenai permasalahan penjadwalan perawat yang ada di RSIA Kendangsari Surabaya. Dengan melakukan interview ini diharapkan peneliti mendapatkan informasi mengenai penjadwalan perawat yang sudah diterapkan Rumah Sakit Ibu dan Anak kendangsari Surabaya Untuk menjaga dan menjamin kerahasiaan, maka data – data yang bersifat pribadi akan dirahasiakan oleh peneliti.

~~A-~~A-2

List Pertanyaan dan Jawaban Kepada Senior management
Berikut adalah daftar pertanyaan untuk Senior Manager sebagai penanggungjawab proses bisnis yang dijalankan pada RSIA kendangsari Surabaya:

1. Ada berapakan Tipe staff perawat dipekerjakan? (Skill Type)?
   Answer: klasifikasi Perawat dilakukan berdasarkan Pelatihan keterampilan yang dimiliki masing masing perawat. Secara keseluruan perawat yang dimiliki ole RSIA hanya memiliki sampai PK2 (Pelatihan Keterampilan tingkat 2) sedangkan untuk perawat sebior atau kepalaperawat memiliki PK3 (Pelatihan Keterampilan tingkat 3)

2. Berapa jumlah ward atau bangsal yang dimiliki oleh RS?
   Answer : RSIA memiliki 14 unit atau bangsal, terdapat 6 unit RSIA yang memiliki pattern yang unik dalam menjadwalkan staff baik medis maupun non medis yaitu:
   - Kamar Operasi
   - Ruang bayi dan NICU
   - SIM dan RM
   - Instalasi Farmasi
   - Instalasi rawat jalan dan IGD
   - Instalasi Gizi

3. Bagaimana cara pembuatan jadwal perawat saat ini, apakah secara terpusat atau dibagi setiap ward atau bangsal?
   Answer: pembuatan Jadwal dilakukan secara per unit yang dilakukan oleh masing masing PJ unit. PJ unit mengumpulkan jadwal kepada bagian SDM RSIA maksimal tanggal 25 pada setiap bulan untuk bulan berikutnya untuk diinputkan ke dalam sistem mereka.

4. Bagaimanakah aturan umum mengenai penjadwalan perawat? Answer:
    - Staff yang memperoleh shift malam sebanyak maka harus memperoleh libur dihari berikutnya
    - Setiap staff dalam satu bulan harus mendapatkan bagian shift malam
    - Staff memiliki maksimal jam kerja 7 jam per hari dalam 6 hari perminggu
    - jam lembur dihitung minimal 1 jam
5. Siapa yang melakukan penjadwalan perawat?
   Answer : Penjadwalan dilakukan oleh setiap pj unit atau bangsal

6. Berapa kali penjadwalan perawat dilakukan? Apakah setiap minggu, atau bulan?
   Answer: Penjadwalan perawat dilakukan sekali dalam sat bulan

7. Apakah ada rotasi perpindahan perawat yang dilakukan?
   Answer: ada, akan tetapi perpindahan jadwal perawat terseut biasanya tanpa sepengetahuan pihak SDM. Kebanyakan perawat mengganti shift mereka sendiri tanpa melakukan konfirmasi terlebih dahulu kepada pihak SDM.

8. Berapa shift yang dijalankan setiap harinya, dan berapa lama alokasi waktu setiap shift?
   Answer: secara umum terdapat 3 shift dengan pembagian 7 jam yaitu
    - Pagi: 07.00-14.00
    - Sore : 14.00-21.00
    - Malam: 21.00-07.00

untuk mengatasi jam kerja yang sibuk pada jam jam tertentu RSIA menerapkan shift middle yang berlangsung dari jam 10.00-17.00 atau jam 12.00-19.00

9. Berapa maksimum jam kerja yang didapatkan masing masing perawat dalam satu minggu?
   Answer : Tidak ada maksimum jam kerja setiap minggunya. Yang ada hanyalah batas minimal libur untuk setiap bulannya. Setiap bulannya, jatah libur untuk setiap perawat diusahakan harus sama. Apabila tidak sama, maka perawat yang mendapat libur lebih sedikit akan diganti hari liburnya di bulan depan.

10. Bagaimana penjadwalan perawat yang sudah dilakukan saat ini? Apakah terdapat keluhan dari perawat mengenai penjadwalan yang sudah ada?
    Answer: Penjadwalan perawat RSIA saat ini dilakukan secara menual oleh masing masing PJ bangsal atau unit RS. Hal tersbut mengakibatkan memakan banyak waktu.

11. Apa saja kekurangan penjadwalan yang telah diterapkan saat ini? apa yang perlu dioptimalkan misal di tingkat manajemen atau di tingkat perawat?
    Answer: pihak SDM kesulitan untuk mengontrol jalanya penjadwalan dikarenakan banyak dari perawat yang berganti shift kerja tanpa ada izin terlebih dahulu. Hal yang ingin dioptimalkan oleh RS adalah bagaimana cara mengontrol jalannya penjdawalan dengan menggunakan sistem yang langsung terintegrasi langsung dengan fingerprint yang ada sehinga kecurangan atau pelanggaran Perawat akan saling tukar menukar jam dapat diminimalisasi.
    Masalah lain juga adanya jam jam dimana rumah sakit sepi atau sangat ramai. Ketika rumah sakit keadaannya sangat ramai, masalah tersebut biasanya diselesaikan

dengan cara mengoper perawat yang bekerja pada shift tersebut pada unit yang renggang ke unit yang padat pasien. Namun apabila rumah sakit sepi, maka bagian SDM akan meliburkan perawatnya. Misal dokter yang bertugas di RS tersebut harus keluar negeri semuanya untuk mengikuti pekan ilmiah. Melihat hal ini maka SDM berinisiatif untuk meliburkan perawat yang berada pada shift dan unit tersebut.

12. Bagaimana skill yang dimiliki perawat apakah ada perawat khusus yang menangani kasus tertentu (Type skill masing-masing perawat)
    Answer: bagian unit yang harus memiliki perawat skill khusus adalah:
    - Ruang Operasi
    - Rekam Medis

13. Bagaimana jadwal yang sudah diatur, tiba-tiba mengalami perubahan karena salah perawat meminta libur?
    Answer: Jatah ambil cuti harus ditentukan masing masing perawat pada saat PJ unit melakukan penjadwalan. Dalam satu bulan perawat diberikan jatah cuti satu hari. Pengajuan cuti dilakukan perawat maksimal tanggal 25 untuk setiap bulannya. Untuk kasus berbeda, misalkan terdapat keluarga meninggal atau terkena musibah yang lain berarti dapat dikurangi dengan jatah cuti yang sudah dia tentukan pada awal penjadwalan.
14. Apakah ada perawat yang memiliki pengurangan jam kerja? (ex. perawat yang sedang hamil, perawat memiliki keterbatasan)
    Answer: Tidak ada pengurangan jam kerja setiap shiftnya untuk perawat yang hamil. Namun terdapat pemberian cuti selama 3 bulan yaitu 1.5 bulan sebelum

melahirkan dan 1,5 bulan sesudah melahirkan. Pengambilan cuti bersifat fleksibel

JADWAL EXISTING RSIA KENDANGSARI

1) DIVISI FARMASI

No.	ID Staff	Skill/Pekerjaan	ID Skill						Minggu I							Minggu II							Minggu III							Minggu IV			
				R	K	J	Sb	M	Sn	Sl	R	K	J	Sb	M	Sn	Sl	R	K	J	Sb	M	Sn	Sl	R	K	J	Sb	M	Sn	Sl	R	K
				1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
1	101	Apoteker Farmasi	1	P	P	P	P	L	P	P	P	P	P	P	L	P	P	P	P	P	P	L	P	P	P	P	P	P	L	P	P	P	P
2	102	Staff Senior	3	S	MS	S	S	MS	L	MS	MS	MS	S	S	L	S	S	S	S	S	L	MS	S	MS	S	S	MS	L	P	S	MS	S	MS
3	103	Anggota	4	M	M	LM	L	P	S	P	P	M	M	LM	L	CT	P	S	MS	MS	P	CT	MS	M	M	LM	L	P	S	S	S	L	P
4	104	Anggota	4	P	P	L	MS	M	M	LM	L	P	P	CT	S	S	S	M	M	LM	L	P	P	S	S	M	M	LM	L	P	P	S	S
5	105	Anggota	4	LM	L	S	S	S	P	M	M	LM	L	P	MS	P	CT	P	S	M	M	LM	L	P	P	S	S	M	M	LM	L	P	S
6	106	Anggota	4	L	S	M	M	LM	L	S	S	S	CT	S	P	M	M	LM	L	S	S	S	S	S	CT	P	S	S	P	M	M	LM	L
7	107	Anggota	4	S	S	P	P	L	S	S	S	S	S	M	M	LM	L	MS	P	P	S	M	M	LM	L	MS	P	S	S	L	S	M	M

A-A-2

No.	ID Staff	Skill/Pekerjaan	ID Skill						Minggu I							Minggu II							Minggu III							Minggu IV			
				R	K	J	Sb	M	Sn	Sl	R	K	J	Sb	M	Sn	Sl	R	K	J	Sb	M	Sn	Sl	R	K	J	Sb	M	Sn	Sl	R	K
				1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
8	108	Petugas Gudang Obat	2	P	P	P	P	L	P	P	P	P	P	P	L	P	P	P	P	P	P	L	P	P	P	P	P	P	L	P	P	P	P

2) DIVISI IGD

NO	ID staff	TANGGAL																														
		R	K	J	S	M	S	SL	R	K	J	S	M	S	SL	R	K	J	S	M	S	SL	R	K	J	S	M	S	SL	R	K
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
1	401	MD	S	P	L	M	M	LM	MD	S	MD	L	L	S	MD	MS	S	S	P	P	CT	S	MS	S	S	P	L	P	MD	CT	MS

NO	ID staff	TANGGAL																												
		R	K	J	S	M	S	SL	R	K	J	S	M	S	SL	R	K	J	S	M	S	SL	R	K	J	S	M	S	SL	R	K
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
2	402	S	P	S	P	L	P	MD	P	MS	CT	S	S	MS	P	L	CT	M	M	LM	S		S	MS	L	S	P	M	M	LM	L
3	403	L	S	MD	MS	S	CT	S	S	M	M	LM	L	P	S	S	MS	CT	S		S	M	L	P	MD	MS	S	S	S	M	L
4	404	S	S	M	M	LM	L	MS	S	S	S	MD	L	S	S	P	MS	L	MD	M	M	LM	P	MD	S	L	CT	MS	S	S	P

A-A-4

NO	ID staff	TANGGAL																												
		R	K	J	S	M	S	SL	R	K	J	S	M	S	SL	R	K	J	S	M	S	SL	R	K	J	S	M	S	SL	R	K
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
5	405	P	MD	S	MD	L	S	P	MS	L	CT	M	M	LM	S	S	P	MS	L	S	S	S	M	L	P	P	CT	S	P	MS	M
6	406	M	M	LM	L	P	S	S	CT	MS	S	P		M	M	LM	L	P	MS	L	S	S	S	M	M	LM	L	S	MS	S	S
7	407	LM	S	L	P		S	S	S	P	P	MS	L	S	S	M	M	LM	L	L	MD	P	MD	S	S	MD		CT	S	P	MS

NO	ID staff	TANGGAL																												
		R	K	J	S	M	S	SL	R	K	J	S	M	S	SL	R	K	J	S	M	S	SL	R	K	J	S	M	S	SL	R	K
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
8	408	S	MD	S	S	L	MS	M	M	LM	S	P	P	P	MD	MS	S	S	P	L	P	MS	S	S	S	M	M	LM	L	MS	S

3) DIVISI BAYI NICU

ID Staff	Skill/Pekerjaan	ID Skill						Minggu I							Minggu II							Minggu III							Minggu IV			
			R	K	J	Sb	M	Sn	Sl	R	K	J	Sb	M	Sn	Sl	R	K	J	Sb	M	Sn	Sl	R	K	J	Sb	M	Sn	Sl	R	K
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
201	Kepala Unit	1	P	P	P	P	L	P	P	P	P	P	P	L	P	P	P	P	P	P	L	P	P	P	P	P	P	L	P	P	P	P

ID Staff	Skill/Pekerjaan	ID Skill						Minggu I							Minggu II							Minggu III							Minggu IV			
			R	K	J	Sb	M	Sn	Sl	R	K	J	Sb	M	Sn	Sl	R	K	J	Sb	M	Sn	Sl	R	K	J	Sb	M	Sn	Sl	R	K
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
202	Anggota	4	L	P	P	P	M	M	LM	L	P	S	P	P	P	L	P	P	P	L	P	P	P	L	P	P	M	M	L M	L	CT	P
203	Anggota	4	LM	L	L	S	P	P	M	M	LM	L	S	S	S	M	M	LM	L	P	P	CT	S	P	M	M	LM	L	S	S	P	S
204	Anggota	4	P	CT	M	M	LM	L	P	P	Mid	L	S	M	M	L M	L	P	P	S	S	P	L	M	M	LM	L	P	M	M	L M	L
205	Anggota	4	S	S	S	M	M	LM	L	S	S	P	M	M	LM	L	L	M	M	Lm	L	S	S	P	L	S	P	L	P	CT	M	M
207	Anggota	4	M	M	LM	L	S	S	S	P	M	M	LM	L	Mid	P	S	L	S	P	S	M	M	LM	L	LM	S	S	L	L	CT	CT
208	Anggota	4	LM	L	S	S	P	L	L	CT	M	M	LM	L	CT	S	P	S	Mid	M	M	LM	L	S	S	Mid	M	M	L M	L	S	S
209	Anggota	4	M	M	LM	L	L	P	M	M	LM	L	Mid	S	S	S	M	M	LM	L	CT	Mid	M	M	LM	L	S	P	L M	P	M	M
210	Anggota	4	S	Mid	M	M	LM	L	S	Mid	CT	M	M	LM	L	P	S	Mid	L	Mid	M	M	LM	L	S	P	Mid	S	M	M	L M	L
211	Anggota	4	Mid	P	P	L	Mid	S	Mid	L	P	S	L	P	M	M	LM	L	M	M	LM	L	P	Mid	Mid	M	M	LM	L	S	P	P
212	Anggota	4	L	S	Mid	P	M	M	LM	S	S	Mid	M	M	LM	L	Mid	S	S	S	M	M	LM	L	M	M	LM	L	S	M	M	LM

ID Staff	Skill/Pekerjaan	ID Skill						Minggu I							Minggu II							Minggu III							Minggu IV			
			R	K	J	Sb	M	Sn	Sl	R	K	J	Sb	M	Sn	Sl	R	K	J	Sb	M	Sn	Sl	R	K	J	Sb	M	Sn	Sl	R	K
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
213	Anggota	4	M	LM	L	Mid	S	M	M	LM	L	P	P	Mid	M	M	LM	L	M	M	LM	L	Mid	S	P	S	L	Mid	P	P	S	M
214	Anggota	4	P	M	M	LM	L	Mid	P	M	M	LM	L	L	P	L	M	M	LM	L	Mid	S	M	M	LM	L	P	M	M	LM	LM	Mid

4) DIVISI GIZI

Id Staff	PEKERJAAN	Id Skill	TANGGAL																													
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
501	Ahli gizi	1	P	P	P	P	L	P	P	P	P	P	P	L	P	P	P	P	P	P	MOD	P	P	P	P	P	L	L	P	P	P	P
502	chef	5	P	L	S	P1	P	P1	S	S	P	CT	L	L	P1	P1	S	S	S	P	P1	S	L	P	P2	S	S	P1	P2	P	P1	S
503	chef	5	L	MD	MD	MD	P1	MD	CT	MD	S	P1	P1	L	L	S	MD	S	P	MD	MD	P1	MD	MD	S	CT	L	S	MD	MD	MD	P

A-A-8

Id Staf	PEKERJAAN	Id Skil	TANGGAL																												
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
504	chef	5	MD	P2	P1	L	S	S	P	CT	CT	L	MD	S	S	MD	P1	MD	L	S	S	MD	S	S	CT	P	P1	MD	P1	P1	L	CT
505	chef	5	S	S	CT	S	MD	L	MD	P	S	S	S	P1	MD	L	CT	P	MD	L	CT	CT	P1	CT	MD	MD	MD	L	S	S	S	MD
506	Helper	6	P2	S	S	S	S	L	S	S	S	S	S	L	S	P	P	P2	P2	P2	P	S	S	S	S	P2	P	P	L	S	P	S
507	Helper	6	P1	P1	S	CT	L	S	P1	P1	P1	MD	S	MD	L	S	S	P1	P1	P1	S	S	L	P1	P1	P1	S	CT	S	L	CT	P1
508	Helper	6	P2	L	P2	CT	L	P2	P2	P2	L	P	P	S	P2	P2	P2	L	S	S	S	P	P	CT	CT	S	P2	P2	S	S	P2	P2
509	Helper	6	S	S	P	P	P2	L	S	S	S	P2	P2	P2	L	L	CT	S	S	S	P2	P2	P2	P2	S	L	S	S	CT	P2	S	S
510	Penyaji	7	L	CT	S	S	MD	M	M	M	LM	L	P	P	M	M	LM	L	CT	MD	MD	P	M	M	LM	L	S	S	MD	S	P	P
511	Penyaji	7	P	P	MD	MD	P	L	S	P	P	P	MD	MD	L	S	P	MD	S	S	P	L	S	P	P	P	CT	L	CT	CT	CT	CT
512	Penyaji	7	S	S	M	M	M	LM	L	MD	S	S	S	L	CT	CT	M	M	LM	L	S	S	P	P	CT	CT	M	M	LM	L	S	S
513	Penyaji	7	MD	MD	P	P	L	P	P	S	M	M	LM	L	S	S	P	P	M	M	LM	L	S	S	M	M	LM	L	S	P	MD	MD
514	Penyaji	7	M	M	LM	CT	S	MD	S	P	P	P	M	M	LM	CT	S	S	P	P	L	MD	MD	CT	CT	L	L	P	M	M	LM	L

Id Staf f	PEKERJAAN	Id Skil l	TANGGAL																												
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
515	Penyaji	7	P	P	MD	MD	P	S	L	M	M	LM	L	S	P	P	MD	L	MD	MD	M	M	LM	L	S	S	P	P	P	MD	M	M
516	Cafe	8	P	P	P	S	L	PS	P	S	S	S	P	L	PS	S	P	P	P	S	PS	L	CT	CT	CT	CT	PS	PS	L	S	P	P
517	Cafe	8	S	S	S	P	PS	L	S	P	P	P	S	PS	L	P	S	S	S	P	L	PS	S	P	P	P	CT	L	PS	P	S	S
518	Driver	9	S	S	S	P	PS	L	P	S	S	P	P	L	PS	S	S	P	P	L	S	PS	S	P	P	P	S	PS	L	P	S	S
519	Driver	9	CT	CT	CT	S	L	PS	S	CT	CT	S	S	PS	L	CT	CT	S	S	PS	P	L	P	S	S	S	P	L	PS	S	P	P

5) DIVISI OK

ID staff	Skill/Pekerjaan	ID Skill	Tanggal																												
			R	K	J	Sb	M	Sn	Sl	R	K	J	Sb	M	Sn	Sl	R	K	J	Sb	M	Sn	Sl	R	K	J	Sb	M	Sn	Sl	R	K
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
601	Staf	4	P	P	P	P	P	CT	CT	S	S	P	P	L	P	S	P	S	S	L	P	S	S	S	P	L	P	P	P	P	S	P
602	Staf	4	P	S	S	S	L	P	P	P	P	L	S	P	S	P	S	P	L	P	P	P	P	P	L	P	S	P	S	S	L	S
603	Staf	4	P	S	P	L	P	S	S	P	P	S	P	P	L	P	P	S	P	S	L	S	P	P	S	P	CT	L	S	P	P	P
606	Staf	4	S	P	P	P	L	S	S	P	P	P	CT	L	S	P	P	P	P	P	P	L	S	S	S	P	P	L	P	S	S	S
604	Staf	4	L	P	S	S	P	P	P	L	P	S	S	P	P	S	L	P	S	S	P	P	P	L	P	S	P	P	P	P	P	CT
605	Staf	4	P	L	P	P	P	P	L	S	S	P	P	P	P	CT	S	P	P	P	L	P	P	P	P	S	S	P	L	P	P	P

6) DIVISI SIM & RM

ID Staff	Skill/Pekerjaan	ID Skill	Tanggal																													
			Rb	K	J	Sb	Mg	Sn	Sl	Rb	K	J	Sb	Mg	Sn	Sl	Rb	K	J	Sb	Mg	Sn	Sl	Rb	K	J	Sb	Mg	Sn	Sl	Rb	K
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
301	Staf	4	P	CT	P	P	L	MS	S	S	P	P	P	L	S	P	S	M	M	LM	L	MS	S	M	M	M	LM	L	S	S	S	P
302	Staf	4	L	S	S	S	P	M	M	LM	L	S	S	MS	MS	M	M	LM	L	P	P	S	S	P	P	L	S	P	P	P	M	LM
303	Staf	4	S	P	P	MS	S	L	P	P	M	M	LM	L	CT	S	MS	S	S	S	L	M	M	LM	L	P	S	L	S	P	P	M
304	Staf	4	S	M	M	LM	L	S	S	M	LM	L	S	P	P	MS	L	S	P	P	S	P	P	CT	P	S	M	M	LM	L	P	MS
305	Staf	4	P	MS	S	M	M	LM	L	S	S	P	P	S	M	LM	L	P	P	S	MS	L	P	MS	S	P	P	S	M	M	LM	L
306	Staf	4	M	LM	L	L	MS	P	P	P	MS	S	M	M	LM	L	P	P	S	M	M	LM	L	S	S	S	P	MS	P	S	S	S

LAMPIRAN B

HASIL UJI COBA ALGORITMA

1) SELF ADAPTIVE

Number Of Iteration	FARMASI -SA		
	MAX	MIN	AVERAGE
1	0.920652	0.813278	0.919019
2	0.943146	0.813278	0.939413
3	0.935109	0.813278	0.93402
4	0.970917	0.813278	0.962513
5	0.924423	0.813278	0.91916
6	0.878097	0.813278	0.877403
7	0.939528	0.813278	0.938001
8	0.890599	0.813278	0.88978
9	0.906069	0.813278	0.904604
10	0.903125	0.813278	0.9025
11	0.890599	0.813278	0.890341
12	0.963333	0.813278	0.960785
13	0.93677	0.813278	0.93483
14	0.986339	0.813278	0.982586
15	0.964506	0.813278	0.959671
16	0.90034	0.813278	0.897708
17	0.938889	0.813278	0.926884
18	0.929045	0.813278	0.92681
19	0.911672	0.813278	0.909679
20	0.958683	0.813278	0.956074
21	0.922584	0.813278	0.921398

A-A-2

Number Of Iteration	IGD- SA		
	MAX	MIN	AVERAGE
1	0.920652	0.813278	0.919019
2	0.943146	0.813278	0.939413
3	0.935109	0.813278	0.93402
4	0.970917	0.813278	0.962513
5	0.924423	0.813278	0.91916
6	0.878097	0.813278	0.877403
7	0.939528	0.813278	0.938001
8	0.890599	0.813278	0.88978
9	0.906069	0.813278	0.904604
10	0.903125	0.813278	0.9025
11	0.890599	0.813278	0.890341
12	0.963333	0.813278	0.960785
13	0.93677	0.813278	0.93483
14	0.986339	0.813278	0.982586
15	0.964506	0.813278	0.959671
16	0.90034	0.813278	0.897708
17	0.938889	0.813278	0.926884
18	0.929045	0.813278	0.92681
19	0.911672	0.813278	0.909679
20	0.958683	0.813278	0.956074
21	0.922584	0.813278	0.921398

Number Of Iteration	BAYI NICU - SA		
	MAX	MIN	AVERAGE
1	0.933895	0.664773	0.928485
2	0.936336	0.664773	0.926761
3	0.950636	0.664773	0.904196

Number Of Iteration	BAYI NICU - SA		
	MAX	MIN	AVERAGE
4	0.901501	0.664773	0.892771
5	0.970241	0.664773	0.95027
6	0.921483	0.664773	0.917221
7	0.929562	0.664773	0.926254
8	0.925553	0.664773	0.903906
9	0.906024	0.664773	0.90085
10	0.939161	0.664773	0.93569
11	0.918974	0.664773	0.915805
12	0.941934	0.664773	0.935684
13	0.928191	0.664773	0.911252
14	0.935395	0.664773	0.9264
15	0.937685	0.664773	0.929457
16	0.914063	0.664773	0.902577
17	0.931314	0.664773	0.927342
18	0.968424	0.664773	0.964552
19	0.89655	0.664773	0.892664
20	0.942463	0.664773	0.931472
21	0.949702	0.664773	0.934164

Number Of Iteration	GIZI - SA		
	MAX	MIN	AVERAGE
1	0.933895	0.664773	0.928485
2	0.750703	0.70004	0.748305
3	0.771515	0.70004	0.770195
4	0.761221	0.70004	0.758015
5	0.853957	0.70004	0.823321

Number Of Iteration	GIZI - SA		
	MAX	MIN	AVERAGE
6	0.749566	0.70004	0.748481
7	0.744169	0.70004	0.7397
8	0.757412	0.70004	0.752888
9	0.750223	0.70004	0.750047
10	0.801298	0.70004	0.799635
11	0.768976	0.70004	0.76844
12	0.825189	0.70004	0.820097
13	0.820296	0.70004	0.816051
14	0.85056	0.70004	0.844373
15	0.830578	0.70004	0.818828
16	0.817871	0.70004	0.813833
17	0.847334	0.70004	0.841012
18	0.85121	0.70004	0.844393
19	0.836416	0.70004	0.826483
20	0.81533	0.70004	0.807385
21	0.801387	0.70004	0.798787

Number Of Iteration	OK-SA		
	MAX	MIN	AVERAGE
1	0.968559	0.968559	0.968559
2	0.968559	0.968559	0.968559
3	0.981669	0.968559	0.981582
4	0.968559	0.968559	0.968559
5	0.968559	0.968559	0.968559
6	0.968559	0.968559	0.968559
7	0.968559	0.968559	0.968559
8	0.968559	0.968559	0.968559

Number Of Iteration	OK-SA		
	MAX	MIN	AVERAGE
9	0.968559	0.968559	0.968559
10	0.968559	0.968559	0.968559
11	0.981669	0.968559	0.981625
12	0.968559	0.968559	0.968559
13	0.968559	0.968559	0.968559
14	0.968559	0.968559	0.968559
15	0.968559	0.968559	0.968559
16	0.968559	0.968559	0.968559
17	0.968559	0.968559	0.968559
18	0.981669	0.968559	0.981582
19	0.968559	0.968559	0.968559
20	0.968559	0.968559	0.968559
21	0.968559	0.968559	0.968559

Number Of Iteration	SIM RM -SA		
	MAX	MIN	AVERAGE
1	0.98321	0.979592	0.983198
2	0.983806	0.979592	0.983764
3	0.987805	0.979592	0.987046
4	0.992063	0.979592	0.991694
5	0.989691	0.979592	0.989657
6	0.982704	0.979592	0.982683
7	0.98321	0.979592	0.983198
8	0.990504	0.979592	0.989992
9	0.993984	0.979592	0.99384
10	0.990574	0.979592	0.986036

Number Of Iteration	SIM RM -SA		
	MAX	MIN	AVERAGE
11	0.989782	0.979592	0.989489
12	0.989899	0.979592	0.98978
13	0.98321	0.979592	0.983198
14	0.982993	0.979592	0.982978
15	0.992063	0.979592	0.991667
16	0.98321	0.979592	0.983198
17	0.982993	0.979592	0.982902
18	0.992658	0.979592	0.992585
19	0.996552	0.979592	0.995751
20	0.992969	0.979592	0.992729
21	0.99835	0.979592	0.998074

2) HILL CLIMBING

Number Of Iteration	FARMASI - HC		
	MAX	MIN	AVERAGE
1	0.912126	0.665158	0.882523
2	0.814749	0.665158	0.804996
3	0.898578	0.665158	0.890858
4	0.911426	0.665158	0.903236
5	0.874277	0.665158	0.852123
6	0.817925	0.665158	0.816561
7	0.924674	0.665158	0.905974
8	0.880013	0.665158	0.876207
9	0.836873	0.665158	0.832568
10	0.885324	0.665158	0.879126
11	0.88945	0.665158	0.869396
12	0.892298	0.665158	0.888704

13	0.918138	0.665158	0.905487
14	0.943129	0.665158	0.932347
15	0.943129	0.665158	0.932347
16	0.943129	0.665158	0.932347
17	0.943129	0.665158	0.932347
18	0.943129	0.665158	0.932347
19	0.943129	0.665158	0.932347
20	0.943129	0.665158	0.932347
21	0.943129	0.665158	0.932347

Number Of Iteration	IGD - HC		
	MAX	MIN	AVERAGE
1	0.956184	0.813278	0.951822
2	0.966088	0.813278	0.96329
3	0.915254	0.813278	0.91201
4	0.915254	0.813278	0.91201
5	0.920027	0.813278	0.918686
6	0.920027	0.813278	0.918686
7	0.920027	0.813278	0.918686
8	0.920027	0.813278	0.918686
9	0.920027	0.813278	0.918686
10	0.920027	0.813278	0.918686
11	0.920027	0.813278	0.918686
12	0.920027	0.813278	0.918686
13	0.920027	0.813278	0.918686
14	0.920027	0.813278	0.918686
15	0.920027	0.813278	0.918686
16	0.920027	0.813278	0.918686

Number Of Iteration	IGD - HC		
	MAX	MIN	AVERAGE
17	0.920027	0.813278	0.918686
18	0.920027	0.813278	0.918686
19	0.920027	0.813278	0.918686
20	0.920027	0.813278	0.918686
21	0.920027	0.813278	0.918686

Number Of Iteration	BAYI NICU -HC		
	MAX	MIN	AVERAGE
1	0.903883	0.664773	0.89447
2	0.947043	0.664773	0.941062
3	0.918219	0.664773	0.915157
4	0.918219	0.664773	0.915157
5	0.918219	0.664773	0.915157
6	0.918219	0.664773	0.915157
7	0.918219	0.664773	0.915157
8	0.918219	0.664773	0.915157
9	0.918219	0.664773	0.915157
10	0.918219	0.664773	0.915157
11	0.918219	0.664773	0.915157
12	0.918219	0.664773	0.915157
13	0.918219	0.664773	0.915157
14	0.918219	0.664773	0.915157
15	0.918219	0.664773	0.915157
16	0.918219	0.664773	0.915157
17	0.918219	0.664773	0.915157
18	0.918219	0.664773	0.915157
19	0.918219	0.664773	0.915157
20	0.918219	0.664773	0.915157

21	0.918219	0.664773	0.915157

Number Of Iteration	GIZI-HC		
	MAX	MIN	AVERAGE
1	0.832377	0.70004	0.830818
2	0.741014	0.70004	0.740458
3	0.751527	0.70004	0.745643
4	0.742427	0.70004	0.742051
5	0.801288	0.70004	0.791638
6	0.831654	0.70004	0.821651
7	0.752669	0.70004	0.75226
8	0.83811	0.70004	0.833725
9	0.800129	0.70004	0.797856
10	0.81203	0.70004	0.809987
11	0.85056	0.70004	0.838344
12	0.800363	0.70004	0.794447
13	0.812949	0.70004	0.805796
14	0.896015	0.70004	0.881339
15	0.761454	0.70004	0.759685
16	0.866917	0.70004	0.864917
17	0.815404	0.70004	0.808762
18	0.790789	0.70004	0.789106
19	0.879213	0.70004	0.85716
20	0.777428	0.70004	0.775088
21	0.852278	0.70004	0.83144

Number Of Iteration	OK-HC		
	MAX	MIN	AVERAGE
1	0.988359	0.968559	0.988293

Number Of Iteration	OK-HC		
	MAX	MIN	AVERAGE
2	0.968559	0.968559	0.968559
3	0.968559	0.968559	0.968559
4	0.971803	0.968559	0.971792
5	0.971803	0.968559	0.971792
6	0.988359	0.968559	0.988139
7	0.968559	0.968559	0.968559
8	0.981669	0.968559	0.981582
9	0.968559	0.968559	0.968559
10	0.968559	0.968559	0.968559
11	0.968559	0.968559	0.968559
12	0.968559	0.968559	0.968559
13	0.968559	0.968559	0.968559
14	0.968559	0.968559	0.968559
15	0.968559	0.968559	0.968559
16	0.968559	0.968559	0.968559
17	0.968559	0.968559	0.968559
18	0.968559	0.968559	0.968559
19	0.971803	0.968559	0.971738
20	0.968559	0.968559	0.968559
21	0.968559	0.968559	0.968559

Number Of Iteration	SIM RM - HC		
	MAX	MIN	AVERAGE
1	0.991991	0.979592	0.991648
2	0.98321	0.979592	0.983198
3	0.982704	0.979592	0.982673
4	0.98321	0.979592	0.983198
5	0.991738	0.979592	0.988912

Number Of Iteration	SIM RM - HC		
	MAX	MIN	AVERAGE
6	0.992969	0.979592	0.992762
7	0.98321	0.979592	0.983198
8	0.989945	0.979592	0.989299
9	0.982704	0.979592	0.982694
10	0.98321	0.979592	0.983198
11	0.979592	0.979592	0.979592
12	0.983806	0.979592	0.983758
13	0.989782	0.979592	0.989376
14	0.992969	0.979592	0.992111
15	0.989782	0.979592	0.989626
16	0.979592	0.979592	0.979592
17	0.98321	0.979592	0.983198
18	0.989899	0.979592	0.989597
19	0.98321	0.979592	0.983198
20	0.982704	0.979592	0.982694
21	0.983806	0.979592	0.983726