ITS
Institut
Teknologi
Sepuluh Nopember

TUGAS AKHIR – IS 184853

OPTIMASI PENJADWALAN PERAWAT DENGAN ALGORITMA *SELF ADAPTIVE LEARNING - GREAT DELUGE BASED HYPER-HEURISTIC* MENGGUNAKAN DATASET BENCHMARK RUMAH SAKIT DI NORWEGIA

OPTIMIZATION ON NURSE ROSTERING USING SELF ADAPTIVE LEARNING - GREAT DELUGE HYPER-HEURISTIC BASED ALGORITHM FOR NORWEGIAN HOSPITAL BENCHMARK DATASET

SHAFIRA WIDYA HAPSARI
NRP 05211640000041

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

DEPARTEMEN SISTEM INFORMASI
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember
Surabaya 2020

ITS
Institut
Teknologi
Sepuluh Nopember

TUGAS AKHIR – IS 184853

OPTIMASI PENJADWALAN PERAWAT DENGAN ALGORITMA *SELF ADAPTIVE LEARNING – GREAT DELUGE BASED HYPER-HEURISTIC* MENGGUNAKAN DATASET BENCHMARK RUMAH SAKIT DI NORWEGIA

SHAFIRA WIDYA HAPSARI
NRP 05211640000041

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

DEPARTEMEN SISTEM INFORMASI
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember
Surabaya 2020

Halaman ini sengaja dikosongkan

Institut
Teknologi
Sepuluh Nopember

TUGAS AKHIR – IS 184853

OPTIMIZATION ON NURSE ROSTERING USING SELF ADAPTIVE LEARNING - GREAT DELUGE HYPER-HEURISTIC BASED ALGORITHM FOR NORWEGIAN HOSPITAL BENCHMARK DATASET

SHAFIRA WIDYA HAPSARI
NRP 05211640000041

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

DEPARTMENT OF INFORMATION SYSTEMS
Faculty of Intelligent Electrical and Informatics Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2020

Halaman ini sengaja dikosongkan

LP/P/20/039

LEMBAR PENGESAHAN

Optimasi Penjadwalan Perawat dengan Algoritma Self Adaptive Learning - Great Deluge Based Hyper-heuristic Menggunakan Dataset Benchmark Rumah Sakit di Norwegia

TUGAS AKHIR

Disusun Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer (S.Kom)
pada
Departemen Sistem Informasi
Fakultas Teknologi Elektro dan Informatika Cerdas (ELECTICS)
Institut Teknologi Sepuluh Nopember

Oleh

Shafira Widya Hapsari
05211640000041

Surabaya, 14 Agustus 2020

Kepala Departemen Sistem Informasi

Dr. Mudjahidin, ST., MT.
NIP. 197010102003121001

Halaman ini sengaja dikosongkan

LEMBAR PERSETUJUAN

OPTIMASI PENJADWALAN PERAWAT DENGAN ALGORITMA SELF ADAPTIVE LEARNING-GREAT DELUGE BASED HYPER HEURISTIC MENGGUNAKAN DATASET BENCHMARK RUMAH SAKIT DI NORWEGIA

TUGAS AKHIR

Disusun untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Departemen Sistem Informasi
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember
Oleh :

SHAFIRA WIDYA HAPSARI
NRP. 05211640000041

Disetujui Tim Penguji :Tanggal Ujian : 15 Juli 2020
Periode Wisuda : September 2020

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)**

Halaman ini sengaja dikosongkan

OPTIMASI PENJADWALAN PERAWAT DENGAN ALGORITMA SELF ADAPTIVE LEARNING-GREAT DELUGE BASED HYPER HEURISTIC MENGGUNAKAN DATASET BENCHMARK RUMAH SAKIT DI NORWEGIA

Nama Mahasiswa : Shafira Widya Hapsari
NRP : 05211640000041
Jurusan : Sistem Informasi FTEIC-ITS
Pembimbing 1 : Ahmad Muklason, S.Kom., M.Sc., Ph.D.

ABSTRAK

Penjadwalan perawat merupakan sebuah permasalahan yang umum dijumpai oleh instansi kesehatan misalnya rumah sakit. Salah satu karakteristik utama dari penjadwalan perawat terletak pada penerapannya yang berkelanjutan dalam periode waktu. Meski merupakan sebuah permasalahan yang umum, solusi yang ditawarkan cenderung belum mendekati optimal sehingga persoalan ini dikategorikan sebagai permasalahan NP-Hard. Dalam menemukan solusi yang optimal, perusahaan dibenturkan dengan banyaknya batasan ataupun aturan yang harus ditaati. Sejauh ini, masih terdapat sejumlah rumah sakit yang menggunakan penjadwalan secara manual, padahal dengan banyaknya jumlah perawat yang terlibat akan menyebabkan kompleksnya penjadwalan sehingga memungkinkan jadwal yang dihasilkan tidak lagi efisien dan efektif. Pertimbangan mengenai kemampuan yang dimiliki tiap-tiap perawat juga menambah kompleksitas dalam penentuan jadwal kerja. Hal tersebut kemudian dapat diatasi dengan membuat penjadwalan otomatis. Dengan cara otomatis, pembuatan jadwal kerja membutuhkan waktu yang relatif lebih singkat namun mampu memberikan solusi

yang mendekati optimal. Penjadwalan otomatis yang akan diterapkan tentunya akan mempertimbangkan hal-hal penting baik yang termasuk dalam hard constraint bahkan memaksimalkan pemenuhan terhadap soft constraint. Pembuatan jadwal juga akan memperhatikan kemampuan, keadilan, hukum, regulasi dan kebijakan yang diterapkan pada rumah sakit. Berdasarkan permasalahan yang ada maka penyelesaian akan dilakukan oleh penulis dengan menerapkan metode Self Adaptive Learning Great Deluge (SAGD) yang berbasis pada Hyper-heuristic. Dengan menerapkan skenario nilai alpha (α) sebesar 0,9 pada penelitian didapatkan jadwal kerja perawat yang memiliki penurunan pelanggaran sebesar 80,10%% untuk OpTur5 dan 6,30% untuk OpTur7. Persentase ini lebih baik daripada kedua algoritma pembandingnya yaitu Hill Climbing dan Great Deluge Namun, untuk OpTur4 performa SAGD hanya mampu menurunkan nilai penalti sebesar 78,88% yang artinya tidak lebih baik dari pada algoritma Great Deluge yang mampu menurunkan penalti awal sebesar 80,50%

Kata kunci: Penjadwalan Perawat, Self Adaptive Learning - Great Deluge, Hyper-heursitic,

OPTIMIZATION ON NURSE ROSTERING USING SELF ADAPTIVE LEARING – GREAT DELUGE HYPER-HURISTIC BASED ALGORITHM FOR NORWEGIAN HOSPITAL BENCHMARK DATASET

Name	**: Shafira Widya Hapsari**
NRP	**: 05211640000041**
Department	**: Sistem Informasi FTEIC-ITS**
Supervisor	**: Ahmad Muklason, S.Kom., M.Sc., Ph.D.**

ABSTRACT

Nurse Rostering is one of the common problem that encountered by health institution such as hospital. One of the main characteristic of nurse rostering lies in its continous application over a period of time. However, even it is a common problem, the solution that given tend to not being an optimum solution. This problem is categorized as a NP-Hard. While trying to find the optimum solution, an institution will meet rules that must be obeyed. So far, there still a few hospitals that use manual rostering despite the fact that numbers of nurse will increase the complexity that caused the schedule is no longer effective and efficient. Also, each nurse competence needed to be considered. All of that above, can be overcome by making an automatic rostering. Relatively it requires a shorter time and the same time provide a roster that is near optimum. The automatic rostering will consider to fulfill the hard constraint and maximize its fulfillment of soft constraint. In making a roster it necessary to pay attention to the competence, fairness, law, regulation and policies that are applied. Based on the mentioned problem, the solution will be carried out by applying Self Adaptive Learning – Great Deluge based Hyper-heuristic. The application of algorithm is expected to give a better nurse roster

for hospital. Through this research, it shows that Self Adaptive Learning – Great Deluge, with alpha (α) equals to 0.9, give a better roster with 80,10% improvement for OpTur 5 and 6,30% for OpTur7. This shows that the algorithm is capabale of giving a better solution compared to Hill Climbing and Great Deluge. However, for OpTur 4, SAGD performance can only improve the solution up to 78,88% which means that Great Deluge algorithm is better with 80,50% improvement.

Keywords: Nurse Rostering Problem, Self Adaptive Learning - Great Deluge, Hyper-heursitic,

SURAT PERNYATAAN BEBAS PLAGIARISME

Saya yang bertandatangan di bawah ini :

Nama	: Shafira Widya Hapsari
NRP	: 05211640000041
Tempat/Tanggal lahir	: Ujung Pandang, 14 November 1998
Fakultas/Departemen	: FTEIC / Departemen Sistem Informasi
Nomor Telp/Hp/email	: 081244787114 / shafira16@mhs.is.its.ac.id

Dengan ini menyatakan dengan sesungguhnya bahwa penelitian/makalah/tugas akhir saya yang berjudul

OPTIMASI PENJADWALAN PERAWAT DENGAN ALGORITMA *SELF ADAPTIVE LEARNING – GREAT DELUGE* BASED HYPER-HEURISTIC MENGGUNAKAN DATASET BENCHMARK RUMAH SAKIT DI NORWEGIA

Bebas Dari Plagiarisme Dan Bukan Hasil Karya Orang Lain.

Apabila dikemudian hari ditemukan seluruh atau sebagian penelitian/makalah/tugas akhir tersebut terdapat indikasi plagiarisme, maka saya bersedia menerima sanksi sesuai peraturan dan ketentuan yang berlaku.

Demikian surat pernyataan ini saya buat dengan sesungguhnya dan untuk dipergunakan sebagaimana mestinya.

Surabaya, Juli 2020

SHAFIRA WIDYA HAPSARI

NRP. 05211640000041

Halaman ini sengaja dikosongkan

KATA PENGANTAR

Puji syukur terucap kepada Allah SWT yang atas Rahmat dan Karunia-Nya penulis dapat menyelesaikan laporan Tugas Akhir yang berjudul "Optimasi Penjadwalan Perawat dengan Algoritma *Self Adaptive Learning – Great Deluge Based Hyper-Heuristic* Menggunakan *Dataset Benchmark* Rumah Sakit di Norwegia". Dalam penyelesaian laporan ini, penulis telah banyak mendapat bantuan berbagai pihak. Oleh karena itu, penulis mengucapkan terima kasih kepada:

1. Bapak Firdaus Ruswandi dan Ibu Sri Widiyanti, kedua orang tua penulis yang tak henti-hentinya mendoakan, memberikan perhatian, dukungan, semangat dan atas restunya penulis mampu menyelesaikan pengerjaan Tugas Akhir ini
2. Kedua saudara penulis, Anindita Raisa P dan Adiba Khairunnisa M, yang turut memberikan perhatian kepada penulis dengan cara-cara yang menghibur.
3. Bapak Ahmad Muklason, S.Kom., M.Sc., Ph.D. selaku dosen pembimbing yang telah meluangkan waktu, memberikan arahan, bimbingan serta dukungan hingga akhirnya tugas akhir ini dapat terselesaikan.
4. Bapak Edwin Riksakomara, S.Kom., MT. dan Bapak Faizal Mahananto S.Kom, M.Eng., Ph.D. selaku dosen penguji sidang tugas akhir yang memberikan saran dan kritik yang membangun kepada penulis
5. Dosen Departemen Sistem Informasi yang telah membagikan dan mengajarkan ilmu yang tentunya akan bermanfaat baik sekarang maupun dimasa mendatang.
6. Rizal, Dimas dan Sisca selaku tim NRP serta Refi yang telah saling berdiskusi, membantu, memberikan saran, dan semangat satu sama lain sejak awal pengerjaan tugas akhir.
7. Citra, Rofiqoh, Zendika, Sisca, Anisah, Clara, Berry, Fio dan Daffa yang kerap menjadi tempat menampung keluh kesah dan berbagi tawa.

8. Kawan-kawan Artemis, yang sama-sama berjuang sesuai jalannya masing-masing, atas dukungan dan motivasi satu sama lain.
9. Pihak-pihak terkait lain yang tidak dapat penulis sebutkan satu persatu yang telah membantu hingga selesainya penyusunan Tugas Akhir ini.

Semoga kebaikan-kebaikan yang penulis terima selama ini akan kembali kepada pihak-pihak di atas sebagai hal baik. Penulis menyadari penyusunan tugas akhir ini masih terdapat banyak kekurangan dan jauh dari kata sempurna. Oleh karena itu, penulis menerima akan kritik dan saran untuk perbaikan ke depannya. Semoga tugas akhir ini dapat memberikan manfaat bagi pihak-pihak yang sekiranya memerlukan.

Surabaya, Juli 2020

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

Halaman ini sengaja dikosongkan

BAB I
PENDAHULUAN

Bab pendahuluan ini menjabarkan latar belakang topik permasalahan, rumusan masalah, batasan masalah, tujuan tugas akhir, manfaat tugas akhir, serta relevansi dari tugas akhir yang dikerjakan.

1. 1. Latar Belakang

Instansi kesehatan misalnya rumah sakit memiliki tujuan untuk selalu memberikan pelayanan yang terbaik bagi pasiennya. Dalam proses memenuhi tujuannya maka akan melibatkan banyak sumber daya diantaranya yaitu sumber daya manusia. Kehadiran sumber daya manusia yang kompeten dan memiliki motivasi yang kuat sangat vital bagi keberlangsungan pelayanan kesehatan itu sendiri[1]. Sumber daya manusia yang berada pada instansi kesehatan tentunya sangat beragam salah satunya adalah perawat. Jumlah perawat untuk setiap instansi berbeda-beda tergantung pada kapasitas dari instansi masing-masing begitu pula dengan tugas yang menjadi tanggung jawabnya.

Namun, secara umum tugas perawat akan selalu berhubungan dengan pasien hampir 24 jam per hari yang kemudian mengharapkan agar perawat mampu memberi pelayanan yang efektif dan juga efisien[2]. Sehubungan dengan hal tersebut, maka diperlukan upaya untuk menjaga kinerja dari perawat yang bertugas salah satunya dengan melakukan penjadwalan yang optimal dan layak untuk diimplementasikan pada rentang waktu tertentu. Selanjutnya berdasarkan dengan kesepakatan bersama, pengelolaan jadwal sedemikian rupa memiliki harapan untuk dilaksanakan sebaik mungkin oleh pihak perawat.

Melakukan penjadwalan terhadap banyak sumber daya manusia membutuhkan begitu banyak pertimbangan terhadap batasan-batasan yang ada. Permasalahan yang kerap dijumpai dalam pengaturan jadwal perawat adalah

kompleksnya batasan yang harus dipenuhi misalnya batas maksimal dan minimal jam kerja yang mampu ditempuh oleh perawat serta masih banyak batasan lainnya. Salah satu poin penting yang juga harus dipertimbangkan pada pembuatan jadwal adalah keadilan. Keadilan didefinisikan sebagai pemerataan beban kerja yang ditanggung oleh masing-masing perawat. Kurang adilnya jadwal antar perawat memungkinkan pelayanan yang diberikan menjadi tidak optimal.

Rumah sakit tentunya sudah memiliki perencanaan terhadap jadwal kerja bagi perawatnya. Namun, kondisi yang masih sering dijumpai adalah penjadwalan tersebut masih dilakukan secara manual atau sering disebut sebagai *self-scheduling[3]*[3]. Hal ini tentunya akan menyita waktu dengan melihat kompleksnya berbagai pertimbangan serta regulasi dan kebijakan yang harus ditaati. Sehingga dibutuhkan penjadwalan perawat otomatis yang adil dan tentunya sesuai dengan batasan dan regulasi dari tiap-tiap instansi.

Sebagai bentuk referensi maka penjadwalan perawat akan dilakukan berdasarkan *dataset benchmark* dari beberapa rumah sakit yang ada di Norwegia. *Dataset* yang ada telah memberikan gambaran besar mengenai pihak-pihak yang terlibat beserta *hard constraint* yang harus dipenuhi dan juga *soft constraint* yang terus berusaha dimaksimalkan pencapaiannya.

Penelitian ini akan melakukan penjadwalan perawat otomatis dengan menggunakan *Great Deluge* (GD), merupakan salah satu bentuk *local search* yang akan melakukan validasi terhadap solusi awal. Melalui metode ini solusi awal mungkin digantikan dengan solusi terbaik yang berhasil ditemukan[4]. Mekanisme *Great Deluge* memiliki kemiripan dengan *Simulated Annealing* namun dari segi mekanisme memiliki nilai yang lebih baik karena pemberian solusi tidak bergantung pada probabilitas. Pemberian solusi

dengan GD, juga berhasil menyelesaikan persoalan optimasi yang kompleks dengan lebih cepat [5]. Sedangkan, *Self Adaptive Learning* dipilih sebagai sebuah pendekatan yang sebelumnya telah berhasil menyelesaikan persoalan lintas domain. *Self Adapative Learning* akan melakukan mekanisme pemilihan LLH sehingga hasil yang didapat bisa lebih optimal.

Bersumber pada penelitian yang dilakukan sebelumnya, telah dilakukan penggabungan kedua metode yaitu *Self Adaptive Learning Great Deluge*.Metode tersebut dilakukan terhadap permasalahan yang ada pada kerangka kerja HyFlex. Pada kerangka kerja tersebut permasalahan yang karakteristik LLH yang berbeda. Penelitian tersebut berhasil memberikan solusi yang lebih baik pada domain masalah *SAT, Bin Packing, TSP dan VRP* [6].

Berdasarkan latar belakang tersebut maka tugas akhir yang dikerjakan mengangkat topik optimasi penjadwalan perawat pada rumah sakit dengan menggunakan pendekatan *Self Adaptive Learning Great Deluge* yang nantinya akan memperhatikan pemenuhan terhadap batasan-batasan yang ada.

1. 2. Rumusan Masalah

Berdasarkan latar belakang yang ada, tugas akhir mengangkat permasalahan sebagaimana berikut:

1. Bagaimana solusi yang diberikan dengan menerapkan algoritma *Self Adaptive Learning Great Deluge* pada permasalahan penjadwalan perawat untuk beberapa Rumah Sakit di Norwegia?
2. Bagaimana performa algoritma *Self Adaptive Great Deluge* dalam menyelesaikan permasalahan penjadwalan perawat untuk beberapa rumah sakit di Norwegia?

1. 3. Batasan Permasalahan

Batasan masalah dalam mengerjakan tugas akhir ini adalah:

1. Studi kasus yang dijadikan topik utama merupakan persoalan penjadwalan perawat pada beberapa rumah sakit di Norwegia. *Dataset* dapat diunduh melalui situs Sintef[7].
2. Penggunaan algoritma akan dilakukan dengan bahasa pemrograman java.

1. 4. Tujuan Penelitian

Melalui perumusan masalah yang ada maka,tugas akhir yang dikerjakan akan mencapai tujuan sebagaimana berikut:

1. Melakukan pembuatan jadwal kerja perawat pada rumah sakit dengan menggunakan algoritma *Self Adaptive Learning Great Deluge* sebagai solusi permasalahan
2. Melakukan analisis dan pengujian performa terhadap algoritma yang digunakan yaitu *Self Adaptive Great Deluge*

1. 5. Manfaat Penelitian

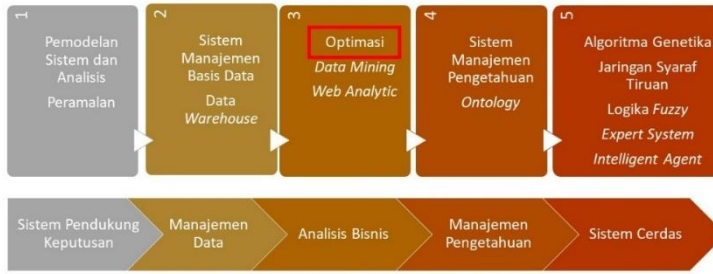
Manfaat yang diharapkan mampu diberikan dengan dikerjakannya penelitian pada tugas akhir adalah:

1. Mampu menghasilkan algoritma yang dapat diterapkan pada permasalahan penjadwalan perawat rumah sakit yang menghasilkan jadwal yang optimal dan dapat memenuhi batasan-batasan yang ada
2. Menambah pengetahuan serta wawasan dalam bidang keilmuan khususnya terkait dengan *Self Adaptive Great Deluge* yang berbasis pada *Hyper-heuristic* pada permasalahan penjadwalan perawat
3. Mampu menemukan algoritma yang mampu menyelesaikan permasalahan penjadwalan perawat di Indonesia dengan menyesuaikan batasan-batasan yang harus dipenuhi

1. 6. Relevansi

Penjadwalan perawat perlu dilakukan dengan mengelola data mengenai perawat yang bekerja pada rumah sakit. Pada pelaksanaannya tentu terdapat hal-hal yang perlu dilakukan untuk menghasilkan solusi yang adil dan optimal.

Pada Gambar 1.6. ditampilkan *roadmap* penelitian yang menjadi ruang lingkup laboratorium RDIB. Penelitian tugas akhir ini secara keseluruhan meliputi perihal optimasi dalam penjadwalan. Sehingga keterkaitan yang paling menonjol antara tugas akhir adalah dengan mata kuliah Optimasi Kombinatorial dan Heuristik (OKH). Secara garis besar mata kuliah ini berusaha untuk menemukan solusi yang optimal ataupun mendekati optimal.



Gambar 1. 1. Roadmap Laboratorium RDIB

1. 7. Ringkasan Pendahuluan

Permasalahan yang diangkat pada penelitian ini merupakan penjadwalan perawat yang ada di rumah sakit. Data yang digunakan merupakan data *benchmark* rumah sakit yang ada di Norwegia. Penelitian ini akan fokus kepada bagaimana algoritma yang diusulkan mampu menghasilkan jadwal yang sesuai dengan batasan yang dimiliki. Algoritma yang digunakan nantinya akan dibuat dengan bahasa pemrograman Java.

Tugas akhir ini berkaitan dengan optimasi dalam penjadwalan. Oleh karena itu, relevansi tugas akhir ini dengan laboratorium Rekayasa Data dan Intelegensi Bisnis

secara khusus terletak pada bidang keilmuan optimasi yang dibahas pada mata kuliah Optimasi Kombinatorial dan Heuristik (OKH). Pada bab selanjutnya akan dibahas mengenai penelitian terdahulu dan juga teori-teori yang akan mendukung penelitian.

BAB II
TINJAUAN PUSTAKA

Bab 2 ini berisikan tinjauan pustaka yang sesuai dengan permasalahan dibahas pada bab pendahuluan. Bab tinjauan pustaka akan terdiri dari penelitian sebelumnya dan dasar teori yang menunjang penelitian tugas akhir ini

Dalam proses pengerjaan tugas akhir terdapat beberapa penelitian yang mampu dijadikan bahan studi literatur untuk menyelesaikan tugas akhir ini sebagaimana pada tabel 2.1

Tabel 2. 1. Penjelasan studi literatur

Studi Literatur 1	
Nama Penelitian	Martin St_levik, Tomas Eric Nordlander, Atle Riise, and Helle Fr_yseth
Tahun Penelitian	2011
Judul Penelitian	*A hybrid approach for solving real-world nurse rostering problems*[8]
Deskripsi Umum	Literatur ini berisikan penyelesaian penjadwalan perawat dengan menggunakan pendekatan *hybrid*. Dengan tujuan efisiensi meminimalkan biayanya namun, di saat yang bersamaan perlu dilakukan pembagian *workload* yang adil. Selain dua hal tersebut, dari segi pasien perlu dipertimbangkan waktu tunggu yang singkat dan perawatan dengan personil yang memiliki kompetensi yang sesuai.
Keterkaitan Penelitian	Literatur ini akan dijadikan pedoman dalam mengerjakan penelitian. Sehingga permasalahan beserta tujuan yang ingin dicapai melalui penelitian memiliki lingkup yang serupa.

Studi Literatur 2	
Nama Penelitian	Nabeel R. AL-Milli
Tahun Penelitian	2010
Judul Penelitian	*Hybrid Genetic Algorithm with Great Deluge for Course Timetabling*[9]
Deskripsi Umum	Literatur ini mengembangkan algoritma *Great Deluge* dengan menggabungkannya dengan algoritma genetika (Genetic Algorithm). Algoritma tersebut akan digunakan untuk menyelesaikan permasalahan penjadwalan mata kuliah. Pada penelitian ini, dilakukan alokasi mata kuliah ke ruangan dan periode waktu dengan memenuhi sejumlah *constraints.* Sehingga tujuan dari peneilitian ini adalah memenuhi hard constraint serta meminimalkan pelanggaran terhadap soft constraint. Penelitian dilakukan dengan menggunakan bahasa pemrograman C++.
Keterkaitan Penelitian	Keterkaitan yang ada antara literatur dan penelitian yang dilakukan terletak pada pemrosesan solusi dengan melakukan modifikasi berupa penggabungan pada algoritma *Great Deluge*. Jika pada literatur ini digunakan untuk menyelesaikan penjadwalan pada mata kuliah, maka pada penelitian penjadwalan dilakukan terhadap perawat rumah sakit.

Studi Literatur 3	
Nama Penelitian	Fachrur Zaffrinda Prayogo
Tahun Penelitian	2018
Judul Penelitian	Optimasi Penjadwalan Staf Rumah Sakit Dengan Menggunakan Algoritma *Self-Adaptive Learning Hyper-heuristic* (Studi Kasus: RSIA Kendangsari Surabaya)[10]
Deskripsi Umum	Penelitian ini mengambil studi kasus pada Rumah Sakit Ibu dan Anak Kendangsari Surabaya yang berada di MERR. Metode yang digunakan pada penelitian ini adalah Algoritma *Self Adaptive Learning Hyper-heuristic.* Hasil penjadwalan akan dibandingkan dengan jadwal yang sudah berlaku di rumah sakit. Penjadwalan akan dilakukan kepada 6 divisi berbeda yang ada pada rumah sakit. Sebagai tolok ukur efektivitas jadwal, dilakukan perhitungan terhadap nilai *Jain's Fairness Index* yang memiliki rentang nilai 0-1. Pada penelitian ini pula, akan dibandingkan dengan algoritma Hill Climbing.
Keterkaitan Penelitian	Relevansi antara literatur dengan penelitian tugas akhir yang dilakukan adalah perihal metode yang digunakan dalam menyelesaikan permasalahan yang diangkat menjadi tugas akhir yakni algoritma *Self Adaptive Learning* yang berbasis pada *Hyper-heuristic.* Selain itu, permasalahan yang diangkat juga serupa yaitu penjadwalan pada perawat di rumah sakit.

Studi Literatur 4	
Nama Penelitian	Vicha Azthanty Supoyo, Ahmad Muklason
Tahun Penelitian	2019
Judul Penelitian	Pendekatan *Hyper Heuristic* dengan Kombinasi Algoritma pada *Examination Timetabling Problem*[11]
Deskripsi Umum	Tujuan penelitian ini dilakukan untuk membandingkan performa algoritma dalam menyelesaikan permasalahan penjadwalan ujian dengan batas waktu yang bervariasi. Penelitian dilakukan terhadap dua algoritma terhadap satu data set, dimana *dataset* yang digunakan adalah *dataset* Toronto. Hasil yang diperoleh pada penelitian ini menyatakan bahwa algoritma S*imulated Annealing* memberikan solusi yang lebih baik daripada *Hill Climbing*.
Keterkaitan Penelitian	Relevansi antara literatur dengan penelitian terletak pada inti dari permasalahan yang dihadapi, dimana permasalahan yang dimaksud adalah penjadwalan. Permasalahan tersebut kemudian diselesaikan dengan menggunakan algoritma berbasis *Hyper-heuristic*.

2. 1. Dasar Teori

Pada sub bab ini menjelaskan teori-teori yang akan menunjang pengerjaan penelitian tugas akhir.

2. 1. 1. Perawat

Perawat didefinisikan sebagai tenaga kesehatan *professional* yang memiliki kompetensi, tanggung jawab serta kewenangan dalam melakukan pelayanan keperawatannya[12]. Rumah sakit mengharapkan sumber

daya yang dimilikinya untuk melakukan perannya sesuai dengan kemampuan masing-masing.

Dalam melaksanakan tugasnya sebagai pemberi layanan perawat dapat dibedakan menjadi tiga fungsi yakni fungsi independen, interdependen dan dependen[13]. Perawat secara teori memiliki fungsi sebagaimana berikut:

- Fungsi independen
  Merupakan fungsi yang memberikan wewenang bagi perawat untuk bertugas tanpa perintah dokter atau bersifat mandiri yang tentunya didasarkan oleh ilmu keperawatan yang telah diterapkan.
- Fungsi interdependen
  Interdependen menandakan bahwa perawat bertindak dengan dasar kerjasama dengan tim kesehatan yang bertugas
- Fungsi Dependen
  Sedangkan fungsi dependen maka, perawat memiliki fungsi untuk membantu dokter dalam memberikan tindakan medis.

2. 1. 2. Penjadwalan

Penjadwalan menurut jenisnya dapat dibagi menjadi tiga yaitu[14] :

1. Penjadwalan Operasi, yang mana merupakan pengalokasian pekerjaan pada *workstation* tertentu
2. Penjadwalan Permintaan, dimana pelanggan dijadwalkan di waktu yang pasti untuk dipenuhi permintaannya
3. Penjadwalan Personil, yakni penyusunan jadwal yang dibutuhkan untuk menentukan pelaksanaan kerja tenaga kerja

Penjadwalan merupakan proses pengambilan keputusan yang dibutuhkan dalam berbagi industri. Dalam pengalokasiannya membutuhkan pertimbangan pemanfaaatan sumber daya untuk periode waktu tertentu.

Tujuan penjadwalan adalah untuk memberikan alternatif yang optimal agar suatu tujuan tercapai. Dalam menentukan penjadwalan maka, terdapat enam (6) kriteria yang perlu dipertimbangkan yaitu: *Coverage*, *Quality, Stability, Flexibility, Fairness*, dan *Cost*[15].

- *Coverage* mengacu pada seberapa baik penjadwalan mencakup kebutuhan dari pasien yang membutuhkan perawat dan apakah penjadwalan akan memberikan pelayanan yang adil bagi pasien.
- *Quality* yang berarti penjadwalan dinilai dari kepuasan perawat akan jadwal yang ditentukan. Kepuasan tersebut bisa dipengaruhi dengan adanya preferensi dari tiap-tiap perawat.
- *Stability* berarti penilaian penjadwalan dengan melihat apakah jadwal bisa berubah atau tetap selama periode waktu tertentu.
- *Flexibility* memberikan penilaian dari segi kemampuan jadwal untuk beradaptasi apabila terdapat perubahan kondisi dan situasi yang terjadi saat itu.
- *Fairness* yang mengacu pada pembagian merata yang dibebankan pada tiap-tiap perawat.
- Terakhir, *Cost* yang tentunya akan menilai penjadwalan dari segi biaya. Penjadwalan akan dievaluasi berdasarkan efektivitas dan efisiensinya dalam mengalokasikan sumber daya dalam konteks ini maka perawat.

2. 1. 3. Penjadwalan Perawat

Setiap instansi yang berdiri tentunya akan melibatkan berbagai personel dalam kegiatannya. Personel tersebut pun memiliki peran dan tanggung jawab yang berbeda. Permasalahan umum yang dihadapi instansi-instansi yang memiliki pegawai yakni sering mengalami permasalahan dalam melakukan penjadwalan bagi

pegawainya. Hal tersebut juga terjadi pada instansi kesehatan layaknya rumah sakit.

Permasalahan Penjadwalan Perawat (*Nurse Rostering Problem)* merupakan persoalan pengoptimalan yang akan menugaskan sejumlah perawat kepada sejumlah *shift* dengan tujuan memenuhi kebutuhan rumah sakit[16]. Pada permasalahan ini akan melakukan proses pengalokasian jadwal kerja kepada perawat dengan mempertimbangkan kemampuan, kompetensi, keadilan, hukum dan regulasi yang berlaku[8]. NRP dikategorikan sebagai permasalahan NP-Hard yang berarti hingga saat ini belum ada metode eksak yang berhasil memberikan solusi yang optimal.

Batasan yang akan terlibat dalam penjadwalan perawat dapat dibagi menjadi dua yakni *hard constraint* dan *soft constraint*. *Hard constraint* merupakan batasan yang harus dipenuhi dalam pemenuhan solusi. Selanjutnya terdapat *soft constraint* yang diartikan sebagai batasan yang seharusnya atau sebaiknya dipenuhi. *Soft constraint* tidak bersifat wajib untuk dipenuhi. Namun, solusi akan dinilai semakin baik apabila semakin banyak *soft constraint* yang dapat dipenuhi.

2. 1. 3. 1. Pemodelan Matematika

Pemodelan akan dilakukan terhadap *hard constraint* dan *soft constraint* yang terlibat pada penjadwalan perawat. Selain *hard* dan *soft constraint* terdapat pula fungsi objektif yang akan menghitung bagaimana performa dari solusi yang dihasilkan. Berikut penjelasan variabel, parameter, variabel keputusan, *derived information*, *hard constraint*, *soft constraint* serta fungsi objektif untuk penjadwalan perawat pada dataset rumah sakit yang ada di Norwegia[17].

1. Variabel

Berikut pada tabel 2.2 di bawah merupakan variabel yang akan terlibat pada model matematika yang diterapkan

Tabel 2. 2. Penjelasan variabel model matematika

Variabel	Penjelasan
$\boldsymbol{S}$	Himpunan dari *shift* dimana *shift s* adalah anggota dari hanya satu kategori *c*
$\boldsymbol{s}$	Anggota dari himpunan *shift S*
$\boldsymbol{C}$	Himpunan kategori dari *shift*
$\boldsymbol{c}$	Anggota dari himpunan kategori shift *C*
$\boldsymbol{D}$	Himpunan dari hari
$\boldsymbol{d}$	Anggota dari himpunan *D*, dimana hari ke-1 adalah hari Senin
$\boldsymbol{E}$	Himpunan dari perawat
$\boldsymbol{e}$	Anggota dari himpunan karyawan *E*
$\boldsymbol{I_d}$	Himpunan dari kumpulan *shift* hari *d* dan hari *d+1* yang tidak kompatibel karena jarak antar *shift* terlalu dekat
$\boldsymbol{U}$	Himpunan dari pola *shift* yang tidak dikehendaki
$\boldsymbol{u}$	Anggota dari himpunan pola *U*
$\boldsymbol{V}$	Himpunan dari pola *shift* yang dikehendaki
$\boldsymbol{v}$	Anggota dari himpunan pola *V*
$\boldsymbol{W}$	Himpunan dari minggu
$\boldsymbol{w}$	Anggota dari himpunan *W*
$\boldsymbol{x}$	Solusi berupa jadwal kerja perawat

2. Parameter

Parameter yang ada pada model matematika dijelaskan pada tabel 2.3

Tabel 2. 3. Penjelasan parameter model matematika

Parameter	Penjelasan
p_m	Penalti dari pelanggaran *soft constraint m*, dimana *m* bernilai 1 hingga 9
N_c^{min}	Jumlah minimum *shift* dengan kategori *c* yang berurutan untuk setiap karyawan
N_c^{max}	Jumlah maksimum *shift* dengan kategori *c* yang berurutan untuk setiap karyawan
N^{min}	Jumlah minimum *shift* dengan kategori apapun untuk setiap karyawan
N^{max}	Jumlah minimum *shift* dengan kategori apapun untuk setiap karyawan
N_{ec}^{min}	Jumlah minimum *shift* kategori *c* yang harus dikerjakan perawat *e*
N_{ec}^{max}	Jumlah maksimum *shift* kategori *c* yang harus dikerjakan perawat *e*
P_{ds}	Jumlah *shift* tipe *s* yang dibutuhkan untuk hari *d*
T_e	Waktu kerja maksimum perawat *e* setiap minggu
T_e^h	Waktu kerja perawat *e* yang harus dikerjakan sesuai dengan kontrak kerja
T^f	Waktu minimal untuk *free time* setiap minggu
T_s	Durasi dari *shift s*
A_{sc}	Bernilai 1 apabila *shift s* ada pada kategori *c;* Bernilai 0 jika sebaliknya
B_{es}	Bernilai 1 apabila perawat *e* memiliki kompetensi untuk bekerja pada *shift s;* Bernilai 0 jika sebaliknya

3. Variabel Keputusan

Dalam menerapkan model matematika, akan melibatkan sebuah variabel keputusan yaitu q_{eds}.

q_{eds} = Bernilai 1 apabila perawat *e* ditugaskan pada hari *d* kepada *shift s;* dan bernilai 0 jika sebaliknya

4. *Derived Information*

Terdapat 3 *derived information* yang turut berperan dalam model matematika, yaitu :

$f_{ew}(x)$ = Panjang dari *free periode* terpanjang untuk perawat *e* pada week *e* dalam solusi *x*

$u_{ei}(x) =$ Jumlah pola *shift i* yang tidak dikehendaki untuk perawat *e* pada solusi *x*

$v_{ei}(x) =$ Jumlah pola *shift i* yang dikehendaki untuk perawat *e* pada solusi *x*

5. *Fungsi Objektif*

Fungsi tujuan yang ingin dicapai ialah meminimalkan *penalty* dari setiap pelanggaran *soft constraint* sebagaimana pada persamaan (1).

$$Minimize: f = \sum_{m=1}^{9} K_m p_m \qquad (1)$$

Dimana K_m merupakan konstanta tetap yang mencerminkan bobot *penalty,* sedangkan P_m didefinisikan sebagai nilai *penalty* yang diperoleh dari pelanggaran terhadap *soft constraint.*

6. *Hard Constraint*

Selanjutnya, merupakan pemodelan matematika dari *hard constraint* yang berlaku. Persamaan (2) menyatakan tiap 1 perawat untuk 1 hari dialokasikan pada maksimal 1 *shift.*

$$\sum_{s \in S} q_{eds} \leq 1 \, , \forall \, e \in E, d \in D \qquad (2)$$

Persamaan (3) berfungsi untuk memastikan kebutuhan *shift* (*manpower plan*) harus terpenuhi setiap harinya.

$$\sum_{e \in E} q_{eds} = P_{ds} \, , \forall \, d \in D, s \in S \qquad (3)$$

Pemodelan (4) menyatakan bahwa total jumlah jam kerja untuk setiap perawat tidak terlalu menyimpang dari jam kerja yang telah disetujui pada kontrak. Besar penyimpangan berada dalam rentang ± 2%.

$$(1 - 0.02)T_e^h \leq \sum_{d \in D, \; s \in S} T_s q_{eds} \leq (1 + 0.02)T_e^h \, , \forall \, e \in E \qquad (4)$$

Persamaan (5) memiliki tujuan untuk setiap perawat hanya bekerja pada *shift* yang sesuai dengan kompetensinya.

$$\sum_{s \in S} B_{es} q_{eds} = 1 \, , \forall \, e \in E, d \in D \qquad (5)$$

Pada persamaan (6) menyatakan tidak boleh ada *shift* yang berurutan untuk setiap perawat. Dilakukan penentuan pasangan *shift* yang tidak boleh dialokasikan secara berurutan. Penentuan pasangan *shift* dilihat dari selisih waktu libur antar *shift*. Apabila pasangan shift memiliki waktu yang kurang dari yang telah ditentukan maka tidak boleh dialokasikan pada perawat. Salah satu contoh *shift* berurutan yang dilarang adalah ketika pada

hari ke-*d* mendapatkan *shift* malam dan kemudian pada saat *d+1* perawat mendapatkan *shift* pagi.

$$q_{eds} + q_{e(d+1)s'} \leq 1 , \forall\, e \in E, d \in D, (s, s') \in I_d \quad (6)$$

Persamaan (7) memiliki maksud untuk setiap minggu harus ada minimum periode bebas kerja.

$$f_{ew}(x) > T^f, \forall\, e \in E, w \in W \quad (7)$$

Persamaan (8) menentukan bahwa batas waktu kerja maksimal setiap minggu tidak boleh dilanggar.

$$\sum_{d=7w}^{7w+6} \sum_{s \in S} T_s q_{eds} \leq T_e , \forall\, e \in E, w \in W \quad (8)$$

7. *Soft constraint*

Berikut *soft constraint* pada penjadwalan perawat untuk *dataset benchmark* dari rumah sakit di Norwegia. Terdapat 9 *soft constraint* yang perlu dipertimbangkan dalam pengalokasian jadwal.

Pertama, ada persamaan (9) memiliki maksud tidak boleh terlalu banyak hari kerja berurutan dengan kategori *shift* yang sama.

$$p_1(\mathrm{x}) = \sum_{c \in C} \sum_{e \in E} \sum_{d=1}^{|D|-N_c^{max}} \left(\prod_{d'=d}^{d+N_c^{max}} y_{ecd'} \right) \quad (9)$$

Batasan untuk tidak boleh terlalu banyak hari kerja berurutan diatur pada persamaan (10).

$$p_2(\mathrm{x}) = \sum_{e \in E} \sum_{d=1}^{|D|-N^{max}} \left(\prod_{d'=d}^{d+N^{max}} z_{ed'} \right) \qquad (10)$$

Persamaan (11) berfungsi untuk membatasi sehingga tidak boleh terlalu sedikit hari kerja yang berurutan dengan kategori *shift* yang sama.

$$p_3(\mathrm{x}) = \sum_{c \in C} \sum_{e \in E} \sum_{d=1}^{|D|-N_c^{min}} \max\left(0, y_{ec(d+1)} - y_{ecd}\right) \left(N_c^{min} - \sum_{d'=d+1}^{d+N_c^{min}} \left(\prod_{d''=d+1}^{d+N_c^{min}} y_{ecd''} \right) \right) \qquad (11)$$

Persamaan (12) yakni untuk memberikan batasan tidak boleh terlalu sedikit hari kerja yang berurutan.

$$p_4(x) = \sum_{e \in E} \sum_{d=1}^{|D|-N^{min}} \max\left(0, z_{e(d+1)} - z_{ed}\right) \left(N^{min} - \sum_{d'=d+1}^{d+N^{min}} \left(\prod_{d''=d+1}^{d+N^{min}} z_{ed''} \right) \right) \qquad (12)$$

Batasan bahwa tidak boleh terlalu banyak menyimpang dari jumlah minimum dan maksimum dari tiap kategori diatur pada persamaan (13).

$$p_5(\mathrm{x}) = \sqrt{\sum_{c \in C} \sum_{e \in E} \left(\max\left(0, N_{ec}^{min} - \sum_{d \in D} y_{ecd}, \sum_{d \in D} y_{ecd} - N_{ec}^{max}\right) \right)^2} \qquad (13)$$

Selanjutnya persamaan (14) dimaksudkan untuk membatasi sehingga tidak boleh terlalu menyimpang dari kontrak jam kerja perawat.

$$p_6(\mathrm{x}) = \sqrt{\sum_{e \in E} \left(T_e^w - \sum_{d \in D} \sum_{s \in S} T_s q_{eds} \right)^2} \qquad (14)$$

Persamaan (15) ini, ditujukan untuk mengelompokkan waktu libur dari perawat.

$$p_7(\mathrm{x}) = \sqrt{\sum_{e \in E} \left(\sum_{d=1}^{|D|-1} \max(0, z_{ed} - z_{e(d+1)}) \right)^2} \qquad (15)$$

Persamaan (16) memiliki tujuan untuk membatasi sehingga tidak boleh terlalu sedikit pola *shift* yang dikehendaki.

$$p_8(x) = |E||D| - \sum_{e \in E, d \in D} v_{ei}(x) \qquad (16)$$

Terakhir, persamaan (17) memiliki tujuan untuk membatasi sehingga tidak boleh terlalu banyak pola *shift* yang tidak dikehendaki.

$$p_9(x) = \sum_{e \in E, i \in U} u_{ei}(x) \qquad (17)$$

Dimana,

$y_{ecd} = \sum_{s \in S} A_{sc} q_{eds}$ Bernilai 1 jika perawat *e* bekerja pada *shift* kategori *c* pada hari *d*; dan bernilai 0 jika sebaliknya.

$z_{ecd} = \sum_{s \in S} q_{eds}$ Bernilai 1 apabila perawat *e* bekerja pada hari *d*; dan bernilai 0 apabila sebaliknya.

2. 1. 4. Nurse Rostering Dataset

Dataset Nurse Rostering diperoleh melalui pengunduhan pada situs Sintef. Sintef sendiri merupakan sebuah organisasi riset terbesar di Eropa. *Dataset* berisikan kasus terkait permasalahan penjadwalan perawat di rumah sakit di Norwegia yang diperoleh melalui *software* vendor

Gatsoft AS. Gatsoft AS adalah perusahaan yang mengembangkan perangkat lunak terkait pengelolaan personil yang telah memberikan layanan kepada 80% Rumah Sakit di Norwegia[8].
Dataset terdiri dari 7 unit rumah sakit yang pada *dataset* disebut sebagai OpTur. Pada setiap OpTur tertera degan jelas informasi terkait perawat, jadwal kerja, rencana kerja, batasan dan pola. Jumlah perawat yang terlibat untuk setiap OpTur akan berbeda-beda sebagaimana pada Tabel 2.4.

Tabel 2. 4. Jumlah perawat untuk setiap OpTur

OpTur	Jumlah Perawat
1	51
2	82
3	29
4	30
5	20
6	54
7	15

Setiap perawat akan memiliki ID unik yang akan membedakan perawat satu dengan yang lainnya. Pada *dataset* juga disediakan informasi mengenai jam kerja perawat selama satu minggu hingga kompetensi yang dikuasai oleh perawat. Kemudian informasi selanjutnya yaitu mengenai *shift* selama 7 hari dalam seminggu yang memiliki sejumlah kategori.

2. 1. 5. Self Adaptive Learning

Self Adaptive Learning merupakan algoritma yang didasarkan pada *Self Adaptive Strategy*. Sehingga dapat didefinisikan sebagai algoritma yang secara adaptif bertujuan untuk memberikan arahan strategis dalam memberikan solusi yang optimal. Awal mulanya algoritma ini ditujukan untuk memilih lingkungan pada

bee colony algorithm yang memiliki tujuan menyelesaikan persoalan penjadwalan *flowshop*[18]. Metode akan membantu penentuan penggunaan *Low Level Heuristic* (LLH) yang sebaiknya digunakan untuk menyelesaikan masalah. Dengan *Self Adaptive Strategy* maka lebih sering memilih LLH yang memberikan nilai solusi terbaik. Berdasarkan referensi, terdapat variabel seperti pada tabel 2.5 yang digunakan pada algoritma.

Tabel 2. 5. Daftar variabel yang digunakan algoritma *Self Adaptive Learning*

Variabel	Penjelasan
P	Domain permasalahan
St	Kondisi pemberhentian
M	Jumlah LLH yang digunakan pada ***P***
LH	ArrayBlockingQueue dari Integer untuk menyimpan indeks LLH secara sementara
S	Ukuran dari ***LH***
WH	LinkedList dari Integer (tipe data) untuk menyimpan LLH yang memiliki performa yang baik
S_{best}	Fungsi objektif dengan solusi terbaik sejauh ini
I	LLH terpilih
newFunctionVal	Nilai fungsi objektif dari solusi yang dihasilkan dengan menggunakan LLH ***I***
S_{cur}	Fungsi objektif yang digunakan saat melakukan iterasi
B	*Boundary level*
α	*Decay rate*
N	Jumlah iterasi
R	ArrayBlockingQueue dari Double (tipe data) untuk menyimpan nilai solusi fungsi objektif yang diterima
L	Ukuran dari ***R***
C	Elemen indeks yang terpilih pada ***R***

2. 1. 6. Great Deluge

Great Deluge (GD) merupakan algoritma yang secara luas digunakan untuk menyelesaikan persoalan optimasi[19]. Secara garis besar *Great Deluge* memiliki kemiripan dengan *Simulated Annealing* dan *Hill Climbing* yang mana merupakan algoritma *local search*. Sebagaimana *local search* pada umumnya, maka akan dicari solusi terbaik dengan lingkup kecil terlebih dahulu untuk kemudian akan selalu berubah dengan berpindahnya lingkungan. Hingga akhirnya ditemukan solusi yang akan memenuhi kriteria.

Secara analogi algoritma ini akan diibaratkan sebagai seorang yang ingin menghindari banjir sehingga akan terus mencari titik tertinggi dalam suatu lingkungan. Ketika pada selanjutnya titik tersebut juga akan terendam banjir maka solusi akan berpindah ke titik yang lebih tinggi lainnya. Berdasarkan analogi maka, apabila nilai hasil solusi sama dengan ataupun berada di batas atas dari *water level* yang telah disepakati maka solusi akan diterima[20].

2. 1. 7. Hyper-heuristic

Hyper-heuristic merupakan sebuah metode yang memanfaatkan *heuristic* untuk menyelesaikan permasalahan komputasi yang biasanya dipadukan dengan penggunaan *machine learning* melalui dua cara yaitu[21] :

- Melakukan pemilihan atau kombinasi *heuristic* yang lebih *simple*
- Menghasilkan *heuristic* baru dari *heuristic* yang sudah ada

Metode ini biasanya diujikan untuk melakukan otomasi. Dalam menyelesaikan persoalan optimasi terdapat dua tingkatan proses yang akan dilalui yaitu *Low Level Heuristic* (LLH) dan *High Level Heuristic*. Solusi yang diperoleh melalui pendekatan *hyper-heuristic* memang

bukan solusi yang pasti optimal atau terbaik layaknya *exact method*, namun melalui *Hyper-heuristic* akan memberikan solusi yang bernilai mendekati optimal. Hal ini disebabkan oleh karakteristik untuk tiap permasalahan optimasi berbeda-beda sehingga, menyebabkan perbedaan pada LLH yang digunakan pun akan berbeda[22]. *High Level Heuristic* kemudian memiliki tugas untuk memanfaatkan LLH agar dapat menghasilkan hasil yang lebih optimal.

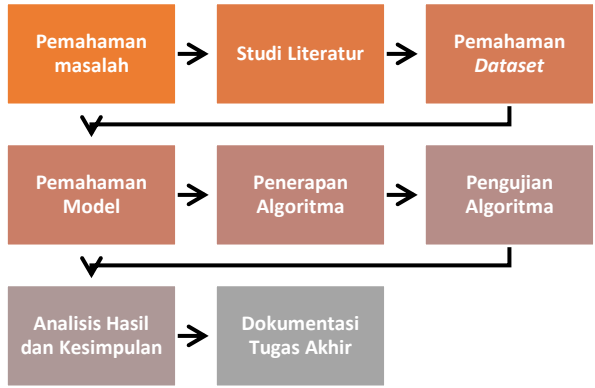
Dua metode utama dalam *High Level Heuristic* yaitu LLH *Selection* dan *Move Acceptance.* LLH *selection* bertujuan menemukan LLH yang akan digunakan untuk menghasilkan solusi yang baru. Kemudian *Move Acceptance* akan memutuskan untuk menerima atau tidak solusi yang dihasilkan[23]. Tujuan akhir penggunaan *Hyper-heuristic* pada penelitian adalah penciptaan metode umum yang memberikan solusi yang optimal berdasarkan LLH.

2. 2. Ringkasan Tinjauan Pustaka

Penelitian tugas akhir ini berdasarkan pada beberapa penelitian terdahulu dan dasar teori yang berkaitan. Keterkaitan tersebut bisa berupa pembahasan mengenai penjadwalan hingga algoritma yang diterapkan. Algoritma yang diusulkan adalah algoritma Self Adaptive Learning-*Great Deluge*. Algoritma ini merupakan modifikasi dari algoritma *Great Deluge* dimana *Great Deluge* akan berperan dalam pencarian solusi dengan nilai yang lebih baik sedangkan *Self Adaptive Learning* akan membantu pemilihan penggunaan *Low Level Heuristic* (LLH). Algoritma yang diusulkan sudah pernah diterapkan pada kerangka kerja HyFlex dan mampu memberikan solusi lebih baik pada beberapa domain. Kemudian pada bab ke-3 akan menjabarkan metodologi dari pelaksanaan pengerjaan tugas akhir.

BAB III
METODOLOGI

Selanjutnya, pada bab ini akan menjelaskan tahap metodologi yang akan dilaksanakan dalam mengerjakan tugas akhir. Proses pengerjaan akan mengacu pada penelitian terdahulu dan juga dasar-dasar teori yang sudah dibahas pada bab tinjauan pustaka. Metodologi ini digunakan sebagai panduan dan memberikan alur pengerjaan tugas akhir yang sistematis. Alur metodologi pengerjaan tugas akhir dapat dilihat pada gambar 3.1



Gambar 3. 1. Metodologi penelitian

Penjelasan alur pengerjaan tugas akhir adalah sebagaimana berikut

3. 1. Pemahaman Masalah

Sesuai pada Gambar 3.1 pengerjaan tugas akhir dimulai dengan memahami permasalahan yang ada pada studi kasus. Studi kasus yang dijadikan pokok pembahasan adalah permasalahan penjadwalan perawat di beberapa rumah sakit di Norwegia. Secara luas permasalahan ini dikenal sebagai *Nurse Rostering Problem*. Sehingga tujuan dari penelitian ini adalah untuk menemukan solusi

yang optimal untuk penjadwalan perawat dengan mempertimbangkan batasan-batasan yang ada. Hasil akhir dari tahap ini adalah inti permasalahan yang kemudian menjadi topik penelitian dalam tugas akhir.

3. 2. Studi Literatur

Selanjutnya pada tahap studi literatur, dilakukan pematangan persiapan dengan mencari referensi yang mendukung tugas akhir yang akan dikerjakan. Topik yang menjadi fokus pada penelitian ini adalah penjadwalan perawat, sehingga dikumpulkan dan dipelajari literatur yang berkaitan dengan topik tersebut. Pada tahap ini pula, dilakukan penentuan algoritma yang akan digunakan untuk menyelesaikan masalah. Dalam memilih algoritma penyelesaian, dilakukan analisis terhadap metode yang digunakan penelitian-penelitian sebelumnya.

3. 3. Pemahaman Dataset

Tahap berikutnya adalah pemahaman *dataset*. *Dataset* merupakan data *benchmark* dari rumah sakit yang berada di Norwegia yang terbuka untuk publik. Pengunduhan dilakukan melalui situs Sintef. Setelah data didapatkan, dilakukan pemahaman terhadap *dataset* yang didalamnya mencakup poin-poin mengenai perawat yang terlibat. *Dataset* ini akan melibatkan sejumlah perawat. Selain poin mengenai perawat terdapat pula poin mengenai jadwal kerja, serta batasan yang perlu diperhatikan

3. 4. Pemahaman Model

Penelitian menggunakan data *benchmark* sehingga secara umum pemodelan matematika telah dilakukan pada penelitian sebelumnya. Sehingga setelah memahami data, maka penulis harus memahami pula model matematikanya. Melalui pemahaman model maka penulis akan lebih mudah untuk kemudian melakukan penerapan algoritma.

3. 5. Penerapan Algoritma

Setelah pembuatan model matematika berdasarkan langkah sebelumnya, dilakukan penerapan algoritma penyelesaian. Algoritma yang akan diimplementasikan adalah algoritma *Self Adaptive Learning - Great Deluge*. Bahasa pemrograman yang akan digunakan adalah Java.

3. 6. Pengujian Algoritma

Melanjutkan tahap sebelumnya yang akan menerapkan algoritma, maka pada tahap ini adalah untuk memastikan bahwa hasil akhir yang dicapai memenuhi *hard constraint* dan memaksimalkan pemenuhan bagi *soft constraint*. Pengujian algoritma ditujukan untuk mengetahui kinerja dari algoritma berdasarkan skenariosasi yang dilakukan.

3. 7. Analisis Hasil dan Kesimpulan

Langkah ini dilakukan dengan melakukan analisis terhadap performa solusi yang dihasilkan terhadap permasalahan yang menjadi pokok bahasan. Setelah dianalisis, kemudian dilakukan penarikan kesimpulan terhadap kinerja dari algoritma tersebut.

3. 8. Dokumentasi Tugas Akhir

Terakhir, pada tahap ini, akan dilakukan dokumentasi terkait tahapan-tahapan yang dilaksanakan selama pengerjaan tugas akhir. Hasil analisis serta kesimpulan juga akan didokumentasikan beserta pernyataan yang mendukung. Luaran dari tahap ini adalah laporan tugas akhir yang menjadi bukti bahwa pengerjaan tugas akhir telah berakhir.

Pada laporan tugas akhir akan terdapat susunan sebagaimana berikut :

1. **BAB I Pendahuluan**
   Bab ini akan menjelaskan latar belakang, rumusan masalah, tujuan masalah serta manfaat pengerjaan tugas akhir.
2. **BAB II Dasar Teori**
   Pada bab selanjutnya akan dijelaskan penelitian-penelitian sebelumnya yang terkait serta berbagai teori yang mendukung permasalahan yang diangkat pada tugas akhir.
3. **BAB III Metodologi**
   Bab ini akan menjelaskan alur pengerjaan tugas akhir mulai dari pemahaman permasalahan hingga pembuatan dokumen tugas akhir.
4. **BAB IV Perancangan**
   Pada bab ini akan menjelaskan tahap persiapan yang dilakukan beserta penjelasan mengenai data-data yang digunakan.
5. **BAB V Implementasi**
   Bab ini berisikan penjelasan dari implementasi dari alur pengerjaan tugas akhir yang telah dibahas pada Bab sebelumnya.
6. **BAB VI Hasil dan Pembahasan**
   Berisikan analisis hasil yang diperoleh dari penyelesaian permasalahan penjadwalan serta pembahasan yang menjelaskan hasil tersebut
7. **BAB VII Kesimpulan dan Saran**
   Bab terakhir ini akan berisikan kesimpulan dari penelitian yang dilakukan serta saran ataupun rekomendasi yang dapat dimanfaatkan untuk penelitian mendatang

BAB IV
PERANCANGAN

Bab ini akan memaparkan rancangan pengerjaan tugas akhir yang telah dibahas pada BAB III. Pada bab perancangan ini akan meliputi pemahaman *dataset*, pemahaman model matematika, pembuatan solusi awal, pemodelan algoritma dan pembuatan skenario.

4. 1. Pemahaman *Dataset*

Dataset benchmark yang digunakan pada tugas akhir ini merupakan data dari rumah sakit yang berada di Norwegia. *Dataset* terdiri dari 7 dokumen *excel* yang menggambarkan 7 unit berbeda pada rumah sakit. Dokumen *excel* ini, pada pembahasan selanjutnya akan disebut OpTur. Setiap dokumen *excel* berisikan 5 *sheet* yang mencakup berbagai hal sebagaimana berikut:

- *Sheet Employee* yang berisikan data mengenai perawat pada sebuah unit. Data yang disediakan terdiri dari id setiap perawat yang terlibat, jumlah jam kerja perawat setiap minggunya, ketentuan perawat akan bekerja pada akhir pekan dan kompetensi yang dimiliki oleh perawat
- *Sheet Shift* yang menjabarkan *shift* yang dibutuhkan mulai dari hari Senin hingga Minggu selama jangka waktu tertentu.
- *Sheet Manpower Plan* yang membahas perencanaan tenaga kerja pada unit secara detail. Pada *sheet* ini dijelaskan jumlah perawat yang dibutuhkan untuk masing-masing *shift.*
- *Sheet Constraint* yang berisikan batasan dan masing-masing nilainya. Batasan yang dibahas meliputi *hard constraint* dan juga *soft constraint.*
- *Sheet Pattern* yang mendaftar pola *shift* baik yang diinginkan maupun tidak diinginkan.

Setiap OpTur memiliki jumlah data dan ketentuan yang berbeda-beda sebagaimana pada tabel 4.1.1 mengenai kompleksitas tiap OpTur berikut. Berdasarkan tabel tersebut,

OpTur7 merupakan unit dengan ukuran terkecil dengan jumlah perawat yang terlibat sebanyak 15 orang perawat dan 6 variasi *shift*.

Tabel 4. 1. Kompleksitas setiap OpTur

Unit	Total Perawat	Jenis *Shift*	Panjang Periode (hari)	Total Perencanaan (jam)
OpTur1	51	9	84	19170,0
OpTur2	82	9	42	12819,0
OpTur3	29	8	42	4159,5
OpTur4	30	8	168	17352,0
OpTur5	20	9	28	2280,0
OpTur6	54	5	84	18978,0
OpTur7	15	6	42	2209,5

4. 2. Pemahaman Model Matematika

Model matematika yang diterapkan telah dijabarkan pada bab II subbab 2.2.3.1 mengenai pemodelan matematika. Sebagaimana telah dijelaskan, maka fungsi tujuannya adalah meminimalkan nilai pelanggaran (*penalty*) pada penjadwalan. Persamaan (1) merupakan model matematika dari fungsi objektif.

$$Minimize: f = \sum_{m=1}^{9} K_m p_m \qquad (1)$$

Penjumlahan dilakukan kepada pelanggaran yang dilakukan pada setiap *soft constraint* yang diterapkan. Bobot (K_m) setiap pelanggaran pada penelitian ini diasumsikan bernilai 1. Hal ini dilakukan karena terbatasnya informasi yang didapatkan oleh penulis dalam mengerjakan penelitian.

4. 3. Pembuatan Solusi Awal

Dalam pembuatan solusi awal, penting untuk memastikan bahwa solusi yang dihasilkan *feasible*. *Feasible* adalah keadaan dimana solusi yang dihasilkan sudah tidak melanggar seluruh *hard constraint* yang ada. Hasil akhir dari solusi awal berupa tabel yang kolomnya merepresentasikan panjang periode sedangkan barisnya menggambarkan jumlah perawat yang akan dilibatkan.

Penyusunan alokasi *shift* dilakukan secara acak dilakukan pada perawat secara acak dengan mendahulukan pengalokasian *shift* didahulukan alokasi pada akhir pekan (hari ke-5 dan ke-6) lalu kemudian hari kerja. Pengalokasian dilakukan melakukan penukaran *shift* hingga akhirnya menemukan solusi yang *feasible*. Pada beberapa *hard constraint* dilakukan pengecekan ketika solusi sedang dibuat sedangkan lainnya dilakukan saat solusi sudah terbentuk.

4. 4. Perancangan Algoritma

Tahap selanjutnya merupakan perancangan algoritma yang diusulkan. Penelitian ini akan melibatkan 3 variasi *Low Level Heuristics* (LLH) yaitu *2-Exchange, 3-Exchange* dan *Double 2-Exchange*. Penjelasan mengenai LLH sebagaimana pada tabel 4.4.1 berikut.

Tabel 4. 2. Penjelasan *Low Level Heuristic*

No	LLH	Penjelasan
1.	*2 - Exchange*	Melibatkan 2 perawat yang berbeda yang kemudian bertukar *shift* pada satu hari yang sama
2.	*3- Exchange*	Pertukaran antara 3 perawat yang berbeda antara 3 *shift* pada satu hari yang sama
3.	*Double 2- Exchange*	Pertukaran yang dilakukan mirip sebagaimana 2-Exchange namun berlaku selama 2 hari yang boleh berurutan ataupun tidak. *Shift* yang ditukar harus memiliki kategori yang sama.

4.4.1. Perancangan *Great Deluge*

Algoritma *Great Deluge* sering dianalogikan seperti seseorang yang menghindari banjir. Algoritma ini akan mencari nilai solusi berdasarkan *Neighborhood Search*. Apabila solusi baru lebih baik dari solusi awal maka solusi baru akan diterima dan disimpan. Hingga nanti di akhir algoritma akan mengembalikan nilai terakhir sebagai solusi akhir yang dihasilkan. Algoritma ini memiliki kemampuan toleransi terhadap nilai solusi yang lebih buruk. Skenariosasi dilakukan dengan melakukan perubahan pada nilai *decay rate*. Detail *pseudocode* dari algoritma *Great Deluge* ada pada Gambar 4.4.1 sebagaimana dibawah ini.

```
    Great Deluge Algorithm
1   Input:  initial solution Sol
2   Initiate water level wL
3   Initiate decay rate d
4   for i < iteration
5      if (penalty newSol < penalty Sol) OR
6         (penalty newSol <= wL)
7           then
8           // Accept new Solution
9              Sol = new Sol
10          else
11          // Reject new Solution
12          If (penalty newSol < Sbest)
13               Sbest <- new Sol
14       end if
15      end if
16        wL = wL- d
17  end for
18  return Sbest
19  End Great Deluge
```

Gambar 4. 1. Pseudocode Algoritma Great Deluge

4.4.2. Perancangan Algoritma Self Adaptive Learning – Great Deluge

Pada algoritma ini, dilakukan modifikasi pada algoritma *Great Deluge*. Algoritma *Self Adaptive* akan berperan dalam melakukan pemilihan LLH yang memiliki nilai

yang paling baik. Pemilihan LLH dilakukan dengan komposisi 75% LLH yang menghasilkan solusi lebih baik dan sisanya 25% secara acak dari LLH yang ada pada permasalahan. Nilai *water level* diatur sama dengan nilai dari solusi awal. Apabila nilai penalti yang diperoleh dari hasil optimasi lebih kecil dari solusi awal maka solusi akan disimpan sebagai solusi sekarang. Namun, jika nilai yang didapatkan lebih besar dari nilai solusi awal dan juga lebih besar dari nilai *water level* maka solusi akan ditolak.

```
    Self Adaptive Learning – Great Deluge Algorithm
1   Input:  initial solution, Sol
2   Set stopping condition, St
3   Initiate water level, wL
4   Initiate decay rate, d
5   Set a list of Low Level Heuristic, LH of size S
6   Set a list of Well performed low level heuristic, WH
7   P.initialSolution(0)
8   S_best <- P.getPenalty(0)
9   S_current <- P.getPenalty(0)
10  while St is not met do
11     if LH is empty
12        then fill with
13        75 % WH
14        25 % LH chosen randomly
15     end if
16        if (penalty new Sol < S_current) OR
17           (penalty new Sol <= wL)
18           then
19           // Accept new Solution
20             S_current <- newSol
21        else
22           // Reject new Solution
23           if (penalty newSol < S_best)
24              S_best <- new Sol
25           end if
26        end if
27         wL = wL- d
28  end while
29  return S_best
30  End Great Deluge
```

Gambar 4. 2. Pseudocode Algoritma *Self Adaptive Learning - Great Deluge*

Performa dari penggunaan algoritma utama yaitu *Self Adaptive Learning -Great Deluge* (SAGD) akan dibandingkan dengan algoritma *Hill Climbing* dan *Great Deluge* dalam mengoptimalkan penjadwalan perawat.

4. 5. Pembuatan Skenario

Skenariosasi dilakukan dengan tujuan untuk melihat dengan kondisi seperti apakah optimasi dihasilkan dengan lebih baik. Pada algoritma dilakukan perubahan pada nilai *decay rate.* Persamaan yang digunakan untuk mendapatkan nilai *decay rate* adalah sebagaimana berikut pada persamaan (18)

$$decay\ rate\ = \frac{water\ level\ -\ desired\ value}{jumlah\ iterasi} \quad (18)$$

Sedangkan untuk menentukan *desired value*, dalam perhitungan ini menggunakan persentase dari nilai solusi awal. Sehingga persamaan (19) sebagaimana dibawah merupakan cara untuk menentukan nilai *desired value*..

$$desired\ value = \alpha * nilai\ solusi\ awal \quad (19)$$

Berdasarkan persamaan (19) dapat dilihat bahwa perhitungan melibatkan nilai *alpha* (α) . Besar variabel *alpha* ini yang akan dijadikan dasar skenariosasi percobaan. Jumlah iterasi yang digunakan sebanyak 1.000.000 iterasi. Perbandingan nantinya dilakukan kepada algoritma utama *Self Adaptive Learing - Great Deluge* (SAGD) serta algoritma *Great Deluge* dan *Hill Climbing* sebagai pembanding.

4. 6. Ringkasan Perancangan

Bagian perancangan merupakan langkah awal yang menggambarkan pengerjaan tugas akhir. Perancangan dimulai dari pemahaman *dataset* yang akan diolah, pemahaman terhadap model matematika yang digunakan, rancangan pembuatan solusi awal, perancangan algoritma *Self Adaptive Learning – Great Deluge* dan pembuatan

skenario yang akan diuji coba. Pada bab selanjutnya akan dilakukan proses implementasi dari perancangan yang sudah dibuat.

Halaman ini sengaja dikosongkan

BAB V
IMPLEMENTASI

Bab V mengenai implementasi akan menjelaskan penerapan perencanaan yang telah dibuat pada bab VI tentang perancangan. Bab ini akan meliputi pembacaan *dataset*, pembuatan solusi awal hingga optimasi dari solusi awal. Optimasi dilakukan dengan menggunakan algoritma *Self Adaptive Learning – Great Deluge* (SAGD)

5. 1. Pembacaan *Dataset*

Dataset yang digunakan pada penelitian ini terdiri dari 7 *file excel* yang masing-masing memiliki 5 *sheet* (*employee, shift, manpower plan, constraint,* dan *pattern*) di dalamnya. Proses diawali dengan melakukan pembacaan seluruh *sheet* dari *dataset* ke dalam program.

Selain melakukan pembacaan *dataset* dilakukan juga penyimpanan informasi dari *sheet* tadi. Langkah selanjutnya adalah membuat objek sebagai wadah berisikan informasi yang akan dibutuhkan dalam proses pembuatan solusi awal dan optimasi. Informasi bisa bernilai sebuah angka namun bisa juga berupa sebuah keterangan "Yes" ataupun "N/A".

Setiap kolom berisikan data akan disimpan dengan tipe data yang sesuai. Objek ini yang nantinya akan dipanggil pada method yang membutuhkan. Detail masing-masing tipe data ada pada tabel 5.1 di bawah ini.

Tabel 5. 1. Tipe data penyimpanan setiap *sheet*

Sheet	**Data**	**Tipe data**
Employees	ID	*integer*
	Work week	*double*
	Working Weekends	*integer array*
	Competence	*String*
Shifts	ID	*integer*
	Duration [Monday-Sunday]	*double array*
	Shift category	*integer*
	Shift name	*String*
	Start Time	*Localtime*
	End Time	
	Competence Needed	*String*
Manpower Plan	*Shift*	*String*
	Plan [Monday-Sunday]	*integer array*
	ID	*integer*
Constraint	*Hard constraint* 1	*boolean*
	Hard constraint 2	
	Hard constraint 3	
	Hard constraint 4	
	Hard constraint 5	*Localtime*
	Hard constraint 6	*integer*
	Hard constraint 7	
	Soft constraint 1	
	Soft constraint 2	
	Soft constraint 3	
	Soft constraint 4	
	Soft constraint 5	
	Soft constraint 6	*boolean*
	Soft constraint 7	
	Soft constraint 8	*integer*
	Soft constraint 9	
Pattern	Start Day	*integer*
	Shift pattern	*String array*

5. 2. Pembuatan Jadwal Solusi Awal

Solusi awal berupa jadwal yang *feasible* merupakan luaran dari tahap ini. Solusi dikatakan *feasible* jika mampu memenuhi keseluruhan *hard constraint* yang berlaku. Pada penelitian kali ini terdapat 7 *hard constraint* yang harus dipenuhi.

Tahapan ini dimulai dengan membuat matriks 2 dimensi dimana kolom merepresentasikan hari yang harus dijadwalkan dan barisnya akan menggambarkan perawat yang terlibat pada unit rumah sakit tersebut. Matriks secara *default* berisi 0 yang menandakan memiliki makna bahwa perawat tidak memiliki jadwal *shift*.

5.2.1. Penerapan *Hard Constraint*

Pengecekan pada *hard constraint* dilakukan secara satu persatu. Hal ini dilakukan dengan pembuatan *method* yang sesuai dengan setiap *hard constraint* yang ada.

5.2.1.1. Penerapan Hard Constraint 1

Pengecualian berlaku pada *hard constraint* 1 tidak terdapat *method* yang diperlukan untuk melakukan pengecekan. Hal ini, karena secara *default* sudah terpenuhi dengan dibuatnya matriks 2 dimensi yang berisikan nilai 0. *Hard constraint* 1 memastikan bahwa setiap perawat maksimal ditugaskan kepada 1 *shift*.

5.2.1.2. Penerapan Hard Constraint 2

Penerapan pada *hard constraint* 2 dilakukan dengan membuat *method needs* yang akan menghitung *shift-shift* yang telah terjadwal. Lalu, *method boolean assignHC2* digunakan dalam mengalokasikan *shift* perawat dengan mempertimbangkan jumlah *shift* yang dibutuhkan. Apabila *shift* yang dialokasikan tidak melebihi kebutuhan maka akan bernilai *true*, dan bernilai *false* jika sebaliknya.

Terdapat pula *method boolean checkHC2* dengan tujuan untuk melakukan pengecekan terhadap *shift*

yang sudah dijadwalkan tadi. Jika jadwal yang terbentuk sudah sesuai dengan kebutuhan maka akan bernilai *true* dan bernilai *false* jika sebaliknya

5.2.1.3. Penerapan Hard Constraint 3

Tahap pertama dalam menerapkan *hard constraint* ke-3 ini adalah dengan membuat *method allWorkHour* yang digunakan untuk menghitung seluruh jam kerja perawat setiap minggu. Hasil perhitungan ini kemudian disimpan pada *array*. Kemudian dibuat *method averageWorkHour* untuk menghitung rata-rata jam kerja tiap perawat yang sudah disimpan tadi. Perhitungan rata-rata dilakukan dengan membagi total jam kerja dengan total periode panjadwalan dalam mingguan.

Dilakukan pula pembuatan *method boolean checkHC3* yang bertujuan untuk melakukan pemeriksaan terhadap jadwal solusi yang dihasilkan. Apabila jadwal melebihi batas yang ditentukan maka akan bernilai *false* yang mana berarti harus dilakukan pertukaran *shift*.

5.2.1.4. Penerapan Hard Constraint 4

Terdapat 2 *method* utama yang digunakan untuk menerapkan *hard constraint* 4 yaitu *assignHC4Com* dan *assignHC4Weekend*. *Method boolean assignHC4Com* berfungsi untuk memastikan *shift* yang membutuhkan keahlian dijadwalkan kepada perawat yang memiliki kompetensi. *Method* ini akan bernilai *true* jika *shift* dijadwalkan sesuai dengan perawat yang memiliki kompetensi.

Method boolean assignHC4Weekend bertujuan untuk menjadwalkan perawat pada akhir pekan sesuai dengan perencanaan kerja yang berlaku. *Method* ini akan bernilai *true* jika penjadwalan di akhir pekan dilakukan sesuai perencanaan kerja. Selanjutnya pengecekan terhadap ketentuan *hard constraint* dilakukan pada masing-masing *method* tadi, dengan *method boolean*

checkHC4Com dan *method boolean checkHCWeekend*. *Method* akan bernilai *true* apabila jadwal yang dihasilkan sudah memenuhi ketentuan.

5.2.1.5. Penerapan Hard Constraint 5

Hard constraint 5 memiliki tujuan untuk menghindari perawat mendapatkan pasangan yang dilarang pada jadwal kerjanya. Pasangan *shift* yang dilarang ini didapatkan dari peerhitungan jam pulang *shift* pertama dan jam mulai *shift* kedua. Apabila jaraknya waktu kosong tidak sesuai dengan ketentuan *hard constraint* 5 maka akan tercatat sebagai pasangan *shift* yang dilarang.

Tahapan penerapan *hard constraint* 5 ini diawali dengan membuat daftar pasangan-pasangan *shift* yang tidak boleh berurutan baik pada hari kerja maupun pada akhir pekan. Selanjutnya dibuat *method boolean* assignHC5 yang akan bernilai *true* asalkan *shift* yang terjadwal bukan termasuk dalam pasangan *shift* yang dilarang. Kemudian seperti pada *hard constraint* lainnya dilakukan pengecekan dengan *method boolean checkHC5* untuk memastikan apakah jadwal yang sudah terbentuk tidak terdapat pasangan *shift* yang dilarang.

5.2.1.6. Penerapan Hard Constraint 6

Tahap pertama dalam menerapkan hard constrant ini adalah dengan membuat *method boolean checkFreeWeekend* yang digunakan untuk melihat apakah perawat memiliki waktu libur minimal tiap minggu. *Method* tadi akan bernilai *true* jika perawat benar mendapatkan wakttu libur yang sesuai ketentuan dan bernilai *false* jika sebaliknya. Kemudian dibuat juga *method boolean* checkHC6 yang akan bernilai *true* jika pada solusi jadwal yang dihasilkan perawat mendapatkan waktu libur sesuai dengan ketentuan pada *hard constraint* 6.

5.2.1.7. Penerapan Hard Constraint 7

Terakhir, *hard constraint* 7 bertujuan untuk memastikan batas waktu kerja maksimal setiap minggu perawat tidak terlampaui. Maka, perlu dilakukan perhitungan jam kerja setiap minggunya dengan menggunakan *method sumTotals*. Selanjutnya dibuat *method boolean assignHC7* yang apabila *shift* dialokasikan ke jadwal kerja yang memenuhi *hard constraint* 7 maka akan bernilai *true*. Setelah *shift* terjadwalkan dibuat *method boolean checkHC7* untuk melakukan pengecekan jadwal yang terbentuk.

5. 3. Perhitungan Penalti Solusi Awal

Ketika seluruh *hard constraint* terpenuhi maka, solusi yang dihasilkan dapat disebut sebagai solusi yang *feasible*.. Perhitungan penalti dimulai dengan membuat *method totalPenalty* yang nantinya akan menjumlah nilai penalti dari pelanggaran dari tiap *soft constraint*.

5.3.1 Nilai Penalti *Soft constraint* 1

Sebagaimana sudah dibahas pada subbab 2.2.3.1 mengenai pemodelan matematika. *Soft constraint* pertama memiliki tujuan untuk menghindari terlalu banyak hari kerja berurutan dengan kategori *shift* yang sama. Dalam penghitungannya dibuat *method double* penaltySC1 yang melakukan identifikasi terlebih dahulu terhadap jumlah maksimal hari berurutan pada masing-masing kategori *shift* (*day*, *evening*, *night*). Jika batas maksimal tidak bernilai "N/A" maka dilakukan perhitungan pelanggaran. Namun jika bernilai "N/A" maka perhitungan penalti tidak dilakukan

5.3.2 Nilai Penalti *Soft constraint* 2

Kemudian *soft constraint* kedua ditujukan untuk menghindari terlalu sering bekerja pada hari yang berurutan. Perhitungan penalti batasan ini dilakukan dengan membuat *method penaltySC2*. *Method* ini

nantinya memastikan batas maksimal hari kerja berurutan yang boleh dilakukan. Apabila bernilai 0 atau "N/A" maka perhitungan tidak dilakukan. Namun, jika tidak sama dengan 0 atau "N/A" maka perhitungan pelanggaran terhadap *soft constraint* dilakukan.

5.3.3 Nilai Penalti *Soft constraint* 3

Jika pada *soft constraint* pertama dibahas batas maksimal maka pada *soft constraint* ke-3 ini akan dibahas batas minimal dilakukannya penjadwalan dengan hari kerja berurutan dengan kategori *shift* yang sama. *Method penaltySC3* dibuat dengan melihat batas minimal yang berlaku untuk ketiga kategori *shift* (*day, evening* dan *night*). Apabila nilai yang tertera pada batasan adalah "N/A" maka tidak dilakukan perhitungan nilai penalti. Namun, jika nilai batasan yang berlaku tidak sama dengan "N/A" maka *method* akan melakukan perhitungan.

5.3.4 Nilai Penalti *Soft constraint* 4

Soft constraint 4 bertujuan untuk menghindari terlalu sedikit bekerja pada hari yang berurutan. Dilakukan perhitungan *penalty* dengan *method penaltySC4*. *Method* ini memeriksa batas minimal yang berlaku pada batasan. Apabila tidak bernilai "N/A" maka *method* akan melakukan perhitungan dengan melihat pelanggaran yang dilakukan terhadap *soft constraint* 4. Apabila batas minimal bernilai "N/A" maka sebaliknya.

5.3.5 Nilai Penalti *Soft constraint* 5

Method penaltySC5 dibuat untuk melakukan perhitungan pada penalti *soft constraint* 5. *Soft constraint* 5 sendiri bertujuan untuk membatasi penyimpangan dari batas minimal maupun maksimal jumlah *shift* yang dijadwalkan pada setiap kategori *shift*. Pada *method* ini terdapat variabel *min* dan *max* dengan tipe *integer* yang nantinya akan menyimpan batas minimal dan maksimal

yang sesuai dengan batasan. Jika batas tidak bernilai "N/A" maka perhitungan penalti dilakukan dan sebaliknya.

5.3.6 Nilai Penalti *Soft constraint* 6

Perawat yang terlibat memiliki kontrak kerja yang berlaku untuk periode waktu tertentu. Hal ini sudah dibahas pada bab IV sub bab 4.1 mengenai pemahaman *dataset*. Pada bab tersebut dibahas bahwa terdapat perencanaan detail yang mengatur kebutuhan *shift* di unit rumah sakit. Dalam pengaplikasiannya *method* penaltySC6 perlu membandingkan antara jumlah jam kerja yang direncanakan dan jam kerja yang dimiliki perawat. Nilai penalti diperoleh dari nilai selisih waktu antara keduanya.

5.3.7 Nilai Penalti *Soft constraint* 7

Nilai penalti *soft constraint* 7 diperoleh melalui perhitungan yang dilakukan *method* penaltySC7. Batasan ini pada dasarnya memperbanyak kemungkinan perawat memiliki hari libur berurutan. Jika terdapat hari libur yang berurutan maka, nilai penalti akan berkurang dan begitu pula sebaliknya. Pada *method* ini melibatkan sebuah variabel yang menghitung waktu libur yang tidak berurutan pada jadwal yang dihasilkan.

5.3.8 Nilai Penalti *Soft constraint* 8

Soft constraint 8 dimaksudkan untuk memperbanyak kemungkinan terjadi pola *shift* yang diinginkan (*wanted pattern*). Dalam melakukan perhitungan terhadap *soft constraint* ini, dibuat *method penaltySC8*. *Method* akan mengacu pada jumlah pola *shift* yang diinginkan Semakin banyak *wanted pattern* yang terbentuk, maka semakin berkurang nilai penaltinya.

5.3.9 Nilai Penalti *Soft constraint* 9

Terakhir, *soft constraint* ke-9 bertujuan untuk meminimalkan pola *shift* yang tidak diinginkan (*unwanted pattern).* Hampir sama dengan *soft constraint* 8 hanya saja batasan ini akan mengacu pada pola yang tidak diinginkan terbentuk pada jadwal kerja. Perhitungan penalti pada batasan ini akan dilakukan dengan membuat *method penaltySC9*.

5. 4. Penyimpanan Jadwal Solusi Awal

Setelah mengetahui nilai penalti yang dimiliki maka solusi akan disimpan terlebih dahulu untuk kemudian dilanjutkan pada tahap optimasi solusi. Solusi awal akan disimpan ke *file* dengan ekstensi .*txt* dan berisikan berisikan baris dan kolom. Nomor-nomor yang muncul dalam *file* yang dihasilkan merepresentasikan *shift* yang dialokasikan pada unit tersebut.

5. 5. Optimasi Solusi Awal

Optimasi yang dilakukan akan melibatkan 3 *Low level heursitic* (LLH) yang sebelumnya dijabarkan pada bab IV sub bab 4.4 tentang perancangan algoritma. Low level *heuristic* yang dimaksud adalah *2-exchange, 3-exchange dan double 2-exchange.*

5.5.1. Penerapan Algoritma *Self Adaptive Learning – Great Deluge*

Algoritma diterapkan dengan terlebih dahulu membuat *method void* SAGD. Skenariosasi yang dilakukan sebagaimana telah dibahas pada sub bab 4.5 adalah melakukan perubahan pada *decay rate* secara spesifik pada nilai *variable alpha* (α) yang terlibat. Pada *method* juga diinisiasi perulangan yang dilakukan untuk optimasi yaitu sebanyak 1.000.000 iterasi. *Low level heuristic* dipilih dengan bantuan konsep *Self Adaptive Learning.* Dari keseluruhan *low level heuristic* yang dipilih oleh konsep ini 75% berasal dari LLH yang nantinya akan

menghasilkan nilai penalti lebih kecil atau dengan kata lain solusi yang lebih baik. Sedangkan, 25% sisanya akan dipilih secara acak dari LLH yang ada.

Solusi yang berhasil memberikan nilai penalti lebih kecil akan diterima dan menggantikan solusi saat ini. Namun apabila solusi memiliki nilai penalti yang lebih besar terdapat perlu dilakukan perbandingan terlebih dahulu dengan *water level*. Apabila lebih rendah atau sama dengan *water level* maka solusi memiliki kemungkinan diterima. Nilai *water level* yang ada akan semakin berkurang seiring berjalannya iterasi algoritma.

5. 6. Penyimpanan Hasil Optimasi

Hasil optimasi yang dihasilkan akan tersimpan dalam bentuk *file* dengan ekstensi *.txt*. Isi dari *file* akan membentuk matriks 2 dimensi yang secara vertikal merepresentasikan perawat dan secara horizontal menggambarkan panjang periode penjadwalan.

5. 7. Ringkasan Implementasi

Langkah-langkah yang dilakukan pada tahapan ini berdasarkan pada bab terdahulunya. Terdapat 3 bagian utama dalam implementasi pada penelitian tugas akhir ini yakni dimulai dari pembacaan *dataset*, dilanjutkan dengan pembuatan solusi awal dan yang terakhir optimasi solusi awal yang sebelumnya telah dihasilkan dengan menggunakan algoritma *Self Adaptive Learning – Great Deluge* (SAGD). Hasil percobaan penelitian beserta analisisnya akan dibahas pada bab VI mengenai hasil dan pembahasan

BAB VI
HASIL DAN PEMBAHASAN

Bab VI menjelaskan mengenai hasil yang diperoleh dari penelitian beserta analisis yang telah dilakukan pada bab V. Bab ini menjelaskan beberapa hal seperti lingkungan uji coba, solusi awal hingga perbandingan algoritma yang diusulkan dan algoritma pembanding.

6. 1. Lingkungan Uji Coba

Penelitian tugas akhir dilakukan pada lingkungan sebagaimana pada tabel 6.1 tentang lingkungan yang digunakan. Lingkungan yang dimaksud meliputi baik perangkat keras dan perangkat lunak.

Tabel 6. 1. Lingkungan uji coba

1. Perangkat Keras	
Laptop	Lenovo B40-80
Processor	IntelCore i3-5005U CPU @ 2.00 GHz 2.00 GHz
RAM	4 GB
Hard Disk	SSD 240 GB
2. Perangkat Lunak	
IntelliJ IDEA	IDE pengembangan program
Microsoft Word	Pembuatan laporan tugas akhir
Microsoft Excel	Analisis dan rekap data mentah hasil uji coba
Notepad	Penyimpanan hasil uji coba

6. 2. Hasil Solusi Awal

Pembuatan solusi awal dibuat dengan upaya memenuhi seluruh *hard constraint* yang ada. Solusi awal yang diperoleh harus memenuhi *hard constraint* (*feasible*) terlebih dahulu untuk kemudian bisa dioptimasi. Penyusunan alokasi jadwal

mengutamakan jadwal pada akhir pekan terlebih dahulu lalu kemudian pada hari kerja (Senin-Jumat).

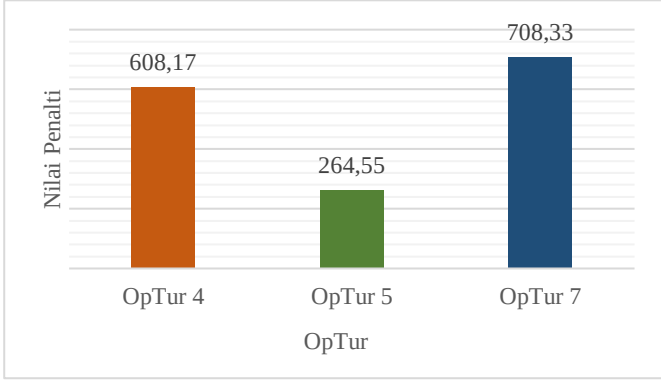
Didapatkan hasil solusi awal dimana terdapat 3 unit yang berhasil dijadwalkan dan *feasible* dari total 7 unit. OpTur yersebut adalah OpTur 4, OpTur 5 dan OpTur 7. Selain ketiga OpTur tersebut tidak berhasil mendapatkan solusi awal yang *feasible.* Hal ini dikarenakan pelanggaran terhadap *hard constraint* 3 yang membatasi penyimpangan antara jam kerja yang perlu dijadwalkan. Kemungkinan lain yang bisa menyebabkan hal tersebut adalah karena adanya keterlibatan faktor acak yang mengakibatkan alokasi *shift* yang tidak *feasible*.

Hasil yang diperoleh pada penelitian berbeda dengan referensi utama [8]. Referensi tersebut memanfaatkan mekanisme *backtrack* yang memungkinkan pengalokasian shift untuk dirunut kembali apabila dihasilkan solusi yang tidak *feasible*. Mekanisme ini tentunya akan memberikan pengaruh besar pada penyusunan jadwal karena dengan mengalokasikan sebuah *shift* maka akan mempengaruhi keputusan pengalokasian selanjutnya.Berikut pada tabel 6.2 detail mengenai hasil solusi awal

Tabel 6. 2. Detail hasil solusi awal

Unit	Hasil Solusi awal	Keterangan
OpTur 1	Tidak *feasible*	Pelanggaran terhadap *Hard constraint* 3
OpTur 2	Tidak *feasible*	Pelanggaran terhadap *Hard constraint* 3
OpTur 3	Tidak *feasible*	Pelanggaran terhadap *Hard constraint* 3
OpTur 4	*Feasible*	-
OpTur 5	*Feasible*	-
OpTur 6	Tidak *feasible*	Pelanggaran terhadap *Hard constraint* 3
OpTur 7	*Feasible*	-

Setelah diperoleh hasil dari ketiga OpTur yang berhasil (OpTur 4, OpTur 5 dan OpTur 7) dilakukan perhitungan penalti pelanggaran terhadap *soft constraint*. Gambar 6.1 merepresentasikan pelanggaran yang terjadi terhadap *soft constraint* yang berlaku. Berdasarkan gambar dapat dilihat bahwa total penalti awal pada OpTur 4 sebesar 608,17. Kemudian OpTur5 memiliki total pelanggaran *soft constraint* sebesar 264,55 dan terakhir OpTur 7 memiliki total nilai penalti awal 708,33 Selain karena pelanggaran yang terjadi banyak sedikitnya pelanggaran yang terjadi bisa disebabkan karena besarnya data yang perlu dijadwalkan.



Gambar 6. 1. Nilai penalti solusi awal

6. 3. Uji Performa Algoritma Self Adaptive Learning – Great Deluge

Sesuai dengan penjelasan pada bab sebelumnya mengenai skenariosasi. Dijelaskan bahwa perubahan yang akan dilakukan adalah kustomisasi pada besar nilai *alpha*. Setelah dilakukan pengujian dilakukan perhitungan rata-rata dari nilai penaltinya. Dilakukan pula perhitungan perbaikan nilai solusi setelah optimasi.

Berikut pada tabel 6.3 merupakan skenario dari percobaan yang dilakukan pada penelitian. Pengujian setiap skenario akan dilakukan dengan iterasi sebanyak 1.000.000.

Tabel 6. 3. Skenario perubahan nilai *alpha* (α)

Skenario	Jumlah iterasi	Nilai *alpha* (α)
1	1.000.000	0,1
2		0,3
3		0,5
4		0,7
5		0,9

6. 3. 1. Hasil Skenario Algoritma pada OpTur 4

Data pada OpTur 4 memilik ukuran yang paling besar diantara ketiga OpTur lainnya. OpTur 4 melibatkan 30 perawat dan dengan periode penjadwalannya selama 168 hari. Dengan menggunakan algoritma *Self Adaptive Learning – Great Deluge* (SAGD) dilakukan optimasi dengan tujuan meminimalkan penalti yang dilanggar. Berikut pada tabel 6.4. merupakan hasil uji coba algoritma dengan skenariosasi. Skenario dengan nilai paling baik diperoleh skenario 5 dengan besar nilai *alpha* adalah 0,9. Perbaikan yang diperoleh skenario ini mencapai 78,88% dari total penalti awal. Sehingga, penalti pada skenario ini mampu mencapai 128,44 dengan nilai terbaik yang dicapai adalah 124,88 dan nilai terburuk 133,02.

Tabel 6. 4. Hasil percobaan pada OpTur 4

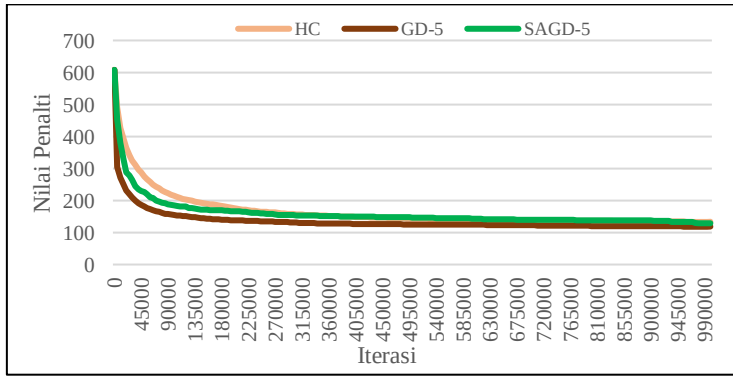
Skenario	Nilai Penalti Awal	Nilai penalti setelah optimasi			Penurunan (%)
		Terburuk	Terbaik	Mean	
1	608,17	142,00	122,66	130,53	78,54%
2		136,29	127,99	131,02	78,46%
3		139,06	124,30	130,41	78,56%
4		135,64	124,26	130,30	78,58%
5		133,02	124,88	128,44	78,88%

Selain pada SAGD pada gambar juga diikut sertakan kedua algoritma yaitu *Hill Climbing* dan *Great Deluge* yang digunakan sebagai pembanding. Ketika dibandingkan dengan algoritma lain seperti *Hill Climbing* dan *Great Deluge* (dengan nilai *alpha* yang sama), SAGD tidak lebih unggul dari *Great Deluge*.Dapat dilihat pada tabel 6.5 di bawah yang melaporkan bahwa algoritma GD mampu memperbaiki hingga 80,50% sedangkan perbaikan yang mampu dilakukan SAGD hanya mencapai 78,88%.

Tabel 6. 5. Perbandingan Optimasi OpTur 4

	HC	GD - 5	SAGD - 5
Terbaik	122,82	110,45	124,88
Terburuk	144,75	127,56	133,02
Rata-rata	133,54	118,60	128,44
Perbaikan (%)	78,04%	80,50%	78,88%

Perolehan nilai penalti per iterasi dapat dilihat pada gambar 6.2 di bawah ini. Pada hasil percobaan di OpTur ke empat ini, algoritma memiliki kinerja yang baik dalam memperbaiki nilai solusi. Hal ini terus berlangsung hingga memasuki iterasi ke-100.000 dimana, dapat dilihat garis yang merepresentasikan nilai penalti hampir berada pada titik yang nilai yang sama hingga akhir iterasi.



Gambar 6. 2. Perbandingan performa algoritma pada OpTur 4

6. 3. 2. Hasil Skenario Algoritma pada OpTur 5

Selanjutnya OpTur 5 memiliki data yang lebih kecil dibandingkan dengan data pada OpTur4. Pada OpTur ini dilakukan pengalokasian pada 20 perawat dengan periode 28 hari. Nilai penalti solusi awal yang dihasilkan dari OpTur ini adalah sebesar 264,55. Berikut pada tabel 6.6 merupakan hasil percobaan dengan 5 skenariosasi yang sebelumnya sudah dibahas.

Percobaan skenario menunjukkan bahwa skenario ke-5 memberikan hasil yang lebih baik dibandingkan dengan skenario lainnya. Pada skenario ke-5 diperoleh nilai rata-rata penalti sebesar 52,64 dengan perbaikan sebesar 80,10% dari nilai penalti awal dengan nilai terbaik yang dicapai adalah 51,82 dan nilai terburuk 53,64.

Tabel 6. 6. Hasil percobaan pada OpTur 5

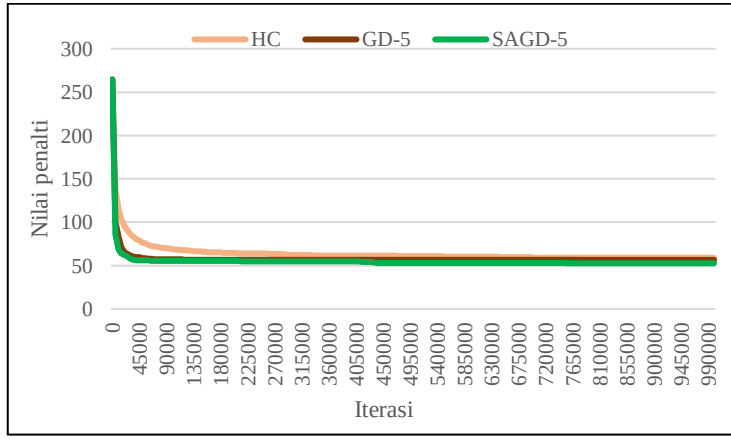
Skenario	Nilai Penalti Awal	Nilai penalti setelah optimasi			Penurunan (%)
		Terburuk	Terbaik	Mean	
1	264,55	61,69	51,22	56,62	78,60%
2		57,10	52,70	55,46	79,04%
3		59,06	55,05	57,00	78,45%
4		62,78	49,69	56,82	78,52%
5		53,64	51,82	52,64	80,10%

Dengan menggunakan skenario yang sama berikut pada tabel 6.7 dilakukan perbandingan dengan algoritma *Great Deluge* serta algoritma *Hill Climbing*. Berbeda dengan OpTur 4, kali ini SAGD dinilai paling unggul jika ditinjau dari persentase perbaikan solusi. Algoritma ini berhasil menurunkan nilai penalti hingga 80,10 % yang mana lebih baik dari pada kedua algoritma pembandingnya.

Tabel 6. 7. Perbandingan optimasi OpTur 5

	HC	GD - 5	SAGD - 5
Terbaik	53,25	52,40	51,82
Terburuk	65,58	58,62	53,64
Rata-rata	58,91	56,53	52,64
Perbaikan (%)	77,73%	78,63%	80,10%

Berdasarkan gambar 6.3 ketiga algoritma mampu memberikan perbaikan pada nilai penalti di awal iterasi. Tetapi, memasuki iterasi ke-70.000 ketiga algoritma terjebak dalam suatu nilai hingga iterasi berakhir. Apabila ditinjau dari nilai akhir yang diperoleh maka algoritma *Self Adaptive Learning – Great Deluge* sedikit lebih unggul dibandingkan dengan kedua algoritma lainnya.



Gambar 6. 3. Perbandingan performa algoritma pada OpTur 5

6. 3. 3. Hasil Skenario Algoritma OpTur7

Terakhir pada OpTur 7 memiliki ukuran data yang lebih kecil. OpTur 7 melibatkan 15 perawat dan panjang periode perencanaan adalah 42 hari. Dengan menggunakan 5 perubahan nilai *alpha* diperoleh data sebagaimana pada tabel 6.8 di bawah ini.

Dapat dilihat bahwa rata-rata nilai penalti yang paling memberikan perbaikan berada di skenario 5. Pada skenario 5 nilai alpha adalah sebesar 0,9. Melalui percobaan tersebut didapatkan penurunan rata-rata sebesar 6,30% dari total penalti awal yang sebesar 708,33. Rata-rata total penalti pada skenario ini adalah sebesar 663,71.

Tabel 6. 8. Hasil percobaan pada OpTur 7

Skenario	Nilai Penalti Awal	Nilai penalti setelah optimasi			Penurunan (%)
		Terburuk	Terbaik	Mean	
1	708,33	673,54	667,30	670,09	5,40%
2		675,04	662,29	667,71	5,73%
3		673,79	666,16	669,21	5,52%
4		669,07	663,99	665,84	6,00%
5		665,72	663,32	663,71	6,30%

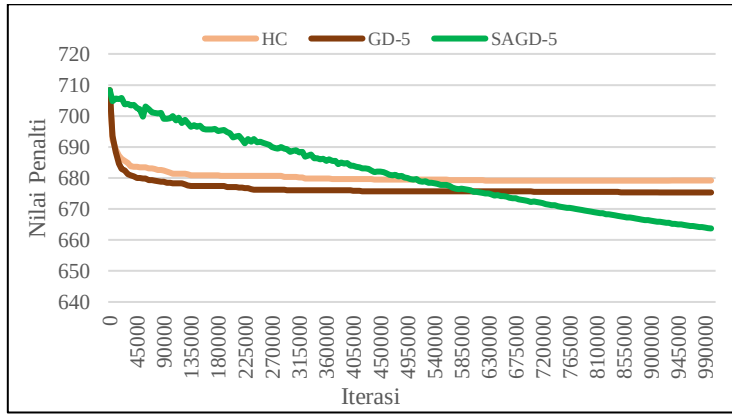
Selain dengan algoritma *Self Adaptive Learning* dilakukan pula uji coba dengan algoritma tunggal *Great Deluge*. Dengan skenario nilai *alpha* yang sama diperoleh hasil perbaikan solusi sebesar 4,66 % dan rata-rata besaran penalti yaitu 675,32 seperti pada tabel 6.9 berikut. Pada tabel juga tertera hasil optimasi yang dilakukan dengan menggunakan algoritma *Hill Climbing*. Algoritma *Hill Climbing* dapat memperbaiki solusi hingga 4,12% yang berarti algoritma tidak lebih baik dari kedua algoritma sebelumnya.

Tabel 6. 9. Perbandingan optimasi OpTur 7

	HC	GD - 5	SAGD - 5
Terbaik	672,95	669,98	663,32
Terburuk	685,76	682,28	665,72
Rata-rata	679,17	675,32	663,71
Perbaikan (%)	4,12%	4,66%	6,30%

Trajectory pada gambar 6.4 menggambarkan alur dari nilai penalti dari awal hingga akhir iterasi. Pada algoritma *Hill Climbing* (HC) dapat dilihat bahwa di awal iterasi nilai penalti mengalami penurunan hingga akhirnya saat memasuki iterasi ke-100.000 nilai penalti yang diperoleh sama hingga akhir iterasi. Hal serupa juga terjadi pada algoritma *Great Deluge* dengan skenario 5. Nilai menunjukkan bahwa algoritma tersebut terjebak pada *local optima*.

Sedangkan pada algoritma *Self Adaptive Learning – Great Deluge* di awal iterasi memang nilai yang diperoleh tidak lebih kecil dari algoritma lain. Hanya penurunan ini terus terjadi hingga akhir iterasi. Sehingga pada akhirnya, saat iterasi berhenti nilai penalti dengan algoritma SAGD mampu mengalami penurunan paling banyak.



Gambar 6. 4. Perbandingan performa algoritma pada OpTur 7

6. 4. Perbandingan Algoritma

Melalui percobaan dapat disimpulkan bahwa perubahan pada nilai *alpha* tidak memberikan pengaruh yang besar terhadap nilai penalti yang diperoleh. Hal ini juga bisa disebabkan karena nilai *decay rate* yang digunakan terlalu kecil karena berdasarkan persamaan (18) dilakukan pembagian dengan jumlah iterasi. Sedangkan, iterasi yang digunakan dalam penelitian adalah sebanyak 1.000.000 iterasi.

Perbandingan yang dibahas pada gambar 6.10 di bawah berdasarkan pada percobaan yang sudah dilakukan. Performa algotima *Great Deluge* (GD) dan *Self Adaptive Learning - Great Deluge* (SAGD) yang tertera merupakan nilai dari skenario 5 yang dinilai memberikan hasil yang paling baik.

Ditinjau dari persentase perbaikan solusi yang dihasilkan performa algoritma *Self Adaptive Learning – Great Deluge* sedikit lebih baik dibandingkan algoritma *Hill Climbing* dan *Great Deluge* untuk OpTur 5 dan OpTur 7. Namun, untuk OpTur4 performa SAGD masih berada di bawah algoritma *Great Deluge* namun masih lebih baik dari *Hill Climbing*.

Tabel 6. 10. Ringkasan performa algoritma ketiga OpTur

Unit	Solusi awal	HC	ΔH	GD	ΔG	SAGD	ΔS
OpTur 4	608,17	133,54	78,04%	118,60	80,50%	128,44	78,88%
OpTur 5	264,55	58,91	77,73%	56,53	78,63%	52,64	80,10%
OpTur 7	708,33	679,17	4,12%	675,32	4,66%	663,71	6,30%

ΔH (%) = Perbaikan dengan algoritma HC terhadap solusi awal (%)
ΔG (%) = Perbaikan dengan algoritma GD terhadap solusi awal (%)
ΔS (%) = Perbaikan dengan algoritma SAGD terhadap solusi awal (%)

6. 5. Perbandingan Nilai Pelanggaran *Soft Constraint*

Selanjutnya pada sub bab ini dilakukan perbandingan nilai penalti dari setiap *soft constraint* yang didapatkan ketiga OpTur dengan menggunakan algoritma *Self Adaptive Learning – Great Deluge* skenario 5. Besaran persentase diartikan sebagai besar dampak pelanggaran *soft constraint* pada total nilai penalti yang diperoleh.

Pada OpTur 4 dan OpTur 5, terdapat sejumlah *soft constraint* yang terpenuhi hal itu dapat dilihat bahwa besar persentase sama dengan 0. Namun beberapa batasan turut memberikan dampak hal ini dapat dilihat pada *soft constraint* 7 yang ternyata memberikan dampak paling besar. *Soft constraint* 7 memberikan dampak sebesar 93,92% dari total penalti yang diperoleh untuk OpTur 4 dan untuk OpTur 5 sebesar 38,87%. *Soft constraint* 7 membahas mengenai pengelompokan dari libur perawat.

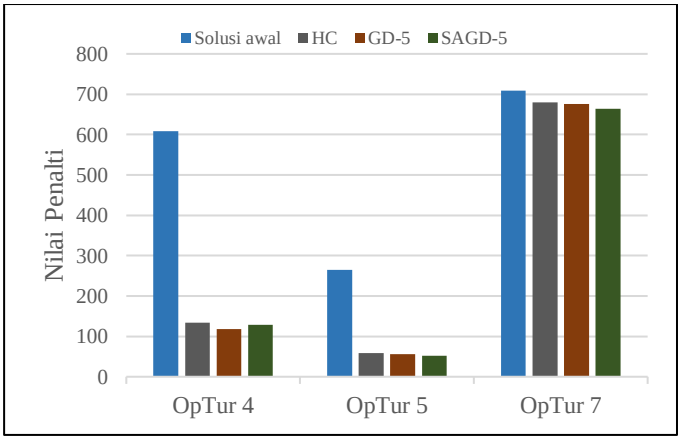
Berbeda dengan sebelumnya untuk OpTur 7, *soft constraint* 8 memberikan dampak sebesar 93,82% dari total penalti yang diperoleh. *Soft constraint* ke-8 sendiri bertujuan untuk membatasi sehingga tidak boleh terlalu sedikit pola *shift* yang dikehendaki.

Tabel 6. 11. Persentase tiap *soft constraint*

	OpTur 4	**OpTur 5**	**OpTur 7**
Nilai Penalti	128,44	52,64	663,71
Soft Constraint 1	1,64%	3,90%	0,00%
Soft Constraint 2	0,00%	3,90%	0,00%
Soft Constraint 3	3,29%	19,48%	1,49%
Soft Constraint 4	0,00%	0,00%	0,00%
Soft Constraint 5	0,00%	31,89%	0,00%
Soft Constraint 6	1,15%	1,97%	1,43%
Soft Constraint 7	93,92%	38,87%	3,25%
Soft Constraint 8	0,00%	0,00%	93,82%
Soft Constraint 9	0,00%	0,00%	0,00%

6. 6. Ringkasan Hasil dan Pembahasan

Pembahasan hasil beserta analisa dijabarkan pada bab ini. Hasil yang diperoleh merupakan percobaan dengan skenario pergantian nilai *alpha* dengan masing-masing percobaan dilakukan dengan iterasi sejumlah 1.000.000 kali. Hasil yang diperoleh berbeda dengan hasil yang tertera pada referensi utama [8] hal ini disebabkan perbedaan dalam menentukan bobot penalti untuk setiap *soft constraint*.
Perbaikan yang terjadi sebagaimana pada visualisasi pada gambar 6.5. di bawah ini. Skenario yang dipilih untuk ditampilkan pada tabel merupakan skenario 5 untuk *Great Deluge* dan *Self Adaptive Learning – Great Deluge*



Gambar 6. 5. Perubahan penalti pada ketiga OpTur

Setiap OpTur memberikan hasil yang berbeda meski diselesaikan dengan algoritma yang sama. Terdapat beberapa hal yang dapat mempengaruhi perolehan nilai tersebut seperti, besarnya data yang perlu dijadwalkan ataupun disebabkan karena ketentuan yang berlaku pada unit OpTur tersebut.
Beberapa *soft constraint* memberikan dampak yang besar pada perolehan total penalti dari suatu OpTur. *Soft constraint* 7 memberikan dampak sebesar 93,92% untuk OpTur 4 dan

38,87% untuk OpTur 5 dari total penalti yang diperoleh dengan algoritma SAGD. Dimana soft constraint 7 ini membahas tentang pengelompokan hari libur dari perawat. Sedangkan untuk OpTur 7 kontribusi perolehan nilai paling besar disebabkan oleh *soft cosntraint* ke-8 yang membahas mengenai pola shift yag diinginkan.

Terdapat beberapa hal yang dapat disimpulkan dari penelitian ini. Selanjutnya, kesimpulan dan saran perbaikan akan dibahas pada bab 7 tentang kesimpulan dan saran.

Halaman ini sengaja dikosongkan

BAB VII
KESIMPULAN DAN SARAN

Bab terakhir pada laporan tugas akhir ini akan membahas tentang kesimpulan yang diperoleh dari hasil uji coba yang telah dibahas pada bab VI. Selain itu, pada bab ini pula terdapat saran yang bisa dipertimbangkan untuk penelitian mendatang.

7.1. Kesimpulan

Berdasarkan percobaan yang telah dilakukan pada penelitian ini, terdapat beberapa kesimpulan yang diperoleh adalah sebagai berikut:

1. Algoritma yang digunakan pada penyusunan solusi awal mampu menghasilkan solusi jadwal yang *feasible* untuk 3 (OpTur 4, OpTur 5 dan OpTur 7) dari 7 total unit rumah sakit.
2. Algoritma *hybrid* yaitu *Self Adaptive Learning – Great Deluge* yang digunakan memberikan hasil yang sedikit lebih baik dari pada algoritma. Hal ini terbukti dengan perbaikan pada OpTur5 dan OpTur7 dengan menerapkan nilai alpha (α) sebesar 0,9 yang secara berurutan mencapai 80,10%, dan 6,30%. Namun secara khusus untuk OpTur4, algoritma SAGD memberikan hasil yang tidak lebih baik dari algoritma *Great Deluge*. Hal ini berdasarkan pada perbaikan dengan algoritma *Great Deluge* yang mencapai 80,50% sedangkan SAGD hanya mencapai 78,88%.
3. Perubahan nilai *alpha* pada skenario tidak memberikan dampak yang besar pada nilai penalti yang diperoleh. Selain nilai *alpha*, nilai *decay rate* juga dinilai terlalu kecil sehingga tidak memberikan dampak yang besar pada penerimaan solusi.
4. Waktu yang diperlukan untuk melakukan penjadwalan berbeda untuk setiap OpTur. Hal tersebut bisa

disebabkan karena perbedaan besar data maupun perbedaan pada ketentuan yang berlaku pada unit data.

5. Waktu yang diperlukan dalam menghasilkan solusi dengan skenario 5 memberikan hasil yang lebih baik. Namun, membutuhkan waktu yang lebih lama dibandingkan skenario lainnya.
6. Pada OpTur 4 dan OpTur 5, *soft constraint* 7 memberikan dampak yang besar dalam perolehan total nilai penalti. Sedangkan untuk OpTur 7, *soft constraint* yang memberikan dampak terbesar adalah *soft constraint* 8. Hal ini kemudian bisa menjadi pertimbangan bagi rumah sakit dalam menentukan urgensitas dari *soft constraint* yang ada.

7.2. Saran

Melalui percobaan yang dilakukan pada penelitian ini, maka saran yang bisa diberikan untuk pertimbangan penelitian selanjutnya adalah :

1. Algoritma yang menghasilkan jadwal solusi awal perlu lebih ditingkatkan agar bersifat lebih fleksibel. Sehingga, solusi awal yang *feasible* bisa dihasilkan dari seluruh unit.
2. Penelitian ini hanya melakukan skenariosasi kepada tingkat *decay rate* yang dimiliki *Great Deluge*. Kedepannya bisa dilakukan skenariosasi yang lebih beragam seperti kustomisasi nilai *alpha* ataupun perubahan pada komposisi pemilihan *Low level heuristic.*
3. Pada penelitian digunakan 3 *Low level heuristic* (LLH). Sehingga hanya sedikit opsi bagi algoritma untuk menemukan LLH yang akan memberikan solusi lebih baik. Sehingga pada penelitian mendatang, akan lebih baik jika menambah variasi dari LLH tersebut.

LAMPIRAN A. JADWAL HASIL SOLUSI AWAL

OpTur 7

E\D	D1	D2	D3	D4	D5	D6	D7	D8	D9
E1			A1		D3			A1	A
E2	D3			D		N1	N1		D3
E3		A1	D3			A	A1		
E4	D		D1	A1				D	
E5		A	N1					N1	N1
E6	N1		D	D		D1	D		
E7				D1	N1			D3	
E8		N1			D			A	
E9		D1		N1					
E10	D1	D			D	D	D1	D	D
E11	A1	D3		D3					A
E12			D		D1	A1	A		D1
E13	D				A1				A1
E14		D	A	A				D1	
E15	A	A	A	A	A				D

E\D	D10	D11	D12	D13	D14	D15	D16	D17	D18
E1	N1	N1	N1				D3	A	
E2		A					A		D
E3	A		D			D	D		A1
E4	A1								D
E5			D	A1	A1	N1	N1		N1
E6		D1					D	D	D1
E7	D		D3			D	D1		
E8		A		D	D1	D1	A		A
E9	D	A1						D1	
E10	D3		D1			A		N1	
E11				N1	N1			D3	D3
E12		D					A1		
E13		D		D1	D	D3		A1	
E14	A		A	A	A			A	
E15	D1	D3	A1			A1		D	A

E\D	D19	D20	D21	D22	D23	D24	D25	D26
E1		N1	N1		D	A		D
E2								
E3				D				
E4		D1	D	D1	D1	D	D1	D1
E5	N1						A1	
E6	D			D	N1	N1	N1	
E7	D3	D	D1	N1		D1		
E8					A	A1	A	
E9	D1	A1	A	A	A		A	A
E10	D					D	D	
E11					D		D3	D3
E12				D3	D3			D
E13	A1				A1			
E14	A					A		N1
E15		A	A1	A1		D3	D	A1

E\D	D27	D28	D29	D30	D31	D32	D33	D34
E1			A1	A1	A			
E2	A1	A1		D	N1			
E3	A	A			D3			
E4			A		A		D	
E5				D3		D		A
E6	N1	N1	N1		D	A1	N1	
E7			D1	N1		D3	D	
E8				A	A1	A	A	A1
E9			D3	D		A		
E10	D	D1	D	A		N1		
E11					D			D
E12	D1	D					A1	
E13						D1		N1
E14					D1	D	D1	D1
E15			D	D1			D3	

E\D	D35	D36	D37	D38	D39	D40	D41	D42
E1				D	A		A	A
E2				D		N1		
E3		A1		A				
E4		D3	D	N1			D1	D1
E5	A	N1	N1		D	D		
E6				D1				
E7			A		D	D1	D	D
E8	A1		A		D1	D		
E9			A1	A1	A	A	N1	N1
E10			D3			A1		
E11	D1	D						
E12		A			D3			
E13	N1			D3		D3		
E14	D	D	D1		N1			
E15		D1	D	A	A1		A1	A1

Halaman ini sengaja dikosongkan

LAMPIRAN B. HASIL PERCOBAAN SKENARIO

OpTur 7

Penalti Awal	P	Penalti Optimasi	Δ	Time(s)
	Nilai α = 0.1			
708,33	P-1	669,83	5,43%	87,00
	P-2	667,60	5,75%	89,00
	P-3	667,94	5,70%	98,00
	P-4	672,13	5,11%	89,00
	P-5	673,54	4,91%	90,00
	P-6	672,51	5,06%	85,00
	P-7	670,85	5,29%	84,00
	P-8	670,16	5,39%	90,00
	P-9	667,30	5,79%	90,00
	P-10	669,03	5,55%	83,00
Rata-rata		670,09	5,40%	88,50
Nilai α = 0.3				
708,33	P-1	664,77	6,15%	100,00
	P-2	669,83	5,43%	90,00
	P-3	666,00	5,98%	87,00
	P-4	662,29	6,50%	102,00
	P-5	668,51	5,62%	91,00
	P-6	663,62	6,31%	88,00
	P-7	671,24	5,24%	95,00
	P-8	668,80	5,58%	91,00
	P-9	675,04	4,70%	85,00
	P-10	667,03	5,83%	93,00
Rata-rata		667,71	5,73%	92,20
Nilai α = 0.5				
708,33	P-1	670,99	5,27%	91,00
	P-2	673,79	4,88%	85,00
	P-3	670,37	5,36%	89,00
	P-4	666,16	5,95%	87,00
	P-5	667,88	5,71%	87,00
	P-6	666,36	5,92%	97,00
	P-7	668,10	5,68%	89,00

	P-8	669,88	5,43%	85,00
	P-9	669,08	5,54%	90,00
	P-10	669,52	5,48%	95,00
Rata-rata		669,21	5,52%	89,50
Nilai α = 0.7				
708,33	P-1	669,07	5,54%	102,00
	P-2	666,46	5,91%	102,00
	P-3	663,99	6,26%	105,00
	P-4	666,03	5,97%	109,00
	P-5	667,24	5,80%	91,00
	P-6	664,51	6,19%	91,00
	P-7	664,04	6,25%	90,00
	P-8	667,93	5,70%	91,00
	P-9	664,81	6,14%	94,00
	P-10	664,34	6,21%	92,00
Rata-rata		665,84	6,00%	96,70
Nilai α = 0.9				
708,33	P-1	663,34	6,35%	100,00
	P-2	663,55	6,32%	157,00
	P-3	663,32	6,35%	118,00
	P-4	663,53	6,32%	109,00
	P-5	663,48	6,33%	114,00
	P-6	663,45	6,34%	112,00
	P-7	663,37	6,35%	108,00
	P-8	663,53	6,32%	106,00
	P-9	665,72	6,01%	104,00
	P-10	663,80	6,29%	106,00
Rata-rata		663,71	6,30%	113,40
Δ = Perbaikan(%), P = Percobaan ke-				

LAMPIRAN C. JADWAL HASIL OPTIMASI

OpTur 7

E\D	D1	D2	D3	D4	D5	D6	D7	D8	D9
E1								A1	A
E2		A1		D	A	N1	N1		
E3	A1	A			D	A	A1	D3	A1
E4	N1		D3	A					D1
E5		D	A	N1	N1				D3
E6	D	N1	N1			D1	D	N1	N1
E7	D1								D
E8	A	A	A	A	A1			D1	A
E9	D	D1							
E10			D	D1	D1	D	D1	D	
E11	D3	D	A1						
E12				A1	D3	A1	A	A	
E13			D1	D3	D				
E14									
E15		D3	D	D				D	D

E\D	D10	D11	D12	D13	D14	D15	D16	D17	D18
E1	A1					D	D	A	
E2									
E3						D3	A1		
E4	D	D	D						D
E5	A		D	A1	A1	N1	N1		
E6								A1	A1
E7	N1	N1	N1				D3	D	D1
E8	A	A		D	D1	A		A	A
E9		D1	A1				D	D1	D
E10								D	D3
E11	D3	A	A	N1	N1				
E12						D	A		
E13		A1	D3	D1	D	A1	A		
E14	D1	D	D1	A	A			N1	N1
E15	D	D3				D1	D1	D3	A

E\D	D19	D20	D21	D22	D23	D24	D25	D26
E1	D	N1	N1	N1			N1	N1
E2				A	A1			
E3						A	A	
E4	D	D1	D	D	D			
E5				A1		A	A	
E6	D3				N1	N1		D
E7		D	D1	D1	D3	D	D	
E8	A							
E9	D1	A1	A				D	A
E10	A1				D1	D		D1
E11							D3	A1
E12				D	A			D3
E13						D3	A1	
E14	N1				D	D1	D1	D
E15		A	A1	D3	A	A1		

E\D	D27	D28	D29	D30	D31	D32	D33	D34
E1			D3	A1	D3	D		
E2	A1	A1		D				
E3	A	A				D	D	
E4			D1	D1	D1			
E5								A
E6	N1	N1	N1		A1	N1	N1	
E7			D	D	D			
E8					A	A	A	A1
E9				A	A	A		
E10	D	D1	A1	N1	N1		D1	
E11						A1	D3	D
E12	D1	D						
E13						D3	A1	N1
E14			A	A				D1
E15			D	D3	D	D1	D	

E\D	D35	D36	D37	D38	D39	D40	D41	D42
E1		D	D			D	A	A
E2		A	A		D	N1		
E3								
E4			D	D3	A		D1	D1
E5	A	N1	N1	N1	N1			
E6			A	A1				
E7					D3	D1	D	D
E8	A1	D3						
E9			A1	A	A	A1	N1	N1
E10				D	A1	A		
E11	D1	A1						
E12			D3	A		D3		
E13	N1							
E14	D	D1	D1	D1	D1			
E15		D		D	D	D	A1	A1

Hasil penjadwalan ketiga OpTur (OpTur 4, OpTur5, dan OpTur7) beserta kode program lengkap dapat diakses pada : *https://tinyurl.com/TA-Penjadwalan*

Halaman ini sengaja dikosongkan

DAFTAR PUSTAKA

[1] M. Willis-Shattuck, P. Bidwell, S. Thomas, L. Wyness, D. Blaauw, and P. Ditlopo, "Motivation and retention of health workers in developing countries: A systematic review," *BMC Health Serv. Res.*, vol. 8, no. May 2014, 2008.

[2] F. R. Pratama, "Penjadwalan Perawat Rumah Sakit Dengan Mempertimbangkan Level Dan Beban Kerja FAUZAN RUSLI PRATAMA," pp. 1–5, 2015.

[3] E. Burke, P. Causmaecker, G. Vanden Berghe, and H. Van Landeghem, "The state of the art of nurse scheduling," *J. Sched.*, vol. 7, no. 6, pp. 441–499, 2004.

[4] J. Sinthia and T. Lely, "Implementasi Vertex Graph Colouring , Particle Swarm Optimization , Dan Constraint Based Reasoning Untuk University Timetabling Problem (Studi Kasus : Fti Untar)," pp. 97–104, 2014.

[5] A. Acan and A. Ünveren, "A two-stage memory powered Great Deluge algorithm for global optimization," *Soft Comput*, 2015.

[6] W. Saputra, *PERMASALAHAN OPTIMASI LINTAS DOMAIN PENDEKATAN HYPER-HEURISTIC MENGGUNAKAN ALGORITMA SELF ADAPTIVE LEARNING – GREAT DELUGE COMPARISON OF CROSS DOMAIN OPTIMIZATION PROBLEM SOLVING METHODS WITH HYPER- HEURISTIC APPROACH USING SELF ADAPTIVE LEARNING-GREAT DE*. 2018.

[7] Sintef, "Nurse Rostering Dataset," 2010. [Online]. Available: https://www.sintef.no/en/digital/applied-mathematics/optimization/health-care-optimization/#Resourcemanagement.

[8] M. Stølevik, T. E. Nordlander, A. Riise, and H. Frøyseth, "A hybrid approach for solving real-world nurse

rostering problems," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2011.

[9] R. A.-M. Nabeel, "Hybrid genetic algorithms with great deluge for course timetabling," *IJCSNS Int. J. Comput. Sci. Netw. Secur.*, vol. 10, no. 4, pp. 283–288, 2010.

[10] F. Z. Prayogo, *Optimasi Penjadwalan Staf dengan Menggunakan Algoritma Self-Adaptive Learning Hyper-Heuristic (Studi Kasus: RSIA Kendangsari Surabaya)*. 2018.

[11] V. A. Supoyo and A. Muklason, "Pendekatan Hyper Heuristic Dengan Kombinasi Algoritma Pada Examination Timetabling Problem," *Ilk. J. Ilm.*, vol. 11, no. 1, p. 34, 2019.

[12] Kusnanto and M. Ester, *Profesi dan Praktik Keperawatan Profesional*. Jakarta: EGC, 2004.

[13] S. Praptianingsih, *Kedudukan Hukum Keperawatan dalam Upaya Pelayanan Kesehatan di Rumah Sakit*. Jakarta: PT Raja Grafindo Persada, 2006.

[14] S. M. Al-Najjar and S. H. Ali, "Staffing and Scheduling Emergency Rooms in Two Public Hospitals: A Case Study," *Int. J. Bus. Adm.*, vol. 2, no. 2, pp. 137–148, 2011.

[15] B. Jaumard, F. Semet, and T. Vovor, "A Generalized Linear Programming Model for Nurse Scheduling," *Eur. J. Oper. Res.*, vol. 107, no. 1, pp. 1–18, 1998.

[16] A. A. El Adoly, M. Gheith, and M. Nashat Fors, "A new formulation and solution for the nurse scheduling problem: A case study in Egypt," *Alexandria Eng. J.*, vol. 57, no. 4, pp. 2289–2298, 2018.

[17] M. Stølevik, T. Eric, and A. Riise, "A mathematical model for the nurse rostering problem," *SINTEF Tech. Rep. no. A19133*, pp. 1–9, 2011.

[18] A. Muklason, "Hyper-heuristics and fairness in examination timetabling problems," no. December, 2017.

[19] G. Dueck, "New Optimization Heuristics: The Great

Deluge Algorithm and the Record-to-Record Travel," *J. Comput. Phys.*, vol. 104, no. 1, pp. 86–92, 1993.

[20] P. Eles, K. Kuchcinski, and Z. Peng, "Optimization Heuristics," in *System Synthesis with VHDL*, 2013, pp. 137–163.

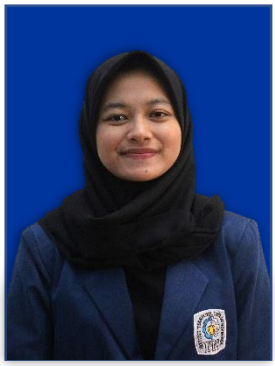
[21] A. Muklason, "Solver Penjadwal Ujian Otomatis Dengan Algoritma Maximal Clique dan Hyper-heuristics," *Semin. Nas. Teknol. Informasi, Komun. dan Ind. 9*, pp. 18–19, 2017.

[22] N. D. Angresti, A. Djunaidy, and A. Muklason, "khazanah informatika Hyper-heuristics untuk Penyelesaian Masalah Optimasi Lintas Domain dengan Seleksi Heuristik berdasarkan Variable Neighborhood Search," pp. 51–60.

[23] S. S. Choong, L.-P. Wong, and C.-P. Lim, "Automatic design of hyper-heuristic based on reinforcement learning," *Inf. Sci. (Ny).*, vol. 436–437, pp. 89–107, 2018.

Halaman ini sengaja dikosongkan

BIODATA PENULIS



Penulis lahir di Ujung Pandang pada tanggal 14 November 1998. Penulis menempuh pendidikan formal di SD Islam Athirah di kota Makassar. Kemudian jenjang pendidikan selanjutnya ditempuh penulis di kota Surabaya, lebih tepatnya di SMP Al-Hikmah yang kemudian dilanjutkan di SMA Al-Hikmah hingga akhirnya melanjutkan studi S1 di Departemen Sistem Informasi, Institut Teknologi Sepuluh Nopember, Surabaya. Selama menempuh masa studinya Penulis juga ikut serta dalam sejumlah organisasi dan kepanitiaan. Tercatat penulis merupakan anggota organisasi BEM Fakultas Teknologi Informasi dan Komunikasi sebagai staf dan tahun berikutnya sebagai staf ahli. Penulis juga berkesempatan melakukan magang di PT Telkom Indonesia bagian Data Management.

Dalam upaya memperoleh gelar Sarjana Komputer (S.Kom), Penulis memilih laboratorium Rekayasa Data dan Intelejensi Bisnis dengan topik utama optimasi penjadwalan. Untuk kepentingan penelitian, penulis dapat dihubungi melalui e-mail: widyafira@gmail.com.