p-ISSN : 2301-8402, e-ISSN : 2685-368X , available at : https://ejournal.unsrat.ac.id/index.php/elekdankom

Heuristic Selection Algorithms Comparison for Examination Timetabling Using Hyper-Heuristics Framework

Perbandingan Algoritma Heurustic Selection untuk Penjadwalan Ujian Menggunakan Kerangka Kerja *Hyper-Heuristics*

Rizal Risnanda Hutama, Tsani Nahdliyah, Aelisa Nailin Nabila, Ahmad Muklason
Dept. of Information Systems, Institut Teknologi Sepuluh Nopember, Kampus ITS Sukolilo, 60111, Indonesia
e-mails : r.risnanda@its.ac.id ; tsaninadia@gmail.com ; aelisanailin08@gmail.com ; mukhlason@is.its.ac.id
Received: 25 April 2023; revised: 3 June 2023 ; accepted: 7 June 2023

***Abstract* — Examination timetabling is an example of a problem in operations research and an exciting issue in higher education. For example, the problems that occur in an institution are the limited number and capacity of rooms, few exam supervisors, and limited exam time. So far, exam scheduling is done manually, so it is pretty time-consuming. Until now, no algorithm can solve this problem in polynomial time. So that in the optimization world, the problem of scheduling exams is an NP-Hard problem. Solving this problem can be done using a heuristic algorithm to produce a reasonably good solution fairly quickly. Currently, the algorithm has been developed in various ways with a *hyper-heuristic* approach. This study will discuss the scheduling of exams using carter benchmark datasets. The charter dataset is a dataset of real-world exam scheduling problems. The trials carried out in this study were to apply *hyper-heuristics* by combining a move acceptance algorithm, namely late acceptance hill climbing, with several heuristic selection algorithms to get the heuristic selection algorithm with the best performance to be combined. The result is the combination of Reinforcement Learning with Late Acceptance *Hill Climbing* can outperform the results of the comparison algorithm in most of the datasets.**

Key words* — *Carter Datasets; Examination Timetabling*; *Hyper-Heuristics*; *Late Acceptance Hill Climbing*; *Reinforcement Learning*; *Tabu Search

***Abstrak* — Penjadwalan ujian merupakan salah satu contoh permasalahan dalam riset operasi serta permasalahan yang menarik untuk dihadapi di pergurugan tinggi. Sebagai contoh, permasalahan yang terjadi di sebuah perguruan tinggi yaitu terbatasnya jumlah dan kapasitas ruangan, terbatasnya pengawas ujian, serta terbatasnya waktu ujian. Selama ini, penjadwalan ujian dilakukan secara manual sehingga cukup memakan waktu. Hingga saat ini belum ada algoritma yang dapat menyelesaikan permasalahan tersebut dalam waktu polinomial. Sehingga dalam dunia optimasi, permasalahan penjadwalan ujian merupakan *NP-Hard problem*. Penyelesaian permasalahan ini dapat dilakukan dengan menggunakan algoritma heuristic untuk menghasilkan solusi yang cukup baik dengan waktu yang cukup singkat. Saat ini algoritma telah berkembang secara variatif dengan pendekatan *hyper-heuristics*. Penelitian ini akan membahas penjadwalan ujian menggunakan *benchmark dataset carter*. Dataset carter merupakan dataset permasalahan penjadwalan ujian *real-world*. Uji coba yang dilakukan pada penelitian ini yaitu menerapkan *hyper-heuristics* dengan mengombinasikan sebuah algoritma *move acceptance* yaitu *late acceptance hill climbing* dengan beberapa algoritma *heuristic selection* untuk mendapatkan algoritma *heuristic selection* dengan performa terbaik untuk dikombinasikan. Hasilnya yaitu kombinasi *Reinforcement Learning* dengan *Late Acceptance Hill Climbing* mampu mengungguli hasil algoritma pembanding dalam sebagian besar dataset.**

Kata kunci* — Dataset Carter; *Hyper-Heuristics*; *Late Acceptance Hill Climbing*; Penjadwalan Ujian; *Reinforcement Learning*; *Tabu Search

I. PENDAHULUAN

Penjadwalan merupakan suatu permalasahan yang sering didengar dalam bidang akademik ataupun keprofesian. Penjadwalan dalam bidang akademik pada perguruan tinggi biasanya mengalami permasalahan dalam menentukan jadwal ujian untuk matakuliah yang diambil oleh setiap mahasiswa. Permasalahan tersebut telah banyak dilakukan pada banyak literatur. Munculnya permasalahan penjadwalan ujian dikarenakan oleh matakuliah apa saja yang diambil oleh setiap mahasiswa. Solusi yang diharapkan pada permasalahan tersebut yaitu tidak ada jadwal ujian matakuliah yang diambil oleh setiap mahasiswa bertabrakan. Beberapa permasalahan penjadwalan ujian yang terjadi di Indonesia yaitu terjadi pada jurusan teknik informatika UNESA [1], jurusan matematika Universitas Brawijaya [2], dan fakultas teknik Universitas Surabaya [3].

Saat ini, permasalahan penjadwalan ujian tergolong *NP-Hard Problem* [4] yang dapat diselesaikan menggunakan algoritma *heuristic*. Pada umumnya masalah penjadwalan tersebut merupakan salah satu permasalahan yang dapat diselesaikan dengan menggunakan optimasi. Optimasi merupakan proses pencarian solusi terbaik berdasarkan fungsi objektif dengan mempertimbangkan batasan yang ditentukan. Solusi penyelesaian permasalahan dengan menggunakan optimasi akan dapat membantu dalam menyelesaikan permasalahan dalam waktu yang singkat. Pembuatan jadwal ujian pada suatu perguruan tinggi harus mempertimbangkan matakuliah apa saja

yang diambil oleh setiap mahasiswa, sehingga pada saat ujian tidak ada jadwal ujian matakuliah yang diambil oleh setiap mahasiswa dilaksanakan secara bersamaan.

Studi kasus ini akan menyelesaikan optimasi penjadwalan ujian pada *benchmark dataset carter* [5] dengan mengkombinasikan beberapa algoritma. Algoritma utama (*move acceptance*) yang digunakan dalam penelitian ini yaitu *Late Acceptance Hill Climbing* (LAHC). LAHC dipilih karena implementasinya yang cukup mudah dan mengahsilkan solusi yang cukup bagus. Algoritma utama tersebut akan dikombinasikan dengan beberapa algoritma *heuristic selection* untuk ditemukan kombinasi algoritma yang terbaik dalam menyelesaikan permasalahan yang digunakan. Pendekatan yang digunakan dalam memberikan solusi yaitu *hyper-heuristics*. Pendekatan tersebut memiliki tujuan untuk memilih dan mengkombinasikan *heuristic* yang lebih sederhana atau menghasilkan heuristic baru. Keunggulan dari pendekatan *hyper-heuristics* yaitu tidak tergantung pada problem domain tertentu, sehingga tidak perlu parameter tuning.

Solusi yang dihasilkan dari studi kasus ini akan dilakukan perbandingan. Algoritma yang diterapkan pada kasus ini yaitu algoritma *Late Acceptance Hill Climbing* (LAHC) sebagai *move acceptance* dikombinasikan dengan *Simple Random* (SR), *Tabu Search* (TS) dan *Reinforcement Learning* (RL). Luarannya yaitu hasil optimasi yang paling optimal dari kombinasi algoritma.

A. Optimasi Kombinatorial

Optimasi kombinatorial merupakan cara yang digunakan dalam pencarian semua kemungkinan nilai real dari suatu fungsi objektif. Optimasi kombinatorial akan mencari nilai maksimum atau minimum bergantung pada permasalahan yang akan diselesaikan. Algoritma dalam optimasi kombinatorial digunakan untuk menyelesaikan masalah yang cukup rumit dengan ruang lingkup yang besar.

B. Dataset Carter

Dataset Carter merupakan dataset yang terdiri dari 13 permasalahan jadwal ujian dalam dunia nyata yang terdiri dari 3 sekolah menengah di Canada, 5 universitas di Canada, 1 universitas di America, 1 universitas di UK, dan 1 universitas di mid-east [5]. Data yang didapatkan berupa jumlah mata kuliah dan jumlah mahasiswa dengan keterangan matakuliah yang diambil oleh setiap mahasiswa. Dataset Carter secara keseluruhan dapat dilihat pada Tabel I.

Pada dataset carter, hard constraint yang berlaku yaitu untuk menghindari adanya jadwal ujian yang bentrok jika terdapat seorang siswa yang mengambil ujian tersebut. Sedangkan fungsi tujuannya yaitu untuk meminimalkan pelanggaran soft constraint yang berupa setiap siswa yang dijadwalkan ujian akan mendapatkan timeslot seharusnya memiliki jarak minimal 5 timeslots.

Beberapa penelitian telah dilakukan dengan menggunakan dataset carter. Penelitian yang telah dilakukan memiliki tujuan yang berbeda-beda seperti yang ditampilkan pada Tabel II. Dari Tabel II dapat diketahui bahwa belum terdapat penelitian yang mengidentifikasi algoritma dengan performa terbaik untuk LLH *selection* pada kombinasi algoritma dengan kerangka kerja *hyper-heuristics*.

C. Hyper-Heuristics

Hyper-heuristic adalah metodologi pencarian untuk memilih atau men-*generate* (mengkombinasikan, mengadaptasi) heuristik untuk menyelesaikan suatu masalah optimasi. Pendekatan *hyper-heuristic* bertujuan untuk: (1) memilih dan mengkombinasikan heuristics yang lebih sederhana (*heuristic selection*), atau (2) menghasilkan *heuristics* baru dari komponen heuristics yang sudah ada (*heuristic generation*), untuk memecahkan permasalahan pencarian komputasi yang sangat sulit dilakukan secara manual [6]. Untuk metode *hyper-heuristic* tidak tergantung pada problem domain tertentu saja, sehingga tidak memerlukan parameter tuning, namun dapat melakukan otomasi untuk parameter tuning ini.

Salah satu ciri khas dari *hyper-heuristics* adalah pemisahan letak logika pemrosesan *problem domain* dan metodologi. Komponen utama *problem domain* pada *hyper-heuristic* adalah *objective function, initial solution,* dan memiliki satu set *low level heuristic*. Selain itu, dalam metodologi terdiri dari dua tahapan, diantaranya pemilihan *heuristic selection* dan *move acceptance* [7].

D. Low-Level Heuristics (LLH)

Seperti yang telah dijelaskan pada bagian sebelumnya, *Low-Level Heuristics* (LLH) merupakan bagian dari *hyperheuristics*. Satu LLH merupakan sebuah langkah kecil untuk mengubah solusi yang telah ada disetiap iterasi. Setiap domain permasalahan dapat memiliki LLH yang berbeda. Disetiap iterasi, LLH dipilih menggunakan algoritma tertentu yang difungsikan sebagai *heuristic selection* atau LLH *selection*.

Untuk permasalahan penjadwalan ujian ini, LLH yang digunakan yaitu:

1. Move1, merupakan LLH yang digunakan untuk memindahkan secara random 1 ujian ke 1 timeslot secara random pula.
2. Move2, merupakan LLH yang digunakan untuk memindahkan secara random 2 ujian ke 2 timeslot secara random pula.
3. Swap, merupakan LLH yang digunakan untuk menukarkan 2 timeslot dari 2 ujian secara random.

E. Hill Climbing

Algoritma *Hill Climbing* adalah salah satu metode yang digunakan dalam menyelesaikan permasalahan pencarian jarak terdekat. Cara kerja dari metode ini ialah menempatkan *node* yang akan muncul sedekat mungkin dengan sasarannya sebagai langkah berikutnya dan menggunakan fungsi heuristik untuk proses pengujian [8].

F. Late Acceptance Hill Climbing (LAHC)

Algoritma *Late Acceptance Hill Climbing* (LAHC) merupakan sebuah algoritma pengembangan dari *hill climbing* tradisional. Perbedaan yang membedakan algoritma ini dengan *hill climbing* tradisional yaitu pada penerimaan solusinya.

Pada *hill climbing* tradisional penerimaan kandidat solusi hanya dibandingkan dengan iterasi sebelumnya. Sedangkan pada algoritma LAHC ini, penerimaan kandidat solusi akan dibandingkan dengan fitness array yang berisikan nilai *objective function* dari beberapa iterasi sebelumnya [9].

TABEL I
DATASET CARTER

Dataset	Exam	Student	Timeslot
car-f-92	543	18419	32
car-s-91	682	16925	35
ear-f-83	189	1108	24
hec-s-92	80	2823	18
kfu-s-93	461	5349	20
lse-f-91	381	2726	18
pur-s-93	3158	30032	42
rye-s-93	486	11483	23
sta-f-83	138	549	13
tre-s-92	261	4360	23
uta-s-92	638	21330	35
ute-s-92	184	2750	10
yor-f-83	180	919	21

Penerapan algoritma ini sudah merambah diberbagai bidang permasalahan dan menghasilkan sebuah hasil yang cukup baik. Contoh permasahalan yang pernah diselesaikan dengan algoritma ini yaitu *patient admission* [19], penjadwalan mata kuliah [20], *traveling purchaser problem* [21], dan lain-lain.

G. Tabu Search

Tabu Search adalah sebuah algoritma optimasi yang berbasis pada *local search*. Proses pencarian dilakukan dengan cara memilih solusi terbaik *current neighbourhood solution* yang tidak termasuk solusi *tabu* [22]. Ide dasar dari algoritma *tabu search* adalah mencegah proses pencarian dari *local search* agar tidak melakukan pencarian ulang pada ruang solusi yang pernah ditelusuri dengan memanfaatkan *tabu list. Tabu list* merupakan struktur memori fundamental dalam *tabu search* yang menyimpan atribut dari sebagian *move* yang telah diterapkan pada iterasi-iterasi sebelumnya, sehingga *tabu list* akan digunakan oleh *tabu search* untuk menolak solusi-solusi yang memenuhi atribut tertentu guna mencegah proses pencarian pada daerah solusi yang sama.

Tabu search telah diimplementasikan dalam berbagai permasalahan dengan hasil yang cukup baik. Hal ini mengindikasikan bahwa *tabu search* merupakan algoritma yang reliabel untuk menyelesaikan berbagai permasalahan. Contoh permasalahan yang telah diselesaikan dengan *tabu search* yaitu *project scheduling* [23], VRP [24], *probabilistic orienteering problem* [25], TTP [26], dan lain-lain.

H. Reinforcement Learning

Reinforcement learning merupakan salah satu Teknik dalam *machine learning* dimana terinspirasi dari teori psikologikal yaitu berupa penghargaan dan hukuman. Sebuah heuristic akan mendapatkan nilai tambah sebagai penghargaan apabila dapat menghasilkan solusi yang lebih baik [27]. Serta sebaliknya, heuristic akan mendapatkan nilai kurang sebagai hukuman apabila tidak menghasilkan solusi yang lebih baik. Heuristic disetiap iterasi akan dipilih berdasarkan nilai yang tertinggi. *Reinforcement learning* juga telah diimplementasikan dalam berbagai permasalahan, sebagai contoh *railway scheduling* [28], *traveling salesman problem* [29] dan lain-lain.

TABEL II
PENELITIAN TERDAHULU

Referensi	Detail
Dewi, Shinta. Raras T., dan Febriyora S. P. [7]	Mengusulkan algoritma Tabu Search untuk penyelesaian
Fong, Cheng Weng dkk [10]	Mengusulkan algoritma Hybrid Black Hole untuk penyelesaian
Muklason, Ahmad [11]	Mengusulkan algoritma Maximal Clique untuk pembentukan *feasible solution*
Al-Betar, Mohammed Azmi [12]	Mengusulkan algoritma β-*hill climbing* untuk penyelesaian
Supoyo, Vicha Azthanty, dan Ahmad Muklason [13]	Membandingkan dua algoritma move acceptance yaitu *hill climbing* dan simulated annealing untuk penyelesaian
Burke, Edmund K., dan Yuri Bykov [14]	Mengusulkan algoritma adaptive flex-deluge untuk penyelesaians
Alzaqebah, M., dan S. Abdullah [15]	Mengombinasikan algoritma artificial bee colony dengan late acceptance *hill climbing* untuk penyelesaian
Asmuni, Hishammudin, dkk [16]	Mengusulkan kombinasi fuzzy multiple heuristics ordering untuk pembuatan feasible solution dan penyelesaian
Burke, Edmund K, dkk [17]	Mengusulkan algoritma hybrid variable neighbourhood untuk penyelesaian
Carter, Michael W, dkk [18]	Penyelesaian permasalahan dataset carter dengan beberapa strategi dasar. Sebagai penulis utama dataset carter.

II. METODE

A. Pembuatan Initial Solution

Pembuatan initial solution ini merupakan tahapan awal yaitu untuk membuat sebuah jadwal yang sudah *feasible*. Jadwal yang sudah *feasible* merupakan jadwal yang sudah tidak terjadi pelanggaran pada *hard constraint*. Pembuatan *initial solution* dilakukan dengan menggunakan cara mengurutkan ujian yang memiliki kemungkinan bentrok paling banyak terlebih dahulu. Kemungkinan tersebut dapat diambil dari pencarian ujian untuk setiap siswa. Jika seorang siswa mengambil lebih banyak ujian, maka ujian tersebut akan memiliki kemungkinan bentrok lebih banyak dan akan dijadwalkan pertama. Dengan urutan ujian yang akan dijadwalkan sudah ditentukan, maka ujian-ujian tersebut akan ditempatkan pada timeslot yang memungkinkan. Jika hasil akhir dari cara ini masih terdapat timeslot yang lebih banyak dari batas pada Tabel I, maka cara selanjutnya akan dioptimasi menggunakan LAHC untuk meminimalkan timeslot agar sesuai pada Tabel I.

B. Optimasi Initial Solution

Optimasi ini dilakukan dengan mengimplementasikan algoritma yang telah ditentukan sebelumnya. Penelitian ini dilakukan dengan menggunakan kerangka *hyper-heuristics*. Sehingga seperti yang telah dijelaskan pada bagian sebelumnya, akan terdapat 2 fungsi bagian algoritma yang digunakan. Algoritma pertama berperan sebagai *move acceptance* serta algoritma kedua akan berperan sebagai LLH *selection*. Dari kedua peran algoritma tersebut dapat dikatakan sebagai kombinasi algoritma.

TABEL III
HASIL INITIAL SOLUTION

Dataset	Maks Timeslot	Hasil Timeslot	Penalti Awal	Keterangan
car-f-92	32	32	10.391	Feasible
car-s-91	35	35	11.956	Feasible
ear-f-83	24	24	57.908	Feasible
hec-s-92	18	18	19.456	Feasible
kfu-s-93	20	20	46.617	Feasible
lse-f-91	18	18	22.461	Feasible
pur-s-93	42	38	16.679	Feasible
rye-s-93	23	23	25.111	Feasible
sta-f-83	13	13	200.768	Feasible
tre-s-92	23	22	15.950	Feasible
uta-s-92	35	35	6.151	Feasible
ute-s-92	10	10	44.398	Feasible
yor-f-83	21	21	53.793	Feasible

Algoritma yang akan berperan sebagai *move acceptance* yaitu algoritma LAHC. LAHC ini dipilih sebagai karena penerapannya yang cukup mudah. Hal tersebut disebabkan karena LAHC ini merupakan pengembangan dari algoritma *hill climbing* tradisional sebagaimana yang diketahui *hill climbing* merupakan salah satu algoritma dasar dalam optimasi. Fungsi peran algoritma ini yaitu menentukan apakah kandidat solusi pada setiap iterasi yang dipilih akan diterima atau ditolak.

Algoritma yang akan berperan sebagai LLH *selection* yaitu algoritma tabu search, reinforcement learning, serta simple random. Simple random merupakan pemilihan LLH secara random tradisional tanpa pertimbangan apapun. Fungsi peran algoritma ini yaitu memilih LLH disetiap iterasi sesuai dengan prosedur masing-masing algoritma ketika dijalankan.

C. Perbandingan Hasil Optimasi

Dari beberapa kombinasi algoritma yang telah dijalankan, pada akhirnya akan dilakukan perbandingan terhadap masing-masing hasil yang dikeluarkan. Perbandingan hasil optimasi ini bertujuan untuk mengetahui performa perbandingan algoritma yang terbaik dalam menyelesaikan permasalahan penjadwalan ujian pada dataset carter.

III. HASIL DAN PEMBAHASAN

A. Pembuatan Initial Solution

Hasil yang didapat yaitu seluruh ujian dapat ditempatkan pada jumlah timeslot yang telah ditentukan seperti pada Tabel III. Dataset dikatakan *feasible* apabila seluruh ujian berhasil dijadwalkan dengan jumlah *timeslot* kurang dari yang telah ditentukan. Dari hasil tersebut dapat diketahui bahwa seluruh dataset telah berhasil *feasible* yang memiliki jumlah *timeslot* kurang dari batas maksimal serta memiliki nilai penalti awal. Dataset yang telah *feasible* dapat dilanjutkan pada tahap optimasi untuk meminimalkan nilai penaltinya.

TABEL IV
LINGKUNGAN UJI COBA

Perangkat	Spesifikasi
Processor	Intel Core i7-7500U 4CPU @2.70GHz
RAM	20 GB
Hard Drive	SSD 256 GB

TABEL V
HASIL OPTIMASI SR-LAHC

Dataset	Penalti Awal	Rata-Rata Hasil Optimasi	Penurunan
car-f-92	10.391	6.120	41.1%
car-s-91	11.956	7.465	37.6%
ear-f-83	57.908	46.716	19.3%
hec-s-92	19.456	14.327	26.4%
kfu-s-93	46.617	18.326	60.7%
lse-f-91	22.461	16.133	28.2%
pur-s-93	16.679	7.636	54.2%
rye-s-93	25.111	17.047	32.1%
sta-f-83	200.768	172.695	14.0%
tre-s-92	15.950	11.676	26.8%
uta-s-92	6.151	4.970	19.2%
ute-s-92	44.398	30.892	30.4%
yor-f-83	53.793	46.482	13.6%

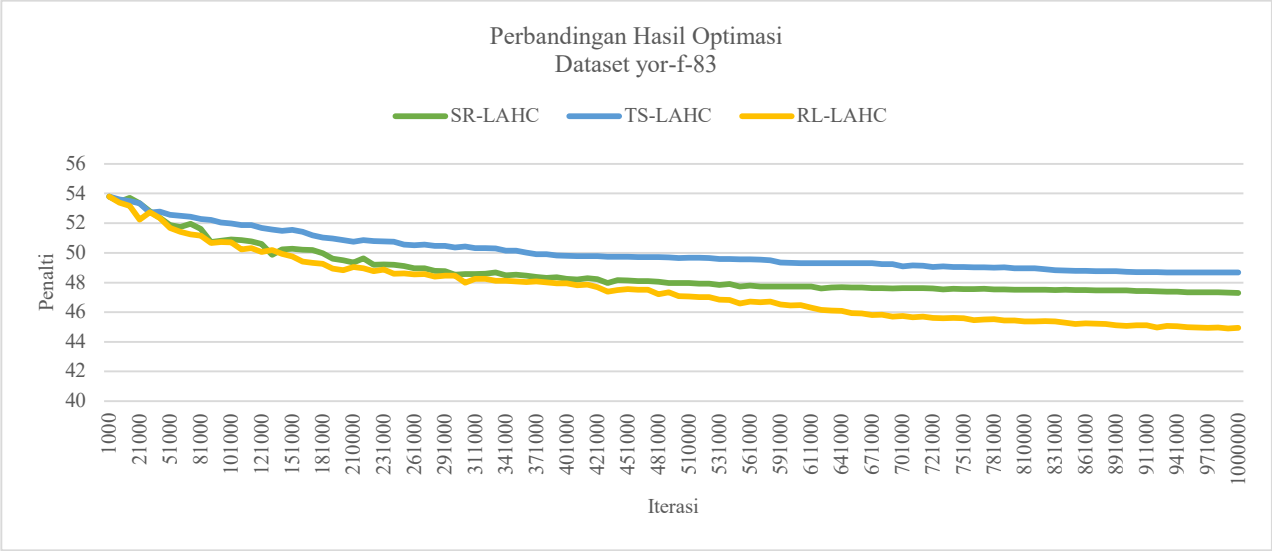
B. Optimasi Initial Solution

Optimasi *initial solution* sangat bergantung pada kemampuan lingkungan uji coba yang digunakan. Penelitian ini menggunakan sebanyak 5 kali perulangan untuk setiap dataset dan setiap algoritma yang digunakan. Dalam setiap perulangannya, optimasi dilakukan dengan menggunakan batasan 1.000.000 iterasi. Setiap skenario, parameter *fitness array* yang digunakan yaitu sepanjang 500. Jumlah tersebut disesuaikan dengan jumlah iterasi yang dilakukan. Jika *fitness array* terlalu panjang dengan total iterasi yang tidak terlalu banyak akan berdampak pada hasil yang tidak terlalu baik, begitupula sebaliknya. Lingkungan uji coba yang digunakan dalam penelitian ini dapat dilihat pada Tabel IV.

Skenario uji coba pertama yaitu penerapan algoritma *simple random* dengan LAHC (SR-LAHC). Hasil yang didapat ditampilkan pada Tabel V yang dapat dikatakan seluruh dataset berhasil menurun nilai penaltinya. Penurunan terbesar terjadi pada dataset kfu-s-93 dengan penurunan 60.7%. Sedangkan penurunan terkecil terjadi pada dataset yor-f-83 dengan penurunan 13.6%.

Skenario uji coba kedua yaitu penerapan algoritma tabu search dengan LAHC (TS-LAHC). Hasil yang didapat merupakan hasil rata-rata dari 5 perulangan dan ditampilkan pada Tabel VI.

p-ISSN : 2301-8402, e-ISSN : 2685-368X , available at : https://ejournal.unsrat.ac.id/index.php/elekdankom



Gambar 1. Perbandingan Hasil Optimasi Dataset yor-f-83

TABEL VI
HASIL OPTIMASI TS-LAHC

Dataset	Penalti Awal	Rata-Rata Hasil Optimasi	Penurunan
car-f-92	10.391	6.214	40.2%
car-s-91	11.956	7.651	36.0%
ear-f-83	57.908	46.784	19.2%
hec-s-92	19.456	14.118	27.4%
kfu-s-93	46.617	18.052	61.3%
lse-f-91	22.461	16.068	28.5%
pur-s-93	16.679	7.642	54.2%
rye-s-93	25.111	17.039	32.1%
sta-f-83	200.768	175.531	12.6%
tre-s-92	15.950	11.334	28.9%
uta-s-92	6.151	5.043	18.0%
ute-s-92	44.398	30.854	30.5%
yor-f-83	53.793	48.114	10.6%

TABEL VII
HASIL OPTIMASI RL-LAHC

Dataset	Penalti Awal	Rata-Rata Hasil Optimasi	Penurunan
car-f-92	10.391	6.114	41.2%
car-s-91	11.956	7.407	38.0%
ear-f-83	57.908	46.667	19.4%
hec-s-92	19.456	13.769	29.2%
kfu-s-93	46.617	18.446	60.4%
lse-f-91	22.461	15.969	28.9%
pur-s-93	16.679	7.649	54.1%
rye-s-93	25.111	17.036	32.2%
sta-f-83	200.768	174.558	13.1%
tre-s-92	15.950	11.178	29.9%
uta-s-92	6.151	4.982	19.0%
ute-s-92	44.398	30.903	30.4%
yor-f-83	53.793	45.931	14.6%

Seluruh dataset berhasil diminimalkan nilai penaltinya yang nantinya akan dibandingkan hasilnya. Sama halnya dengan uji coba pertama menggunakan SR-LAHC, uji coba dengan TS-LAHC juga menemukan hasil yang sama yaitu penurunan terbesar terjadi pada datasset kfu-s-93 dengan penurunan sebesar 61.3% dan penurunan terkecil terjadi pada dataset yor-f-83 dengan penurunan sebesar 10.6%.

Skenario uji coba ketiga yaitu penerapan algoritma reinforcement learning dengan LAHC (RL-LAHC). Hasil yang didapat merupakan hasil rata-rata dari 5 perulangan dan ditampilkan pada Tabel VII. Seluruh dataset berhasil diminimalkan penaltinya. Uji coba ini mengalami sedikit perbedaan dengan sebelumnya. Penurunan terbesar sama dengan uji coba sebelumnya yaitu terjadi pada dataset kfu-s-93 dengan penurunan sebesar 60.4%. Pada penurunan terkecil terdapat perbedaan. Pada uji coba kali ini, penurunan terkecil terjadi pada dataset sta-f-83 dengan penurunan penalti sebesar 13.1%.

C. Perbandingan Hasil Optimasi

Perbandingan dilakukan terhadap seluruh hasil optimasi ketiga skenario. Disetiap dataset akan ditemukan algoritma terbaik dalam melakukan optimasi. Hasil perbandingan tersebut diltampilkan pada Tabel VIII. Pada Tabel VIII dapat dinyatakan algoritma RL-LAHC memiliki performa paling bagus dibandingkan dengan 2 algoritma yang lain. Sebanyak 8 dari 13 total dataset, algoritma RL-LAHC memiliki hasil yang unggul. Pada permasalahan penjadwalan ujian pada dataset carter ini, mekanisme pemilihan LLH dengan cara memberikan sistem “reward and punishment” memiliki performa yang bagus.

Jika dilihat pada grafik trajectory pada Gambar 1, performa algoritma TS-LAHC sangatlah tidak baik bahkan jika dibandingkan dengan SR-LAHC. Algoritma SR-LAHC diawal iterasi mampu bersaing dengan RL-LAHC. Akan tetapi, pada iterasi diatas 120.000 performa RL-LAHC semakin membaik dan mengalahkan SR-LAHC yang cenderung hasilnya stagnan dari iterasi 300.000 hingga 1.000.000 atau akhir iterasi.

TABEL IX
PERBANDINGAN DENGAN PENELITIAN SEBELUMNYA

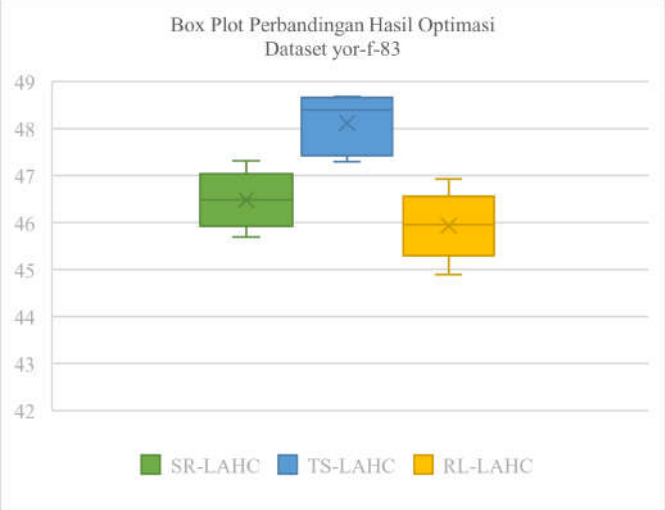
Dataset	RL-LAHC (Our Result)	Dewi [7]	Carter [18]	Supoyo [13]	Alzaqebah [15]
car-f-92	6.114	6.790	6.200	4.590	4.080
car-s-91	7.407	8.520	7.100	4.500	4.740
ear-f-83	46.667	44.480	36.400	37.660	33.720
hec-s-92	13.769	12.500	10.800	11.540	10.590
kfu-s-93	18.446	21.330	14.000	14.810	13.860
lse-f-91	15.969	16.450	10.500	11.900	11.000
pur-s-93	7.649	10.500	3.900	-	-
rye-s-93	17.036	15.510	7.300	-	9.540
sta-f-83	174.558	152.330	161.500	-	157.160
tre-s-92	11.178	11.260	9.600	-	8.140
uta-s-92	4.982	5.330	3.500	3.750	3.330
ute-s-92	30.903	30.180	25.800	-	25.370
yor-f-83	45.931	42.010	41.700	41.780	36.320

TABEL VIII
PERBANDINGAN HASIL OPTIMASI

Dataset	SR-LAHC	TS-LAHC	RL-LAHC
car-f-92	6.120	6.214	**6.114**
car-s-91	7.465	7.651	**7.407**
ear-f-83	46.716	46.784	**46.667**
hec-s-92	14.327	14.118	**13.769**
kfu-s-93	18.326	**18.052**	18.446
lse-f-91	16.133	16.068	**15.969**
pur-s-93	**7.636**	7.642	7.649
rye-s-93	17.047	17.039	**17.036**
sta-f-83	**172.695**	175.531	174.558
tre-s-92	11.676	11.334	**11.178**
uta-s-92	**4.970**	5.043	4.982
ute-s-92	30.892	**30.854**	30.903
yor-f-83	46.482	48.114	**45.931**

Setiap skenario yang dijalankan sebanyak 5 kali tentunya memiliki hasil yang berbeda-beda. Oleh karena itu, selanjutnya akan dilakukan analisis terhadap konsistensi dari masing-masing skenario. Semakin konsisten hasil dari skenario algoritma yang digunakan maka performa algoritma tersebut juga akan semakin baik. Analisis konsistensi algoritma dilakukan dengan menggunakan *box plot*.

Dari perbandingan hasil *box plot* pada Gambar 2, diketahui bahwa seluruh skenario algoritma memiliki hasil yang konsistensi. Dikatakan konsisten karena tidak terdapat *outlier* pada seluruh hasil skenario algoritma. Sehingga ketiga algoritma ini dapat diterapkan dengan baik untuk permasalahan penjadwalan ujian pada dataset carter ini.



Gambar 2. Box Plot Perbandingan Hasil Optimasi Dataset yor-f-83

Selanjutnya pada tahap akhir, hasil yang terbaik dengan menggunakan RL-LAHC akan dibandingkan dengan hasil dari 4 penelitian sebelumnya yang telah disebutkan pada Tabel I. Perbandingan dengan hasil penelitian terdahulu ditampilkan pada Tabel IX. Dari Tabel IX secara sekilas, dapat dikatakan hasil yang kami dapatkan tidak lebih baik daripada penelitian sebelumnya. Hal tersebut, salah satu penyebabnya yaitu disebabkan oleh *running time* atau jumlah iterasi yang tidak sama.

Detail *running time* untuk setiap perulangan tidaklah sama. Penelitian yang dilakukan oleh Carter [18] dijalankan selama 5 hingga 60 menit. Penelitian yang dilakukan oleh Dewi [7] dijalaknkan sebanyak 1.000.000 iterasi. Penelitian yang dilakukan oleh Supoyo [13] dijalankan selama 1 jam. Penelitian yang dilakukan oleh Alzaqebah [15] dijalankan hingga hampir 3 jam. Sedangkan penelitian yang kami lakukan yaitu dengan dijalankan sebanyak 1.000.000 iterasi dengan durasi waktu

maksimal 10 menit untuk 1 kali perulangan. Berdasarkan data tersebut, hasil penelitian ini akan lebih cocok jika dibandingkan dengan penelitian yang dilakukan oleh Dewi. Jika dilakukan perbandingan dengan penelitian yang dilakukan oleh Dewi [7], hasil kombinasi algoritma RL-LAHC yang kami lakukan dapat mengungguli di 7 dari 13 dataset. Artinya, kombinasi ini dapat dikatakan lebih baik daripada Algoritma Tabu Search pada hasil penelitian sebelumnya.

IV. KESIMPULAN DAN SARAN

A. Kesimpulan

Algoritma LLH selection terbaik dengan dikombinasikan dengan algoritma LAHC sebagai move acceptance dalam menyelesaikan permasalahan penjadwalan ujian pada dataset carter ini yaitu *reinforcement learning*. RL-LAHC memiliki hasil yang baik dibandingkan dengan kombinasi yang lain pada 8 dari 13 total dataset yang ada. Penerapan kombinasi sederhana yang dilakukan dengan mengubah LLH selection dapat memberikan hasil yang berbeda. Jika dibandingkan dengan salah satu penelitian yang telah dilakukan sebelumnya dengan jumlah iterasi yang sama, algoritma RL-LAHC dapat menghasilkan solusi yang lebih baik daripada Tabu Search pada penelitian sebelumnya.

B. Saran

Dalam penelitian ini hanya menggunakan algoritma LAHC sebagai *move acceptance*. Selain LAHC, masih banyak algoritma lain yang dapat diimplemtasikan. LAHC ini juga merupakan algoritma *single-based*. Sehingga pada penelitian selanjutnya dapat menggunakan algoritma *move acceptacnce* yang lain dan menggunakan algoritma *population-based* untuk diuji coba.

Dari segi permasalahan, penelitian ini hanya terbatas pada penjadwalan ujian dengan menggunakan dataset carter. Selain itu hanya terbatas dengan menggunakan 3 LLH. Sehingga, pada penelitian selanjutnya dapat dilakukan uji coba pada domain permasalahan yang lain ataupun yang sama dengan dataset yang berbeda serta dengan LLH yang lebih bervariatif.

V. KUTIPAN

[1] A. Qoiriah, "Penjadwalan Ujian Akhir Semester Dengan Algoritma Genetika," *Manaj. Inform.*, vol. 3, no. 2, pp. 33–38, 2014.

[2] N. K. Mawaddah and W. F. Mahmudy, "Optimasi Penjadwalan Perawat Menggunakan Algoritma Genetika," *Kursor*, vol. 2, no. 2, pp. 1–8, 2006, [Online]. Available: wayanfm@ub.ac.id

[3] Suwarno, "Perancangan Sistem Penjadwalan Ujian pada suatu Fakultas dengan Beberapa Jurusan dengan Menggunakan Algoritma Genetika," *Telcomatics J.*, vol. 1, no. 1, pp. 1–11, 2016.

[4] B. A. Aldeeb *et al.*, "Hybrid intelligent water Drops algorithm for examination timetabling problem," *J. King Saud Univ. - Comput. Inf. Sci.*, no. xxxx, 2021, doi: 10.1016/j.jksuci.2021.06.016.

[5] R. Qu, E. K. Burke, B. Mccollum, L. T. G. Merlot, and S. Y. Lee, "A Survey of Search Methodologies and Automated System Development for Examination Timetabling," *J. Sched.*, vol. 12, no. 1, pp. 55–89, 2009, doi: 10.1007/s10951-008-0077-5.

[6] U. R. K. Widayu, A. Mukhlason, and I. Nurkasanah, "Automation and Optimization of Course Timetabling Using the Iterated Local Search *Hyper-Heuristic* Algorithm with the Problem Domain from the 2019 International Timetabling Competition," *3rd 2021 East Indones. Conf. Comput. Inf. Technol. EIConCIT 2021*, pp. 134–138, 2021, doi: 10.1109/EIConCIT50028.2021.9431892.

[7] S. Dewi, R. Tyasnurita, and F. S. Pratiwi, "Solving examination timetabling problem within a hyperheuristic framework," *Bull. Electr. Eng. Informatics*, vol. 10, no. 3, pp. 1611–1620, 2021, doi: 10.11591/eei.v10i3.2996.

[8] V. Y. I. Ilwaru, T. Sumah, Y. A. Lesnussa, and Z. A. Leleury, "Perbandingan Algoritma *Hill Climbing* Dan Algoritma Ant Colony Dalam Penentuan Rute Optimum," *BAREKENG J. Ilmu Mat. dan Terap.*, vol. 11, no. 2, pp. 139–150, 2017, doi: 10.30598/barekengvol11iss2pp139-150.

[9] E. K. Burke and Y. Bykov, "The late acceptance Hill-Climbing heuristic," *Eur. J. Oper. Res.*, vol. 258, no. 1, pp. 70–78, 2017, doi: 10.1016/j.ejor.2016.07.012.

[10] C. W. Fong *et al.*, "University Examination Timetabling Using a Hybrid Black Hole Algorithm," *Int. J. Informatics Vis.*, vol. 6, no. 2, pp. 572–580, 2022, doi: 10.30630/joiv.6.2-2.1092.

[11] 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.

[12] M. A. Al-Betar, "A β *-hill climbing* optimizer for examination timetabling problem," *J. Ambient Intell. Humaniz. Comput.*, vol. 12, no. 1, pp. 653–666, 2021, doi: 10.1007/s12652-020-02047-2.

[13] V. A. Supoyo and A. Muklason, "Pendekatan Hyper Heuristic Dengan Kombinasi Algoritma Pada Examination Timetabling Problem," *Ilk. J. Ilm.*, vol. 11, no. 1, pp. 34–44, 2019, doi: 10.33096/ilkom.v11i1.420.34-44.

[14] E. K. Burke and Y. Bykov, "An adaptive flex-deluge approach to university exam timetabling," *INFORMS J. Comput.*, vol. 28, no. 4, pp. 781–794, 2016, doi: 10.1287/ijoc.2015.0680.

[15] M. Alzaqebah and S. Abdullah, "An adaptive artificial bee colony and late-acceptance hill-climbing algorithm for examination timetabling," *J. Sched.*, vol. 17, no. 3, pp. 249–262, 2014, doi: 10.1007/s10951-013-0352-y.

[16] H. Asmuni, E. K. Burke, J. M. Garibaldi, B. McCollum, and A. J. Parkes, "An investigation of fuzzy multiple heuristic orderings in the construction of university examination timetables," *Comput. Oper. Res.*, vol. 36, no. 4, pp. 981–1001, 2009, doi: 10.1016/j.cor.2007.12.007.

[17] E. K. Burke, A. J. Eckersley, B. McCollum, S. Petrovic, and R. Qu, "Hybrid variable neighbourhood approaches to university exam timetabling," *Eur. J. Oper. Res.*, vol. 206, no. 1, pp. 46–53, 2010, doi: 10.1016/j.ejor.2010.01.044.

[18] M. W. Carter, G. Laporte, and S. Y. Lee, "Examination timetabling: Algorithmic strategies and applications," *J. Oper. Res. Soc.*, vol. 47, no. 3, pp. 373–383, 1996, doi: 10.1057/jors.1996.37.

[19] A. L. aro Bolaji, A. F. Bamigbola, and P. B. Shola, "Late acceptance *hill climbing* algorithm for solving patient admission scheduling problem," *Knowledge-Based Syst.*, vol. 145, pp. 197–206, 2018, doi: 10.1016/j.knosys.2018.01.017.

[20] I. G. A. Premananda and A. Muklason, "Optimasi Penjadwalan Mata Kuliah Menggunakan Algoritma Late Acceptence *Hill Climbing* Berbasis Hiper Heuristik," *JATISI (Jurnal Tek. Inform. dan Sist. Informasi)*, vol. 8, no. 2, pp. 774–782, 2021, doi: 10.35957/jatisi.v8i2.778.

[21] A. Goerler, F. Schulte, and S. Voß, "An Application of Late Acceptance Hill-Climbing to the Traveling Purchaser Problem," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 8197 LNCS, pp. 173–183, 2013, doi: 10.1007/978-3-642-41019-2_13.

[22] M. S. Ariantini and A. M. Dirgayusari, "Implementasi Metode Tabu Search Dalam Penjadwalan Menggunakan Analisa Pieces," *INFORMAL Informatics J.*, vol. 6, no. 2, p. 62, 2021, doi: 10.19184/isj.v6i2.23811.

[23] Y. Ma, Z. He, N. Wang, and E. Demeulemeester, "Tabu search for proactive project scheduling problem with flexible resources," *Comput. Oper. Res.*, vol. 153, no. February, p. 106185, 2023, doi: 10.1016/j.cor.2023.106185.

[24] M. Qiu, Z. Fu, R. Eglese, and Q. Tang, "A Tabu Search algorithm for the vehicle routing problem with discrete split deliveries and pickups," *Comput. Oper. Res.*, vol. 100, pp. 102–116, 2018, doi: 10.1016/j.cor.2018.07.021.

[25] X. Chou, L. M. Gambardella, and R. Montemanni, "A Tabu Search algorithm for the Probabilistic Orienteering Problem," *Comput. Oper. Res.*, vol. 126, p. 105107, 2021, doi:

10.1016/j.cor.2020.105107.
[26] L. Di Gaspero and A. Schaerf, "A composite-neighborhood tabu search approach to the traveling tournament problem," *J. Heuristics*, vol. 13, no. 2, pp. 189–207, 2007, doi: 10.1007/s10732-006-9007-x.
[27] J. Andreanus and A. Kurniawan, "Sejarah, Teori Dasar dan Penerapan Reinforcement Learning: Sebuah Tinjauan Pustaka," *J. Telemat.*, vol. 12, no. 2, pp. 113–118, 2018.
[28] Y. Zhu, H. Wang, and R. M. P. Goverde, "Reinforcement Learning in Railway Timetable Rescheduling," *2020 IEEE 23rd Int. Conf. Intell. Transp. Syst. ITSC 2020*, 2020, doi: 10.1109/ITSC45102.2020.9294188.
[29] A. L. C. Ottoni, E. G. Nepomuceno, M. S. d. Oliveira, and D. C. R. d. Oliveira, "Reinforcement learning for the traveling salesman problem with refueling," *Complex Intell. Syst.*, vol. 8, no. 3, pp. 2001–2015, 2022, doi: 10.1007/s40747-021-00444-4.

Penulis bernama Rizal Risnanda Hutama, lahir di Surabaya, 13 April 1998. Penulis telah menempuh pendidikan formal di SD Negeri Baratajaya Surabaya, SMP Negeri 6 Surabaya, SMA Negeri 5 Surabaya, pendidikan sarjana dan magister di Departemen Sistem Informasi ITS yang telah lulus pada tahun 2022. Saat ini, penulis menjadi dosen baru di Departemen dan Institut yang sama ketika menempuh pendidikan formal. Bidang minat yang ditekuni penulis antara lain, optimasi penjadwalan dan manajemen proyek teknologi informasi. Penulis dapat dihubungi melalui email r.risnanda@its.ac.id.