Terbit *online* pada laman: http://teknosi.fti.unand.ac.id/

Jurnal Nasional Teknologi dan Sistem Informasi

| ISSN (Print) 2460-3465 | ISSN (Online) 2476-8812 |

Arttikel Penelitian

Penerapan Hiperheuristik Berbasis Metode Simulated Annealing untuk Penyelesaian Permasalahan Optimasi Lintas Domain

Nisa Dwi Angresti[a], Arif Djunaidy[a], Ahmad Mukhlason[a]

[a]*Institut Teknologi Sepuluh Nopember, Kampus ITS Sukolilo, Surabaya 60111, Indonesia*

INFORMASI ARTIKEL

Sejarah Artikel:
Diterima Redaksi: 18 Desember 2018
Revisi Akhir: 28 Januari 2019
Diterbitkan *Online*: 30 April 2019

KATA KUNCI

Hiperheuristik

Optimasi

Pencarian Komputasional

Simulated Annealing

HyFlex

KORESPONDENSI

Telepon: +6285274283844
E-mail: nisa.dwi.angresti@gmail.com

A B S T R A C T

Permasalahan optimasi lintas domain merupakan permasalahan optimasi yang sangat rumit karena masing-masing permasalahan mempunyai karakteristik yang berbeda. Penyelesaian terhadap permasalahan optimasi lintas domain tersebut melibatkan metode pencarian komputasional untuk memperoleh hasil yang mendekati optimal. Beberapa peneliti terdahulu mengembangkan metode hiperheutistik untuk memperoleh solusi generik yang diharapkan mampu memberikan hasil yang mendekati optimal. Hasil penelitian terdahulu mengidikasikan bahwa strategi hiperheuristik yang lebih baik diperlukan guna memperoleh solusi yang mendekati optimal untuk lintas domain permasalahan. Dalam penelitian ini, upaya untuk mendapatkan solusi generik yang mendekati optimal terhadap permasalahan optimasi lintas domain dilakukan dengan mengembangkan strategi pencarian komputasional pada tatanan High Level Heuristics (HLH) dalam mengatur proses seleksi pada rangkaian Low Level Heuristics (LLH) kemudian melakukan mekanisme penerimaan solusi. Penelitian ini menguji metode Simulated Annealing (SA) sebagai mekanisme penerimaan solusi dalam tatanan HLH agar dapat menghasilkan solusi mendekati optimal pada berbagai domain masalah optimasi yang dikombinasikan dengan metode seleksi LLH. Penelitian ini melakukan eksperimen untuk menentukan nilai parameter yang tepat untuk mengotomatiskan parameter kontrol SA dalam menyelesaikan permasalahan optimasi lintas domain. Strategi yang digunakan dalam penelitian ini diuji coba untuk menyelesaikan enam permasalahan optimasi domain yang berbeda yang diperoleh dari HyFlex, yaitu Satisfiability (SAT), Bin Packing, Flow Shop, Personnel Scheduling, Travelling Salesmen Problem (TSP), dan Vehicle Routing Problem (VRP). Dari hasil pengujian terhadap enam permasalahan optimasi tersebut, nilai parameter untuk suhu awal T adalah 100 dan faktor penurunan suhu α adalah 0,995.

1. PENDAHULUAN

Optimasi merupakan proses mencari solusi optimal [1]. Solusi optimal tersebut adalah solusi yang memenuhi batasan-batasan tertentu dan memiliki nilai tujuan untuk memaksimalkan atau meminimalkan (fungsi objektif). Saat ini permasalahan optimasi berkembang semakin kompleks karena banyaknya batasan yang harus dipenuhi dalam pencarian serta pencarian dilakukan dalam jumlah data (*instance*) yang besar. Permasalahan optimasi yang kompleks juga semakin berkembang untuk banyak bidang seperti permasalahan dalam mengoptimalkan pendjawalan ujian, rute kendaraan, dan kepuasan kerja.

Dalam menyelesaikan permasalahan optimasi yang kompleks, metode pencarian *exact* belum mampu menyelesaikan permasalahan dengan waktu yang wajar [2]. Oleh karena itu, metode *approximate* digunakan untuk menyelesaikan permasalahan tersebut. Namun, metode *approximate* belum mampu menjamin solusi paling optimal karena pencarian tidak secara deterministik. Akan tetapi, solusi yang dihasilkan dengan metode *approximate* cukup baik dengan jamiman waktu pencarian yang cepat (*polynomial*) [3]. Berkembangnya jenis permasalahan optimasi kompleks pada berbagai bidang, membutuhkan sebuah metode general untuk menyelesaikan berbagai permasalahan optimasi yang kompleks tersebut, seperti hiperheuristik.

Hiperheuristik merupakan sebuah metode *approximate* atau mekanisme pembelajaran untuk mencari atau menghasilkan heuristik sehingga dapat menyelesaikan banyak permasalahan pencarian komputasional [4]. Motivasi metode hipeheuristik adalah untuk meningkatkan generalitas metodologi pencarian. Hiperheuristik pertama kali digunakan pada tahun 1997 [5] untuk menggambarkan protokol yang menggabungkan beberapa metode Artificial Intelligence untuk otomatisasi metodologi pencarian. Metode hiperheuristik terdiri dari dua

https://doi.org/10.25077/ TEKNOSI.v5i1.2019.33-40

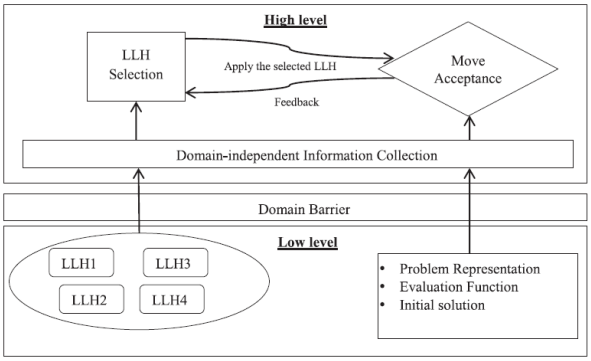
tingkatan proses pencarian, yaitu *high level heuristic* (HLH) yang melakukan pencarian pada rangkaian *low level heuristic* (LLH). Tingkatan proses ini yang menyebabkan hiperheuristik tidak bergantung pada domain permasalahan spesifik sehingga hiperheuristik dapat menyelesaikan berbagai jenis masalah optimasi. Penerapan hiperheuristik dapat menghemat waktu penyelesaian berbagai permasalahan optimasi. Namun, hiperheuristik belum mampu memberikan solusi yang optimal untuk semua permasalahan optimasi yang disebabkan oleh perbedaan karakteristik domain permasalahan. Hal tersebut menjadi tantangan hiperheuristik untuk mengotomatisasi desain dan mengatur metode hiperheuristik dalam memecahkan permasalahan pencarian komputasi yang kompleks pada domain permasalahan optimasi berbeda. Hiperheuristik harus mengembangkan strategi HLH yang lebih umum untuk diterapkan pada banyak permasalahan. Perlu adanya eksperimen agar hiperheuristik dapat bekerja baik (menghasilkan solusi optimal) untuk banyak domain permasalahan. Eksperimen tersebut menguji coba kombinasi metode sebagai strategi HLH untuk menyeleksi LLH dalam mengoptimalkan solusi pada domain permasalahan berbeda.

Penelitian terdahulu telah melakukan eksperimen kombinasi metode SA yang digunakan di dalam hiperheuristik. Penelitian [6] [7] [8] telah mengusulkan algoritma hiperheuristik yang didasarkan pada metode SA dalam menyelesaikan permasalahan optimasi dengan ukuran permasalahan dan rasio ruang berbeda. Namun, penelitian tersebut hanya menguji metode SA pada beberapa permasalahan optimasi sejenis. Hiperheuristik bertujuan untuk menggeneralkan proses pencarian untuk banyak domain permasalahan. Oleh karena itu, perlu dilakukan eksperimen untuk mengotomatisasikan paramater kontrol metode SA dalam strategi HLH sehingga dapat meningkatkan generalitas hiperheuristik.

HLH terdiri atas dua mekanisme utama, yaitu mekanisme seleksi LLH dan penerimaan solusi. Kombinasi yang tepat dari kedua mekanisme merupakan keberasilan hiperheuristik. Kombinasi dari kedua mekanisme merupakan metodologi yang sangat adaptif [4]. Penelitian ini melakukan eksperimen untuk mengkombinasikan metode seleksi LLH dan penerimaan solusi sebagai strategi HLH dengan cara mengadaptasi metode SA. Metode SA digunakan sebagai mekanisme penerimaan solusi yang dikombinasikan dengan metode Simple Random sebagai metode seleksi LLH. Dalam penelitian ini eksperimen dilakukan untuk menentukan nilai parameter yang tepat untuk mengotomatiskan parameter kontrol dalam menyelesaikan permasalahan optimasi lintas domain.

1.1. Hiperheuristik

Hiperheuristik telah ada sejak tahun 1960-an yang ditemukan dalam bidang Operational Research, Computer Science, dan Artificial Intelligence. Namun, istilah hiperheuristik baru diperkenalkan tahun 2000 [4]. Hiperheuristik menggambarkan heuristik untuk menyeleksi heuristik atau heuristik untuk menghasilkan heuristik. Ada dua hal utama yang dapat dikembangkan pada hiperheuristik yaitu seleksi hiperheuristik dan generasi hiperheuristik. Kerangka seleksi hiperheuristik dapat dilihat pada Gambar 1.



Gambar 1. Kerangka Seleksi Hiperheuristik
(Sumber : [10])

Dalam Gambar 1 dapat dilihat hiperheuristik terdiri dari dua level pencarian, yaitu HLH dan rangkaian LLH (LLH1, LLH2, LLH3, ..., LLHn). Hiperheuristik sebagai HLH melakukan pencarian LLH yang tepat untuk masing-masing domain permasalahan. LLH merupakan heuristik yang dihasillkan selama proses pencarian oleh hiperheuristik.

Seleksi hiperheuristik merupakan pendekatan yang membangun sebuah solusi secara bertahap, dimulai dengan inisial solusi, kemudian secara cerdas memilih dan menggunakan heuristik konstruktif untuk membangun solusi yang lengkap.

Keberhasilan kerangka seleksi hiperheuristik ditentukan oleh strategi HLH dalam menyeleksi LLH dan menerima solusi yang dihasilkan oleh LLH. Seleksi LLH dilakukan untuk memilih LLH yang sesuai dalam menemukan solusi, sementara penerimaan solusi digunakan untuk memilih atau menolak solusi yang dihasilkan oleh LLH. Banyak kombinasi seleksi LLH dan penerimaan solusi telah dikembangkan oleh penelitian sebelumnya [10]. Beberapa metode dalam seleksi LLH adalah Simple Random, Choice Function, Tabu Search, Harmony Search, Reinforcement Learning, Monte Carlo Tree Search, Variable Neighborhood Search, dan Multi-Armed Bandit.

Metode penerimaan solusi berfungsi untuk menyeimbangkan pencarian agar tidak terjebak oleh *local optima*, sehingga metode penerimaan solusi yang baik adalah yang tidak hanya menerima hasil yang baik, tetapi juga hasil yang buruk. Penelitian sebelumnya telah banyak mengembangkan metode penerimaan solusi, seperti Only Improvement, All Moves, Late Acceptance, Hill Climbing, dan SA.

Penelitian ini mengkombinasikan metode Simple Random dan SA sebagai staregi HLH. Metode Simple Random melakukan pencarian *low level heuristic* secara acak sampai batas waktu yang ditentukan berakhir. Metode SA menerima solusi baru yang memiliki kualitas yang lebih baik atau sama dibandingkan dengan solusi yang dari iterasi n sebelumnya, di mana n adalah parameter positif [6]. Pada SA, setiap iterasi yang menghasilkan solusi dengan kualitas yang

 https://doi.org/10.25077/ TEKNOSI.v5i1.2019.33-40

sama atau lebih baik dari solusi pada langkah sebelumnya akan diterima. Jika suatu pemindahan menghasilkan solusi kualitas yang lebih buruk, pemindahan itu diterima secara probabilistik berdasarkan pada seberapa jauh solusinya lebih buruk (parameter dipengaruhi waktu). Tingkat solusi buruk yang dapat diterima akan menurun karena waktu berkurang/habis dan kemungkinan pindah ke solusi yang lebih buruk akan berkurang seiring waktu.

1.2. Simulated Annealing

Simulated Annealing (SA) memiliki cara kerja seperti poses penguatan (*annealing*) fisika, yaitu benda logam dipanaskan pada suhu tinggi sehingga molekul logam akan pecah dan meleleh, kemudian secara perlahan didinginkan dan molekul logam saling mengikat kembali (mengeras). Pada suhu tinggi, program akan lebih mudah berubah bentuk, sehingga dapat menerima solusi yang lebih banyak yang tidak berkaitan dengan solusi yang ada. Suhu yang tinggi ini untuk mendukung eksplorasi. Pada saat suhu rendah, program hanya berfokus pada solusi yang ada dan mencari nilai fungsi minimum dari solusi tersebut. Hal ini untuk mendukung eksploitasi pencarian.

Metode SA selalu menerima perubahan yang lebih baik, serta dapat menerima perubahan yang lebih buruk dengan ketentuan kriteria Metropolis atau bergantung pada probabilitas [11]. Pada suhu T, probabilitas penerimaan solusi lebih buruk dihitung dengan rumus seperti persamaan 1.

$$p = e^{\frac{\Delta f}{T}} > r \quad (1)$$

Penurunan suhu dilakukan setiap iterasi. Iterasi pada setiap T, walaupun secara komputasi mahal, harus mempertimbangkan jumlah iterasi yang cukup besar dengan sedikit T, atau jumlah iterasi kecil dengan banyak T, atau keseimbangan antara keduanya. Penuruhan suhu setiap iterasi dihitung seperti rumus yang terdapat pada persamaan 2.

$$T = T * \alpha \quad (2)$$

1.3. Domain Permasalahan

Domain permasalahan merupakan objek permasalahan yang akan diselesaikan menggunakan metode yang dikembangkan. Setiap domain permasalahan optimasi memiliki:

1. *Instance* berupa data graph yang menjadi data input untuk sebuah permasalahan optimasi.
2. Batasan (*constrain*) untuk menentukan solusi layak (memenuhi basatan) atau tidak.
3. Fungsi objektif merupakan tujuan optimasi untuk menentukan kualitas solusi optimal (minimum atau maksimum) yang akan ditingkatan selama pencarian.

Domain permasalahan yang diuji cobakan dalam penelitian ini merupakan domain yang terdapat pada *framework* HyFlex. HyFlex *(Hyper-heuristic Flexible*) merupakan kerangka kerja yang dirancang menggunakan bahasa pemograman Java untuk pengembangan, pengujian, dan perbandingan algoritma pencarian hiperheuristik [9]. Domain permasalahan pada HyFlex memiliki fungsi objektif untuk meminimalkan, sehingga solusi yang memiliki nilai objektif terkecil merupakan solusi yang optimal. Terdapat enam domain pada HyFlex, yaitu sebagai berikut:

1. Satisfiability (SAT) adalah permasalahan optimasi untuk mengoptimalkan kepuasan seseorang terhadap keinginan-keinginan yang akan dicapai [12].
2. Bin Packing (BP) adalah permasalahan optimasi untuk megoptimalkan ruang bin agar semakin banyak barang (item) yang dapat dimasukkan ke dalam bin. [13].
3. Flow Shop (FS) adalah permasalahan optimasi untuk mengoptimalkan waktu penyelesaian pekerjaan oleh mesin [14].
4. Personnel Schedulling (PS) adalah domain permasalahan optimasi untuk mengoptimalkan jam kerja karyaran sehingga tidak ada karywan yang mengalami kelebihan jam kerja[15].
5. Travelling Salesman Problem (TSP) adalah permasalahan optimasi untuk mengoptimalkan jarak tempat yang harus dikunjungi seseorang atau *salesman* [16].
6. Vehicle Routing Problem (VRP) adalah permasalahan optimasi untuk mengoptimalkan rute yang harus dikunjungi oleh kendaraan[16].

Eksperimen ini bertujuan untuk membuktikan metode Simulated Annealing dapat menyelesaikan permasalahan optimasi lintas domain. Eksperimen diuji cobakan terhadap enam permasalahan optimasi kombinatorial yang berbeda, yaitu domain permasalahan satisfiability (SAT), bin packing (BP), flow shop (FS), personnel schedulling (PS), Traveling Salesman Problem (TSP), dan Vehicle Routing Problem (VRP). Enam domain permasalahan yang berbeda tersebut digunakan untuk melihat generalitas metode yang diusulkan dapat menghasilkan solusi optimal. Enam permasalahan optimasi tersebut merupakan permasalahan optimasi yang terdapat di dalam *framework* HyFlex yang memiliki *instance* data yang nyata [9]. Untuk melakukan pengujian, dilakukan perbandingan dengan eksperimen yang sama.

2. METODE

Tahapan yang dilalui dalam penelitian ini terdiri dari identifikasi masalah, studi literatur, penentuan data, desain algoritma, impelementasi, dan pengujian metode hiperheuristik.

2.1. Identifikasi Masalah

Dalam tahap ini dilakukan identifikasi masalah yang akan dalam penelitian ini, seperti yang telah dijelaskan pada bagian pendahuluan.

2.2. Studi Literatur

Dalam tahap studi literatur ini dibahas literatur penelitian hiperheuristik yang digunakan dalam penelitian ini seperti yang telah dijelaskan dalam tinjauan pustaka.

2.3. Penentuan Data

Tahap ini menetapkan data yang digunakan di dalam penelitian. Data yang digunakan adalah data domain permasalahan optimasi

untuk pengujian dari pengembangan metode hiperheuristik. Domain permasalahan optimasi yang digunakan adalah enam domain permasalahan yang terdapat dalam *framework* HyFlex, yaitu domain SAT, bin packing, flow shop, personnel schedulling, TSP, dan VRP.

2.4. Desain Algoritma

Dalam tahap ini dilakukan perancangan metode hiperheurstik sebagai strategi HLH. Dalam strategi HLH penelitian ini, metode untuk mekanisme seleksi LLH dikombinasikan dengan metode untuk mekanisme penerimaan solusi, yaitu metode Simple Random dengan metode SA. Strategi HLH yang digunakan dapat dilihat seperti Gambar 2.

```
1   Find initial solution s;
2   Set initial T, α;
3   Set T← T0;
4   Get functionValueCurrent = f(s)
5   while !hasTimeExpired() do
6   select random LLHx from (LLH1, LLH2,
    LLHn), LLHx generate s'
7   Get functionValueNew = f(s')
8   Δfunction = f(s') – f(s)
9   if Δfunction < 0 then
10     s ← s'
11  else if (rand(0,1) < e^(-Δfunciton/T)) then
12     s ← s'
13  f(s)= f(s')
    end if
    end if
14  after every iteration set T = T * α
15  end while
```

Gambar 2. Pseudocode Simulated Annealing

Dalam Gambar 2 dijelaskan proses di dalam strategi HLH, yaitu:

1. Pencarian dimulai dengan inisiasi solusi atau proses pencarian kandidat solusi. Inisiasi solusi tersebut ditingkatkan selama pencarian.
2. Pendefnisian parameter SA yaitu suhu awal T dan faktor penurunan suhu α.
3. Pengaturan nilai suhu awal T. Nilai suhu awal diatur tinggi agar dapat mengeksplorasi solusi lebih banyak.
4. Pembuatan variabel untuk mendapatkan fungsi objektif dari inisiasi solusi.
5. Pengaturan batasan pencarian. Dalam hal ini, batasan pencarian menggunakan batasan waktu sehingga iterasi pencarian selesai sampai batas yang telah ditentukan.
6. Proses mekanisme seleksi LLH secara acak. LLH dipilih secara acak dari rangkaian LLH dari setiap domain permasalahan. LLH yang terpilih akan menghasilkan solusi baru.
7. Perhitungan nilai fungsi objektif dari solusi baru.
8. Perhitungan perubahan fungsi objektif.
9. Proses mekanisme penerimaan solusi.
10. Jika perubahan fungsi lebih besar dari 0 (meningkat) maka solusi diterima. Solusi baru menggantikan solusi inisiasi.
11. Jika tidak, maka hitung probabilitas penerimaan. Jika nilai random (0 atau 1) lebih besar dari nilai probabilitas, maka solusi diterima.
12. Solusi baru menggantikan solusi inisiasi.
13. Fungsi objekstif solusi inisiasi sama dengan fungsi obektif solusi baru.
14. Nilai fungsi objektif baru akan menggantikan nilai fungsi objektif lama (*incumbent*), untuk dilakukan pencarian solusi berikutnya.
15. Nilai suhu awal T akan dikurangi setiap iterasi.

Nilai parameter T dan α yang diuji coba adalah (1) T adalah 100 dan α adalah 0,85 [6]; (2) T adalah 100 dan α adalah 0,995 [11]. Nilai parameter digunakan pada proses ke-3 dan ke-4 dalam Gambar 2.

Dalam hiperheuristik juga terdapat rangkaian LLH yang diseleksi oleh strategi HLH pada setiap domain permasalahan. Penelitian ini menggunakan LLH yang terdapat pada setiap domain dalam HyFlex. Rangkaian LLH dalam HyFlex dikelompokkan menjadi empat kelompok heuristik, yaitu mutasi, ruin-recreate (RR), local search (LS), dan crossover.

Kelompok heuristik mutasi melakukan perubahan secara acak pada solusi dengan cara menukar, mengubah, membuang, menambah, atau menghapus komponen solusi. Pada kelompok RR perubahan pada solusi dilakukan dengan membongkar sebagian solusi dan membangun atau membuatnya kembali. Kelompok heuristik LS secara iteratif membuat perubahan kecil pada solusi, dan hanya menerima solusi yang meningkat sampai optimum lokal ditemukan atau kondisi berhenti terpenuhi. Heuristik ini berbeda dari kelompok heuristik mutasi karena heuristik menggabungkan proses perbaikan berulang dan mereka menjamin bahwa solusi yang baik akan dihasilkan. Pada kelompok heuristik *crossover* melakukan perubahan dengan mengambil dua solusi, menggabungkannya dan mengembalikan solusi baru. Tabel 1 menjelaskan jumlah LLH pada setiap domain.

Tabel 1. Jumlah LLH setiap Domain

No	Dom	Mut	RR	Cross over	LS	Total
1	SAT	6	1	2	2	11
2	BP	3	2	1	2	8
3	FS	5	2	3	4	15
4	PS	1	3	3	4	12
5	TSP	5	1	3	2	13
6	VRP	3	2	2	3	10

(Sumber: [9])

2.5. Implementasi

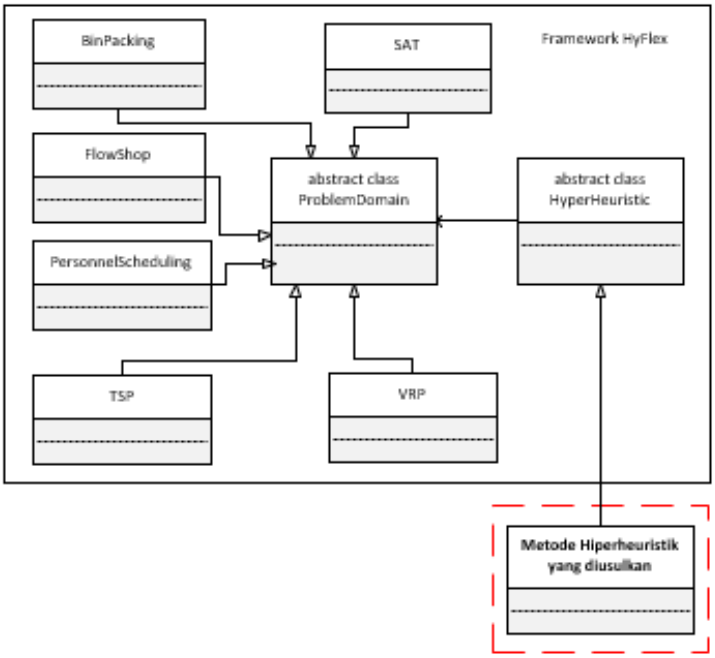
Implementasi desain strategi HLH dalam hiperheuristik dilakukan pada komputer dengan prosesor Intel Pentium N3710 1.6 GHz dan memori 4096MB. Hasil desain algoritma diimplementasikan pada *framework* HyFLex [9] menggunakan *tools* NetBeans IDE 8.2. Implementasi pada HyFlex dilakukan dengan cara memanggil

 https://doi.org/10.25077/ TEKNOSI.v5i1.2019.33-40

fungsi-fungsi (*methods*) dan *libraries* yang ada pada HyFlex. *Libraries* yang digunakan adalah *chesc.jar.*

Untuk mengimplementasikan metode hiperheuristik pada HyFlex, maka harus dibuat kelas baru turunan (*extend*) dari kelas abstrak *HyperHeuristic*. Hubungan kelas baru dengan kelas abstrak *HyperHeuristic* dalam *framework* HyFlex dapat dilihat pada Gambar 3.

Gambar 3 menjelaskan hubungan kelas baru dengan kelas abstrak *HyperHeuristic* yang telah ada adalah *inheritance*, dimana kelas baru merupakan kelas turunan dari kelas *HyperHeuristic*. Semua atribut dan *method* yang bertipe *public* dan *protect* pada kelas *HyperHeuristic* dapat diakses oleh kelas turunannya. Hubungan kelas abstrak *HyperHeuristic* dan *ProblemDomain* adalah asosiasi, dimana *methods* kelas *ProblemDomain* digunakan oleh kelas *HyperHeuristic*.



Gambar 3. Diagram Kelas Pengembangan pada HyFlex

Pengimplementasian metode hiperheuristik pada HyFlex menggunakan beberapa pamarameter yang harus ditentukan. Pengaturan parameter tersebut dapat dilihat pada Tabel 2.

Tabel 2. Parameter yang Digunakan

Variabel	Deskripsi	Nilai	Ref
initialiseSolution	Paramater untuk indeks array solusi pada memori	0	[9]
loadInstance	Parameter indeks *instance* domain permasalahan yang digunakan	0-4	[9]
setTimeLimit	parameter kondisi berhenti untuk membatasi banyak pencarian dengan batasan waktu.	60000 (milidetik)	[17]

2.6. Pengujian

Strategi HLH dalam metode hiperheuristik yang telah diimplementasikan dalam HyFlex, diuji coba pada enam domain permasalahan optimasi. Setiap domain memiliki lima *instance* dataset, sehingga secara keseluruhan terdapat 30 *instance* data yang diuji coba. Uji coba dilakukan untuk menentukan parameter T dan α yang tepat pada metode SA yang digunakan. Setiap *instance* diuji coba dengan nilai parameter yang sama sebanyak 11 kali eksekusi.

3. HASIL DAN PEMBAHASAN

Penerapan metode SA yang diuji sebanyak 11 kali eksekusi menghasilkan kumpulan data untuk dianalisis. Hasil dari uji coba metode SA tersebut menghasilkan kumpulan data nilai fungsi objektif dari masing-masng domain yang akan dianalisis. Dari kumpulan data tersebut, dihitung nilai minimum, median, maksimum dan nilai rata-rata.

Tabel 3. Hasil Uji Coba 1

Domain	Ins	Min	Med	Maks	Rata-rata
SAT	0	18	22	23	21,27
	1	34	34	34	34
	2	32	34	34	33,09
	3	32	32	32	32
	4	40	40	40	40
BinPacking	0	0,020	0,021	0,022	0,021
	1	0,013	0,021	0,021	0,019
	2	0,029	0,029	0,029	0,029
	3	0,029	0,029	0,029	0,029
	4	0,005	0,005	0,005	0,005
FlowShop	0	6417	6417	6433	6423
	1	6393	6408	6408	6407
	2	6457	6457	6515	6473
	3	6366	6366	6397	6374
	4	6366	6438	6438	6431
Personnel Scheduling	0	3375	3424	3549	3436,09
	1	2525	4135	4455	4027,18
	2	425	535	1905	1093,18
	3	24	34	42	33,91
	4	41	45	52	45,73
TSP	0	51426,8	51426,8	51426,8	51426,8
	1	115484,4	115484,4	115484,4	115484,4
	2	7066,5	7066,5	7066,5	7066,5
	3	44428,9	44428,9	44428,9	44428,9
	4	9418,6	9418,6	9418,6	9418,6
VRP	0	7169,3	7169,3	7169,3	7169,3
	1	22690,7	22690,7	22690,7	22690,7
	2	14458,8	14458,8	14458,8	14458,8
	3	10330,4	10330,4	10330,4	10330,4
	4	15358,4	15358,4	15358,4	15358,4

Tabel 4. Hasil Uji Coba 2

Domain	Ins	Min	Med	Maks	Rata-rata
SAT	0	13	13	15	13,55
	1	34	34	34	34
	2	22	25	26	23,82
	3	22	22	22	22
	4	26	26	26	26
BinPacking	0	0,021	0,031	0,031	0,026
	1	0,031	0,054	0,063	0,046
	2	0,028	0,028	0,028	0,028
	3	0,026	0,026	0,026	0,026
	4	0,006	0,006	0,006	0,006
FlowShop	0	6415	6415	6415	6415
	1	6390	6390	6390	6390
	2	6435	6435	6435	6435
	3	6420	6420	6420	6420
	4	6508	6508	6508	6508
Personnel Scheduling	0	3414	3464	3504	3457,8
	1	2830	4240	6025	4283,6
	2	430	606	3160	1163,4
	3	23	38	51	37,2
	4	36	52	67	50,4
TSP	0	51426,8	51426,8	51426,8	51426,8
	1	115484,4	115484,4	118075,8	115720
	2	7066,5	7066,5	7066,5	7066,5
	3	44428,9	44428,9	44428,9	44428,9
	4	9453,0	9453,0	9453,0	9453
VRP	0	7117,7	7117,7	7117,7	7117,7
	1	21647,9	21647,9	21647,9	21647,9
	2	14433,9	14433,9	14433,9	14433,9
	3	7283,3	7283,3	7283,3	7283,3

 https://doi.org/10.25077/ TEKNOSI.v5i1.2019.33-40

	4	14328,5	14328,5	14328,5	14328,5

Hasil dari uji coba setiap parameter yang diuji cobakan dalam metoda SA dapat dilihat pada tabel 3 dan tabel 4. Tabel 3 merupakan hasil uji coba untuk parameter kontrol T adalah 100 dan αdalah 0,85. Tabel 4 merupakan hasil uji coba untuk parameter kontrol T adalah 100 dan α adalah 0,995.

Dalam memilih paramater kontrol yang generik dilakukan pengujian peforma setiap paramater yang diusulkan dalam mencari solusi optimal. Solusi optimal tidak dapat diukur nilai optimalitasnya karena metode ini adalah metode apporoximate. Namun, untuk memilih metode yang terbaik dalam mencari solusi optimal, perbandingan hasil uji coba dengan metode lainnya [3]. Disini, perbandingan dilakukan dengan cara membandingkan hasil uji coba parameter yang dilakukan terhadap domain yang sama. Nilai median dan nilai minimum hasil eksekusi nilai fungsi objektif menjadi nilai perbandingan yang akan diuji [5].

$$\text{Instance Terbaik } (\%) = ((\text{inst min}) / (\text{total instance}) \times 100) \quad (3)$$

Perbandingan nilai median dan nilai mnimum untuk melihat metode yang lebih banyak menghasilkan solusi optimal. Metode dikatakan terbaik, jika nilai fungsi objektif lebih minimal. Instance yang lebih minimal dijumlahkan pada seluruh instance domain.. Banyak instance yang lebih baik dihitung persentase dengan cara seperti persamaan 3. Hasil Untuk menghitung nilai peningkatan dengan cara seperti pada persamaan 3. Perbandingan nilai median dan nilai medium dapat dilihat pada Tabel 5 dan Tabel 6

Tabel 5. Perbandingan Nilai Median

Domain	Ins	Nilai Median	
		Uji Coba 1	Uji Coba 2
SAT	0	22	13
	1	34	34
	2	34	25
	3	32	22
	4	40	26
BinPacking	0	0,021	0,031
	1	0,021	0,054
	2	0,029	0,028
	3	0,029	0,026
	4	0,005	0,006
FlowShop	0	6417	6415
	1	6408	6390
	2	6457	6435
	3	6366	6420
	4	6438	6508
Personnel Scheduling	0	3424	3464
	1	4135	4240
	2	535	606
	3	34	38
	4	45	52
TSP	0	51426,8	51426,8
	1	115484,4	115484,4
	2	7066,5	7066,5
	3	44428,9	44428,9
	4	9418,6	9453,0
VRP	0	7169,3	7117,7
	1	22690,7	21647,9
	2	14458,8	14433,9
	3	10330,4	7283,3
	4	15358,4	14328,5
Ins Terbaik		17	19
Persentase		57%	63%

Perbandingan nilai median dan nilai mnimum untuk melihat metode yang lebih banyak menghasilkan solusi optimal. Metode dikatakan terbaik, jika nilai fungsi objektif lebih minimal. Instance yang lebih minimal dijumlahkan pada seluruh instance domain.. Banyak instance yang lebih baik dihitung persentase dengan cara seperti persamaan 3. Hasil Untuk menghitung nilai peningkatan dengan cara seperti pada persamaan 3. Perbandingan nilai median dan nilai medium dapat dilihat pada Tabel 5 dan Tabel 6. Warna kuning pada Tabel 5 dan Tabel 6 menjelaskan peforma yang lebih baik pada setiap *instance*. Fungsi objektif yang memiliki nilai sama akan diberikan warna yang sama.

Dari hasil perhitungan pada perbandingan nilai median dan nilai minimum setiap *instance*, dapat dilihat penerapan metode Simulated Annealing dengan nilai parameter kontrol dalam uji coba 2 dengan T adalah 100 dan α adalah 0,995 lebih dapat meningkatkan peforma pencarian dalam menemukan solusi optimal pada 19 *instance* dari 30 *instance* dengan persentase 63% dan 22 instance dengan persentase 73%. Secara keseluruhan, perbedaan uji coba 1 dan uji coba 2 tidak terlalu jauh. Namun, pada beberapa *instance* mengalami peningkatan secara signifikan. Metode Simulated Annealing uji coba 2 lebih ebih baik daripada uji coba 1.

Tabel 6. Perbandingan Nilai Minimum

Domain	Ins	Nilai Minimum	
		Uji Coba 1	Uji Coba 2
SAT	0	18	13
	1	34	34
	2	32	22
	3	32	22
	4	40	26
BinPacking	0	0,020	0,021
	1	0,013	0,031
	2	0,029	0,028
	3	0,029	0,026
	4	0,005	0,006
FlowShop	0	6417	6415
	1	6393	6390
	2	6457	6435
	3	6366	6420
	4	6366	6508
Personnel Scheduling	0	3375	3414
	1	2525	2830
	2	425	430
	3	24	23
	4	41	36
TSP	0	51426,8	51426,8
	1	115484,4	115484,4
	2	7066,5	7066,5
	3	44428,9	44428,9
	4	9418,6	9453,0
VRP	0	7169,3	7117,7
	1	22690,7	21647,9
	2	14458,8	14433,9
	3	10330,4	7283,3
	4	15358,4	14328,5
Ins Terbaik		10	22
Persentase		33%	73%

4. KESIMPULAN

Berdasarkan hasil pengujian yang dilakukan pada bab sebelumnya, maka dapat disimpulkan nilai parameter yang digunakan metode SA untuk mengotomatiskan parameter kontrol dalam menyelesaikan masalah optimasi lintas domain, adalah nilai suhu awal T adalah 100 dan α adalah 0,995. Penetapan parameter dilakukan terhadap 11 kali eksekusi pada setiap instance domain permasalahan. Hasil eksekusi parameter yang berbeda dibandingkan dan diberi skor plus satu untuk instance yang meningkat (nilai fungsi objektif lebih minimal).

Untuk penelitian berikutnya, sebaiknya metode untuk mekanisme seleksi low level heuristic Simple Random diuji coba dengan metode lainnya yang dikominasikan dengan metode Simulated Annealing agar dapat mencari solusi yang lebih baik (optimal) pada permasalahan optimasi lintas domain.

DAFTAR PUSTAKA

[1] G. Lindfield dan J. Penny, "An Introduction to Optimization," in *Introduction to Nature-Inspired Optimization*, Elsevier, 2017, pp. 1–18.

[2] M. Gendreau dan J.-Y. Potvin, Eds., *Handbook of metaheuristics*, 2. ed. New York, NY: Springer, 2010.

[3] E. K. Burke, *Search methodologies*. New York: Springer, 2013.

[4] E. K. Burke *et al.*, "Hyper-heuristics: a survey of the state of the art," *J. Oper. Res. Soc.*, vol. 64, no. 12, pp. 1695–1724, Dec. 2013.

[5] E. K. Burke, M. Hyde, G. Kendall, G. Ochoa, E. Özcan, and J. R. Woodward, "A classification of hyper-heuristic approaches," in *Handbook of metaheuristics*, Springer, 2010, pp. 449–468.

[6] R. Bai dan G. Kendall, "An Investigation of Automated Planograms Using a Simulated Annealing Based Hyper-Heuristic," in *Metaheuristics: Progress as Real Problem Solvers*, vol. 32, T. Ibaraki, K. Nonobe, and M. Yagiura, Eds. New York: Springer-Verlag, 2005, pp. 87–108.

[7] B. Bilgin, E. Özcan, dan E. E. Korkmaz, "An Experimental Study on Hyper-heuristics and Exam Timetabling," in *Practice and Theory of Automated Timetabling VI*, vol. 3867, E. K. Burke and H. Rudová, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 394–412.

[8] K. A. Dowsland, E. Soubeiga, dan E. Burke, "A simulated annealing based hyperheuristic for determining shipper sizes for storage and transportation," *Eur. J. Oper. Res.*, vol. 179, no. 3, pp. 759–774, Jun. 2007.

[9] G. Ochoa *et al.*, "Hyflex: A benchmark framework for cross-domain heuristic search," in *European Conference on Evolutionary Computation in Combinatorial Optimization*, 2012, pp. 136–147.

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

[11] R. Qu, "Meta-heurisitic Agorithm." Apr-2016.

[12] M. Hyde, G. Ochoa, J. A. Vazquez-Rodriguez, and T. Curtois, "A HyFlex Module for the MAX-SAT Problem," p. 4.

[13] M. Hyde, G. Ochoa, J. A. Vazquez-Rodriguez, and T. Curtois, "A HyFlex Module for the One Dimensional Bin Packing Problem," p. 5.

[14] J. A. Vazquez-Rodrıguez, G. Ochoa, T. Curtois, and M. Hyde, "A HyFlex Module for the Permutation Flow Shop Problem," p. 4.

[15] T. Curtois, G. Ochoa, M. Hyde, and J. A. Vázquez-Rodríguez, "A HyFlex Module for the Personnel Scheduling Problem," p. 12.

[16] A. Lehrbaum, "A New Hyperheuristic Algorithm for Cross-Domain Search Problems," in *Learning and Intelligent Optimization*, vol. 7219, Y. Hamadi and M. Schoenauer, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 437–442.

[17] E. K. Burke *et al.*, "The cross-domain heuristic search challenge–an international research competition," in *International Conference on Learning and Intelligent Optimization*, 2011, pp. 631–634.

NOMENKLATUR

p	: probabilitas penerimaan solusi lebih buruk
Δf	: perubahan nilai fungsi
r	: angka random antara 0 dan 1
T	: suhu saat ini
e	: fungsi eksopenensial
α	: faktor penurunan suhu

BIODATA PENULIS

Nisa Dwi Angresti, S.SI.
Penulis merupakan mahasiswa pascasarjana Institut Teknologi Sepuluh Nopember program magister Sistem Informasi. Konsentrasi keilmuan adalah bidang optimasi.

Prof. Ir. Arif Djunaidy, M.Sc., Ph.D.
Penulis merupakan guru besar di Institut Teknologi Sepuluh Nopember yang merupakan lulusan computer science, University of Manchester dengan bidang keilmuan penggalian data dan analitika bisnis.

Ahmad Mukhlason, S.Kom., M.Sc., Ph.D.
Penulis merupakan dosen Departemen Sistem Informasi Institut Teknologi Sepuluh Nopember. Penulis lulusan Information Technology, School of Computer Science, The University of Nottingham, UK

https://doi.org/10.25077/ TEKNOSI.v5i1.2019.33-40