Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK) DOI: 10.25126/jtiik2023106842
Vol. 10, No. 4, Agustus 2023, hlm. 893-898 p-ISSN: 2355-7699
Akreditasi KEMENRISTEKDIKTI, No. 36/E/KPT/2019 e-ISSN: 2528-6579

OPTIMASI RUTE RENCANA PERJALANAN PESAWAT MENGGUNAKAN ALGORITMA LATE ACCEPTANCE HILL CLIMBING (STUDI KASUS : TRAVELLING SALESMAN CHALLENGE 2.0)

Ahmad Muklason[*1], I Gusti Agung Premananda[2]

[1,2]Institut Teknologi Sepuluh Nopember, Surabaya
Email: [1]mukhlason@is.its.ac.id, [2]igustiagungpremananda@gmail.com
[*]Penulis Korespondensi

(Naskah masuk: 26 Desember 2022, diterima untuk diterbitkan: 26 Juli 2023)

Abstrak

Permasalahan Traveling Salesman Problem (TSP) merupakan permasalahan klasik yang popular diteliti dalam bidang optimasi kombinatorika. Permasalahan ini bertujuan menentukan rute perjalanan terpendek untuk mengunjungi setiap lokasi tepat satu kali dan diakhir perjalanan harus kembali ke lokasi awal perjalanan dimulai. Permasalahan ini telah digolongkan sebagai permasalahan NP-Hard, sehingga membutuhkan algoritma *non-deterministic* untuk dapat menyelesaikan permasalahan ini. Dalam permasalahan nyata, salah satu penerapan TSP ada pada permasalahan untuk menentukan rute perjalanan termurah untuk mengunjungi beberapa kota di beberapa negara. Kompetisi Travelling Salesman Challenge 2.0 (TSC 2.0) mengangkat permasalahan ini dalam sebuah kompetisi pada tahun 2018. Untuk menyelesaikan studi kasus tersebut, penelitian ini menyembangkan algoritma Late Acceptance Hill Climbing (LAHC) menggunakan metode hiper-heuristik. Algoritma LAHC merupakan algoritma yang sederhana namun telah terbukti mampu mengoptimasi dengan baik pada beberapa permasalahan TSP. Algoritma LAHC diuji coba pada 14 dataset dari TSC 2.0. Hasil penelitian menunjukan algoritma LAHC menghasilkan solusi yang kompetitif dengan mampu menurunkan biaya perjalanan dengan rata-rata 58% dan menghasilkan hasil yang lebih baik dengan rata-rata 9% dari algoritma Threshold Acceptance (TA) yang digunakan sebagai algoritma pembanding.

Kata kunci: *traveling salesman problem, traveling salesman challenge 2.0, late acceptance hill climbing, hiper-heuristik*

OPTIMIZATION OF AIRCRAFT TRAVEL PLANNING ROUTES USING THE LATE ACCEPTANCE HILL CLIMBING ALGORITHM (CASE STUDY: TRAVELLING SALESMAN CHALLENGE 2.0)

Abstract

The Traveling Salesman Problem (TSP) is a classic problem that is popularly researched in the field of combinatorics optimization. This problem aims to determine the shortest travel route to visit each location exactly once and, at the end of the trip, must return to where the trip started. This problem has been classified as an NP-Hard problem. Therefore it requires a non-deterministic algorithm to solve it. In the real world, one of the applications of TSP is the problem of determining the cheapest travel routes to visit several cities in several countries. The Traveling Salesman Challenge 2.0 (TSC 2.0) competition raised this issue in a competition in 2018. This study developed the Late Acceptance Hill Climbing (LAHC) algorithm using the hyper-heuristic method to complete the case study from TSC 2.0. The LAHC algorithm is simple but has been proven to optimize well for several TSP problems. The LAHC algorithm was tested on 14 datasets from TSC 2.0. The results show that the LAHC algorithm produces competitive solutions by reducing travel costs by an average of 58% and making better results by an average of 9% than the Threshold Acceptance (TA) algorithm used as a comparison algorithm.

Keywords: *traveling salesman problem, traveling salesman challenge 2.0, late acceptance hill climbing, hyper-heuristic*

1. PENDAHULUAN

TSP merupakan permasalahan klasik yang popular dalam penelitian di bidang optimasi kombinatorika (Pandiri and Singh, 2019). Permasalahan ini merupakan permasalahan optimasi diskrit yang bertujuan untuk menentukan rute dari sesorang sales untuk mengunjungi beberapa tempat tepat satu kali dan diakhir perjalanan akan kembali ke

tempat dimana perjalanan dimulai (Akhand et al., 2020). Tujuan dari permasalahan ini umumnya untuk meminimalkan jarak atau biaya dalam perjalanan (Al-Furhud and Hussain, 2020). Permasalahan TSP telah banyak diadaptasi pada permasalahan nyata seperti pada permasalahan transportasi dan pengiriman barang (Cheikhrouhou and Khoufi, 2021).

Permasalahan TSP telah diklasifikasikan sebagai permasalahan NP-Hard, yang berarti pertumbuhan jumlah permasalahan berbanding eksponensial terhadap kombinasi solusi yang dihasilkan (Cinar, Korkmaz and Kiran, 2020). Sebagai contoh, permasalahan TSP dengan 5 kota yang harus dikunjungi akan menghasilkan 120 kombinasi solusi. Jika permasalahan dinaikan dua kali menjadi 10 kota, kombinasi yang dihasilkan meningkat secara eksponensial yaitu menjadi 3.628.800. Jika permasalahan ditingkatkan menjadi 15 kota, maka akan menghasilkan 1,3 triliun kombinasi solusi. Hal ini menyebabkan hanya algoritma *non-deterministic* yang mampu menyelesaikan permasalahan ini dalam waktu yang wajar (Premananda, Tjahyanto and Muklason, 2022).

Salah satu algoritma *non-deterministic* yang telah banyak dikembangkan yaitu algoritma LAHC. Algoritma ini dikembangkan pertama kali dengan menguji coba pada dua permasalahan optimasi kombinatorika yaitu permasalahan TSP dan permasalahan penjadwalan (Burke and Bykov, 2017). Hasilnya menunjukan kompetitifnya algoritma ini dibandingkan dengan beberapa algoritma lain seperti Simulated Annealing, TA dan Great Deluge. Penelitian lain juga telah dilakukan oleh (Bolaji, Bamigbola and Shola, 2018) pada permasalahan penjadwalan pasien, penelitian oleh (Clay et al., 2021) pada permasalahan TSP, dan penelitian oleh (Fonseca, Santos and Carrano, 2016) pada permasalahan penjadwalan sekolah. Hasil dari ketiga penelitian tersebut menunjukan hasil yang sama yaitu algoritma LAHC mempu menghasilkan hasil yang kompetitif.

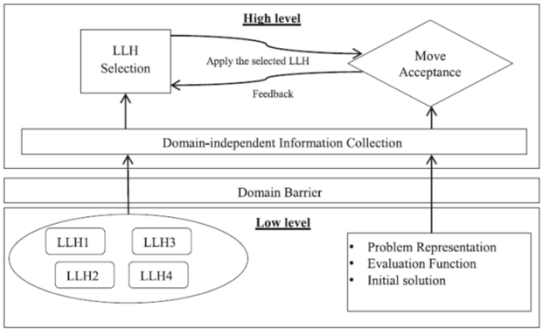
Sementara itu dalam mengembangkan algoritma *non-deterministic*, metode hiper-heuristik merupakan salah satu yang banyak digunakan peneliti. Metode ini merupakan metode pengembangan algoritma dengan mengembangkan tiga komponen yaitu *move acceptance*, *Low-Level Heuristic* (LLH) *selection* dan LLH (Burke et al., 2019). Metode ini telah diterapkan pada berbagai permasalahan optimasi kombinatorika seperti pada penelitian (Choong, Wong and Lim, 2019) dengan mengembangkan algoritma Artificial Bee Colony untuk permasalahan TSP, penelitian (Kheiri et al., 2021) dengan mengembangkan algoritma Hidden Markov untuk permasalahan penjadwalan suster, penelitian (Qin et al., 2021) dengan mengembangkan algoritma Reinforcement Learning untuk permasalahan *vehicle routing*.

Studi kasus TSC 2.0 menyajikan permasalahan nyata dari bentuk permasalahan TSP. Permasalahan ini bertujuan untuk mencari rute perjalanan termurah untuk mengunjungi beberapa negara, dimana pada setiap negara hanya akan mengunjungi satu kota dan diakhir perjalanan akan kembali ke kota dimana perjalanan dimulai. Permasalahan ini dikembangkan oleh perusahaan Kiwi yang merupakan perusahaan yang bergerak dibidang penjualan tiket transportasi udara. Permasalahan ini juga menyajikan dataset buatan dan juga dataset dari permasalahan nyata.

Berdasarkan latar belakang yang telah diuraikan diatas, maka penelitian ini mengembangkan algoritma LAHC dengan metode hiper-heuristik untuk dapat menyelesaikan permasalahan TSP pada studi kasus TSC 2.0. Dataset yang digunakan merupakan 14 dataset dari TSC 2.0 yang terdiri dari dataset buatan dan real-world. Hasil dari pengembangan algoritma LAHC dibandingkan dengan algoritma TA untuk melihat performa yang dihasilkan.

2. METODE PENELITIAN

Pada penelitian ini terdapat beberapa tahapan yang dilakukan mulai dari pembentukan solusi *feasible* (solusi yang tidak melanggar batasan permasalahan), pengembangan algoritma menggunakan metode hiper-heuristik, pengaturan parameter dan evaluasi hasil. Pada penggunaan metode hiper-heuristik, terdapat tiga komponen utama yaitu *move acceptance*, LLH *selection* dan LLH. Gambar 1 menunjukan keterkaitan antara ketiga komponen tersebut (Choong, Wong and Lim, 2019). Pada penelitian ini akan memfokuskan pada pengembangan *move acceptance* menggunakan algoritma LAHC dan pengembangan LLH. Sementara pada bagian LLH *selection* akan menggunakan metode acak seperti pada penelitian (Kheiri, Özcan and Parkes, 2016) dengan memilih satu dari kumpulan LLH secara acak pada setiap iterasinya.



Gambar 1. Komponen Hiper-Heuristik

2.1. Dataset TSC 2.0

TSC 2.0 menyajikan 14 dataset dalam kompetisi yang diadakan. Berdasarkan sumber dataset, dataset dapat dibagi menjadi dua yaitu dataset 1 sampai 10 yang dibentuk secara buatan dan dataset 11 sampai 14 yang diambil dari permasalahan nyata. Berdasarkan jumlah kota, dataset ini dapat dibagi menjadi tiga

bagian yaitu dataset 1-3 merupakan dataset kecil, 4-6 merupakan dataset sedang dan 7-14 merupakan dataset besar. Tabel 1 menunjukan deskripis dari masing-masing dataset.

Tabel 1. Dataset TSC 2.0

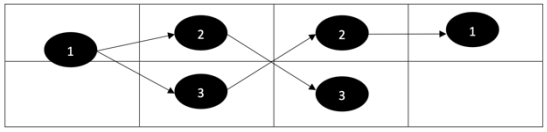
Dataset	Area	Kota	Jenis
1	10	10	Buatan
2	10	15	Buatan
3	13	38	Buatan
4	40	99	Buatan
5	46	138	Buatan
6	96	192	Buatan
7	150	300	Buatan
8	200	300	Buatan
9	250	250	Buatan
10	300	300	Buatan
11	150	200	Nyata
12	200	250	Nyata
13	250	275	Nyata
14	300	300	Nyata

2.2. Pembentukan Solusi Feasible

Pada permasalahan TSC 2.0, solusi *feasible* tidak dapat dihasilkan hanya dengan menjadwalkan rute secara acak seperti pada permasalahan TSP pada umumnya. Hal ini terjadi karena pada permasalahan TSC 2.0, tidak semua kota memiliki penerbangan ke seluruh kota lainnya. Sehingga saat membentuk rute secara acak, memungkinkan untuk menemukan kota yang tidak memiliki penerbangan ke kota lainnya yang belum dikunjungi pada hari berikutnya.

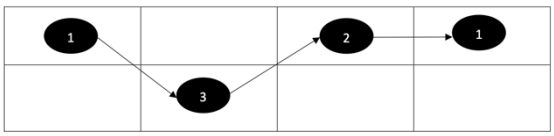
Untuk mengatasi hal tersebut maka pada penelitian ini akan memetakan terlebih dahulu seluruh kota pada setiap harinya dalam matrix dua dimensi. Sumbu x akan menunjukan hari perjalanan, sedangkan sumbu y akan menunjukan kota yang tersedia pada hari tersebut. Gambar 2 menunjukan ilustrasi dari pemetaan kota dan hari dalam matriks.

Langkah selanjutnya adalah melakukan eliminiasi kota-kota yang tidak memiliki penerbangan ke kota yang tersedia pada hari selanjutnya. Gambar 3 menunjukan ilustrasi dari hasil proses eliminiasi yang dilakukan terhadap gambar 2. Pada hari ke tiga, kota nomor tiga tidak memilii penerbangan ke kota di hari selanjutnya, sehingga kota ini dihapus. Penghapusan ini juga berdampak pada kota dua di hari kedua yang menjadi tidak memiliki penerbangan ke kota lain di hari selanjutnya. Sehingga kota dua pada hari kedua juga dihapus.



Gambar 2. Ilustrasi pemetaan kota dan hari

Setelah proses eliminasi dilakukan, Setiap kota pada setiap hari yang tersedia dalam matriks akan dipilih secara acak. Pemilihan kota juga dilakukan dengan memperhatikan batasan-batasan yang ada dalam permasalahan TSP. Hasil dari tahapan ini berupa rute penerbangan yang bisa digunakan dengan tidak melanggar batasan yang ada.



Gambar 3. Ilustrasi hasil eliminasi kota

2.3. Pengembangan Algoritma LAHC

Pada tahapan ini algoritma LAHC dikembangkan pada tahapan *move acceptance* yaitu tahapan untuk menentukan apakah solusi baru yang dihasilkan akan digunakan atau tidak. Gambar 4 menunjukan pseudocode dari algoritma LAHC (Burke and Bykov, 2017). Algoritma ini bekerja dengan menyimpan data solusi dari iterasi sebelumnya dalam sebuah daftar penyimpanan dengan panjang daftar yang telah ditentukan. Saat solusi baru dihasilkan, maka solusi baru akan dibandingkan dengan solusi saat ini. Jika solusi baru memiliki biaya perjalanan yang lebih rendah atau sama, maka solusi baru akan menggantikan solusi saat ini. Jika solusi baru menghasilkan biaya yang lebih mahal dari pada solusi saat ini, maka solusi baru akan dibandingkan dengan salah satu solusi yang tersimpan dalam daftar solusi. Daftar solusi akan diperbarui setiap ada solusi baru yang lebih baik dari yang disimpan dalam daftar solusi.

```
Produce an initial solution s
Calculate initial cost function C(s)
Specify Lh
For all k∈{0...Lh-1} fk:=C(s)
First iteration I:=0; Iidle:=0
Do until (I>100000) and (Iidle>I*0.02)
   Construct a candidate solution s*
   Calculate a candidate cost function C(s*)
   If C(s*)≥C(s)
      Then increment the idle iterations number Iidle:=Iidle+1
      Else reset the idle iterations number Iidle:=0
   Calculate the virtual beginning v:=I mod Lh
   If C(s*)<fv or C(s*)≤C(s)
      Then accept the candidate s:=s*
      Else reject the candidate s:=s
   If C(s)<fv
      Then update the fitness array fv:=C(s)
   Increment the iteration number I:=I+1
```

Gambar 4. Pseudocode Algoritma LAHC

2.4. Pengembangan LLH

Pada penelitian ini, LLH yang digunakan adalah *swap* yaitu jenis LLH yang menukar dua kota yang dipilih secara acak. Pengembangan *swap* dilakukan dengan menjalankan *swap* sebanyak empat kali. Setelah itu akan dilakukan pengecekan dari hasil swap terakhir apakan akan terima oleh algoritma LAHC atau tidak. Jika tidak diterima maka proses *swap* terakhir akan dikembalikan. Hal ini akan dilakukan secara berulang sampai ada solusi yang diterima LAHC atau sampai pada titik dimana semua proses swap telah dikembalikan kekeadaan awal.

2.5. Uji Parameter

Untuk mendapatkan performa algoritma yang optimal, penelitian ini melakukan tahapan uji coba parameter. Pada algoritma yang dikembangkan terdapat satu parameter yang menentukan performa dari algoritma. Parameter tersebut adalah panjang daftar penyimpanan riwayat solusi. Penelitian sebelumnya menunjukan menggunakan nilai yang terlalu besar akan menyebabkan pencarian solusi tidak mampu menurunkan biaya perjalanan dengan baik, namun menggunakan nilai yang terlalu kecil akan menyebabkan pencarian solusi terjebak dalam kondisi local optima.Pada penelitian ini akan menggunakan nilai 50, 500, 1000 dan 5000. Hasil terbaik akan digunakan dalam tahapan evaluasi hasil.

2.6. Evaluasi Hasil

Untuk melihat performa dari algoritma yang dikembangkan, maka hasil dari algoritma LAHC akan dibandingkan dengan algoritma TA. Algoritma TA merupakan algoritma yang memiliki konsep sama dengan LAHC yaitu melakukan pencarian pada satu solusi. Selain itu algoritma TA juga hanya memiliki satu parameter yaitu parameter ambang batas untuk menentukan solusi baru akan diterima atau tidak. Algoritma TA juga akan dilakukan uji coba parameter dengan menggunakan empat nilai yang diuji coba yaitu 0.1%, 0.25%, 0.5% dan 1%. Proses uji coba dilakukan dengan metode yang sama dengan algoritma LAHC.

3. HASIL DAN PEMBAHASAN

3.1. Pembentukan Solusi Feasible

Pada tahapan pembentukan solusi *feasible*, setelah melakukan proses pemetaan dalam matriks dan eliminasi kota, seluruh dataset telah ditemukan solusi *feasible*. Tabel 2 menujukan nilai biaya perjalanan yang diperlukan untuk setiap solusi *feasible* yang ditemukan.

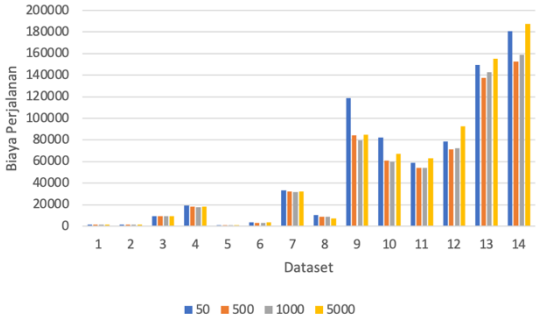
3.2. Hasil Optimasi LAHC

Setelah didapatkan solusi *feasible*, maka solusi tersebut di optimasi menggunakan algoritma LAHC. Algoritma LAHC dengan empat jenis nilai parameter dijalankan dengan jumlah iterasi sebanyak 100.000.000 sebanyak 15 kali pada setiap datasetnya. Hasilnya didapatkan nilai parameter 1000 menghasilkan nilai solusi terbaik. Hasil perbandingan antara keempat nilai parameter dapat dilihat pada gambar 5. Nilai parameter yang terlalu besar akan menghasilkan eksplorasi solusi yang telalu lama sehingga tidak bisa menurunkan solusi dengan optimal. Sebaliknya, nilai parameter yang terlalu kecil akan terlalu berfokus untuk menurunkan nilai solusi dan kurang melakukan eksplorasi. Sehingga pencarian akan rentan terjebak dalam kondisi local optima.

Tabel 2. Hasil Pembentukan Solusi Feasibel

Dataset	Biaya	Status
1	24077	Feasible
2	1498	Feasible
3	20976	Feasible
4	54369	Feasible
5	5263	Feasible
6	13136	Feasible
7	96355	Feasible
8	24290	Feasible
9	296088	Feasible
10	375497	Feasible
11	106724	Feasible
12	147863	Feasible
13	193882	Feasible
14	235770	Feasible

Hasil optimasi ini menunjukan hasil yang baik saat dibandingkan dengan solusi awal. Algoritma LAHC mampu menurunkan biaya perjalanan dengan rata-rata 58% dari solusi awal. Pada dataset buatan, algoritma LAHC mampu menurunkan biaya perjalanan dengan rata-rata 66%. Pada dataset nyata, algoritma LAHC menurunkan biaya perjalanan dengan rata-rata sebesar 40%. Tabel 3 menunjukan perbandingan antara solusi awal dan hasil optimasi pada setiap datasetnya.



Gambar 5. Perbandingan Parameter Algoritma LAHC

Tabel 3. Perbandingan Solusi Awal dan Hasil Optimasi

Dataset	Solusi Awal	Hasil optimasi	Persentase
1	24077	1456	94%
2	1498	1498	0%
3	20976	9431	55%
4	54369	17766	67%
5	5263	1184	77%
6	13136	3270	75%
7	96355	31637	67%
8	24290	8708	64%
9	296088	79880	73%
10	375497	59935	84%
11	106724	54100	49%
12	147863	72152	51%
13	193882	142565	26%
14	235770	158895	33%

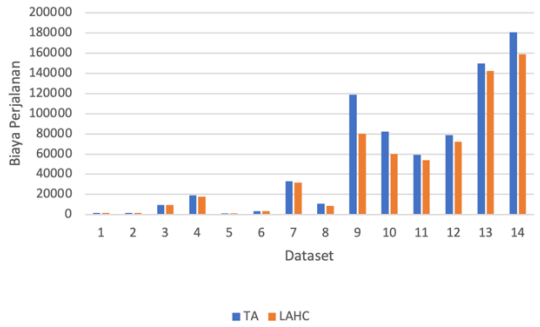
3.3. Perbandingan dengan Algoritma TA

Pada algoritma TA, setelah dilakukan uji coba parameter dengan metode yang sama, didapatkan hasil nilai ambang batas penerimaan terbaik sebesar 0.25%. Nilai solusi dari algoritma TA dengan nilai ambang batas 0.25% digunakan sebagai pembanding dengan algoritma LAHC. Hasilnya didapatkan

algoritma LAHC mampu menghasilkan hasil yang lebih baik pada 10 dataset. Algoritma TA hanya unggul pada dua dataset yaitu dataset ke 3 dan 5. Sementara itu pada dataset 1 dan 2, kedua algoritma menghasilkan hasil solusi yang sama. Secara keseluruhan algoritma LAHC mampu menghasilkan hasil dengan rata-rata menghasilkan biaya perjalanan lebih murah sebesar 9%. Tabel 4 dan gambar 6menunjukan perbandingan hasil dari kedua algoritma.

Tabel 4. Perbandingan Algoritma TA dan LAHC

Dataset	TA	LAHC	Persentase
1	1456	1456	0%
2	1498	1498	0%
3	9342	9431	-1%
4	19202	17766	7%
5	1166	1184	-2%
6	3485	3270	6%
7	33194	31637	5%
8	10567	8708	18%
9	118782	79880	33%
10	82200	59935	27%
11	59124	54100	8%
12	78793	72152	8%
13	149626	142565	5%
14	180696	158895	12%



Gambar 6. Algoritma LAHC dan TA

4. KESIMPULAN

Penelitian ini mengembangkan algoritma LAHC dengan menggunakan metode hiper-heuristik untuk menyelesaikan salah satu penerapan permasalahan TSP yaitu TSC 2.0. Pemilihan algoritma LAHC dan metode hiper-heuristik dilakukan karena penelitian sebelumnya dibidang optimasi kombinatorika telah membuktikan performa yang baik dari algoritma tersebut. Hasil dari pengembangan algoritma menunjukan hasil yang baik dengan mampu mengoptimasi dengan rata-rata 58% dari solusi awal. Selain itu algoritma ini juga unggul dari algoritma TA dengan mampu menghasilkan biaya perjalanan dengan rata rata 9% lebih rendah.

DAFTAR PUSTAKA

AKHAND, M.A.H., AYON, S.I., SHAHRIYAR, S.A., SIDDIQUE, N. and ADELI, H., 2020. Discrete Spider Monkey Optimization for Travelling Salesman Problem. *Applied Soft Computing Journal*, 86. https://doi.org/10.1016/j.asoc.2019.105887.

Al-FURHUD, M.A. and HUSSAIN, Z., 2020. Genetic Algorithms for the Multiple Travelling Salesman Problem. *International Journal of Advanced Computer Science and Applications*, [online] 11(7). https://doi.org/10.14569/IJACSA.2020.0110768.

BOLAJI, A.L. ARO, BAMIGBOLA, A.F. and SHOLA, P.B., 2018. Late acceptance hill climbing algorithm for solving patient admission scheduling problem. *Knowledge-Based Systems*, 145, pp.197–206. https://doi.org/10.1016/J.KNOSYS.2018.01.017.

BURKE, E.K. and BYKOV, Y., 2017. The late acceptance Hill-Climbing heuristic. *European Journal of Operational Research*, [online] 258(1), pp.70–78. https://doi.org/10.1016/j.ejor.2016.07.012.

BURKE, E.K., HYDE, M.R., KENDALL, G., OCHOA, G., ÖZCAN, E. and WOODWARD, J.R., 2019. A classification of hyper-heuristic approaches: Revisited. In: *International Series in Operations Research and Management Science*. https://doi.org/10.1007/978-3-319-91086-4_14.

CHEIKHROUHOU, O. and KHOUFI, I., 2021. A comprehensive survey on the Multiple Traveling Salesman Problem: Applications, approaches and taxonomy. *Computer Science Review*, 40, p.100369. https://doi.org/10.1016/J.COSREV.2021.100369.

CHOONG, S.S., WONG, L.P. and LIM, C.P., 2019. An artificial bee colony algorithm with a Modified Choice Function for the traveling salesman problem. *Swarm and Evolutionary Computation*, 44. https://doi.org/10.1016/j.swevo.2018.08.004.

CINAR, A.C., KORKMAZ, S. and KIRAN, M.S., 2020. A discrete tree-seed algorithm for solving symmetric traveling salesman problem. *Engineering Science and Technology, an International Journal*, [online] 23(4), pp.879–890. https://doi.org/10.1016/j.jestch.2019.11.005

CLAY, S., MOUSIN, L., VEERAPEN, N. and JOURDAN, L., 2021. CLAHC - Custom late acceptance hill climbing: First results on TSP. In: *GECCO 2021 Companion - Proceedings of the 2021 Genetic and Evolutionary Computation Conference Companion*. Association for Computing

Machinery, Inc. pp.1970–1973. https://doi.org/10.1145/3449726.3463129.

FONSECA, G.H.G., SANTOS, H.G. and CARRANO, E.G., 2016. Late acceptance hill-climbing for high school timetabling. *Journal of Scheduling*, [online] 19(4), pp.453–465. https://doi.org/10.1007/s10951-015-0458-5.

KHEIRI, A., GRETSISTA, A., KEEDWELL, E., Lulli, G., Epitropakis, M.G. and Burke, E.K., 2021. A hyper-heuristic approach based upon a hidden Markov model for the multi-stage nurse rostering problem. *Computers and Operations Research*, 130. https://doi.org/10.1016/j.cor.2021.105221.

KHEIRI, A., ÖZCAN, E. and PARKES, A.J., 2016. A stochastic local search algorithm with adaptive acceptance for high-school timetabling. *Annals of Operations Research*, [online] 239(1), pp.135–151. https://doi.org/10.1007/s10479-014-1660-0.

PANDIRI, V. and SINGH, A., 2019. An artificial bee colony algorithm with variable degree of perturbation for the generalized covering traveling salesman problem. *Applied Soft Computing Journal*, 78. https://doi.org/10.1016/j.asoc.2019.03.001.

PREMANANDA, I.G.A., TJAHYANTO, A. and MUKLASON, A., 2022. Hybrid Whale Optimization Algorithm for Solving Timetabling Problems of ITC 2019. In: *2022 IEEE International Conference on Cybernetics and Computational Intelligence (CyberneticsCom)*. IEEE. pp.317–322. https://doi.org/10.1109/CyberneticsCom55287.2022.9865647.

QIM, W., ZHUANG, Z., HUANG, Z. and HUANG, H., 2021. A novel reinforcement learning-based hyper-heuristic for heterogeneous vehicle routing problem. *Computers & Industrial Engineering*, 156, p.107252. https://doi.org/10.1016/J.CIE.2021.107252.