

Perbandingan Algoritma *Simulated Annealing* dan *Harmony Search* dalam Penerapan *Picking Order Sequence*

Tanti Octavia^{1*}, Septianda Angelica¹

Abstract: Implementation of mobile rack warehouse is commonly used in manufacturing industry because it can minimize the warehouse area used. Applying picking orders in taking of Stock Keeping Unit (SKU) on mobile rack warehouses could give fast loading order. This research aims to find out which algorithm is better in applying picking order sequence in mobile rack warehouse. The algorithms used are Simulated Annealing (SA) and Harmony Search (HS) algorithm. Both of these algorithms will be compared in terms of the gap with the shortest path method and CPU time. The result shows that the HS algorithm produces a better solution than the SA algorithm with lower CPU time, but the convergence rate of HS is lower than that of SA. HS was able to produce a better solution than the shortest path method of nine cases, while SA only eight cases from fifteen cases.

Keywords: Picking order sequence, simulated annealing, harmony search.

Pendahuluan

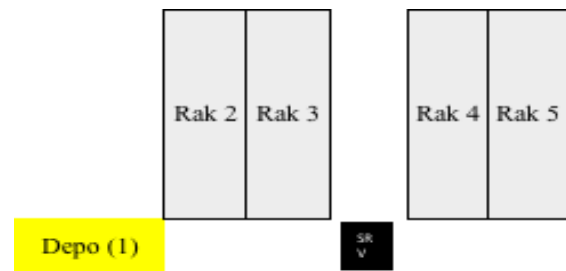
Penyimpanan memiliki peranan penting dalam industri manufaktur. Gudang sangat dibutuhkan dalam proses penyimpanan agar barang yang disimpan tidak rusak dan tetap terjaga. Pengaturan gudang yang optimal juga dapat meminimalkan penggunaan tempat sehingga dapat meningkatkan kapasitas gudang yang ada. Adapun pengaturan gudang dapat dilakukan seperti mengatur peletakan barang yang disimpan, mengubah tata letak gudang yang ada, serta penggunaan media penyimpanan seperti rak.

Penggunaan *mobile rack warehouse* umumnya cocok digunakan karena dapat menghemat tempat yang tersedia dan dapat bergerak secara otomatis. Dimana, pengambilan *stock keeping unit* (SKU) yang terletak pada rak dilakukan oleh *storage and retrieval vehicle* (SRV). Kecepatan dan waktu yang dibutuhkan untuk mengambil SKU sesuai yang diperlukan akan sangat tergantung pada penataan serta relokasi rak yang optimal agar waktu tempuh SRV dalam mengambil SKU semakin optimal.

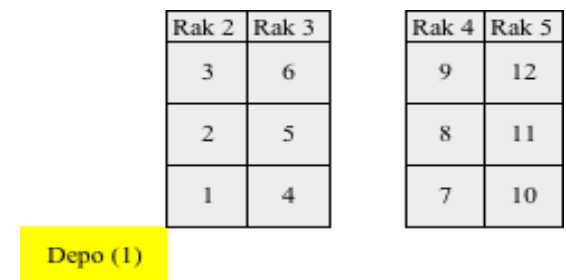
Penataan serta urutan pengambilan (*picking order sequence*) SKU dengan lokasi yang berdekatan akan mengoptimalkan waktu pengambilan SKU. Dimana pada penelitian ini akan dilakukan optimalisasi kecepatan dan waktu pengambilan SKU dengan menggunakan metode algoritma *simulated annealing* (SA) and *harmony search* (HS).

¹ Fakultas Teknologi Industri, Program Studi Teknik Industri, Universitas Kristen Petra, Jl. Siwalankerto 121-131, Surabaya 60238, Indonesia.
Email:tanti@petra.ac.id

* Penulis korespondensi



Gambar 1. Gambaran permasalahan



Gambar 2. Contoh penempatan SKU

Seperti contohnya sebuah gudang memiliki empat buah rak dan satu depo (Gambar 1). *Picking order* yang diberikan berupa nomor SKU yang memiliki *demand* tertentu dan selanjutnya dikelompokkan ke masing-masing rak.

Setiap SKU ditempatkan pada suatu rak saja dan tidak berpindah-pindah. Peletakan SKU pada setiap rak dilakukan secara berurutan, dimana nomor SKU terkecil akan terletak pada nomor rak terkecil (Gambar 2).

Pengambilan *order* yang pertama kali berdasarkan dari rak yang terbuka pertama kali. Rak yang terbuka pertama kali didapatkan dari membangkitkan bilangan acak. Pengambilan *order* membutuhkan

alokasi rak untuk membuka *aisle* dimana setiap rak hanya memiliki satu *aisle* saja. Pertama kali SRV akan berada di depo, kemudian SRV akan bergerak mengambil order ke masing-masing rak.

Pengaksesan SKU membutuhkan pergerakan rak selama empat meter/menit. SRV akan memiliki kapasitas tertentu, dimana pada penelitian ini SRV memiliki kapasitas 100.

Metode Penelitian

Pengembangan Model

Model yang akan dikembangkan pada penelitian ini terdiri dari: (1) model awal dengan algoritma *simulated annealing* (SA) dan *harmony search* (HS); (2) model dengan *shortest path* (SP). Model awal dikembangkan dengan cara membangkitkan rute pengambilan SKU secara *random*, sedangkan model *shortest path* dikembangkan dengan cara mencari rute dengan waktu tempuh terpendek. Metode ini akan digunakan sebagai pembandingan untuk algoritma SA dan HS karena metode *shortest path* cukup mampu menghasilkan solusi paling minimum meskipun membutuhkan waktu yang cukup lama.

Batasan pada penelitian ini adalah hanya menggunakan satu buah SRV dan SRV tidak diperbolehkan mengalami *backtracking*. Batasan ini ditujukan agar dapat meminimalkan waktu tempuh dari SRV. Adapun total waktu tempuh optimal dilihat pada persamaan berikut ini:

$$\text{Min } F(x) = \sum_{i=2}^n \sum_{j=1}^m \sum_{k=2}^n \sum_{l=1}^m W_{ik} X_{ij} X_{kl} \quad (1)$$

dengan kendala:

$$\sum_{i=2}^n di \times \sum_{j=1}^m \sum_{k=2}^n \sum_{l=1}^m X_{kl} X_{ij} \quad (2)$$

$$\sum_{j=1}^m X_{ij} = 1, \quad i = 2,3,4,5 \dots n \quad (3)$$

$$\sum_{i=2}^n X_{ij} = 1, \quad j = 2,3,4,5 \dots n \quad (4)$$

dimana:

W_{ik} = Waktu tempuh rak i ke rak k

$X_{ij} = \begin{cases} 1 \\ 0 \end{cases}$

$X_{ij} = \text{integer}$

$i \neq k, j \neq l$

Persamaan (2) digunakan untuk membatasi total permintaan yang dibawa oleh SRV tidak melebihi kapasitas. Persamaan (3) dan (4) menunjukkan bahwa semua kemungkinan peletakan rak i pada rute j akan dilakukan sesuai dengan jumlah rak (n) dan rute (m) dan rak i hanya berada pada satu rute j . Jumlah rak dimulai dari angka 2 karena angka 1 menunjukkan depo dimana depo tidak boleh dikunjungi apabila kapasitas SRV masih ada. X_{ij} bernilai 1 jika rak i berada pada rute j dan akan bernilai 0 jika berlaku sebaliknya. Contoh terdapat sebuah rute

yaitu (2,3). X_{ij} pada rute ini adalah X_{21} yang berarti rak 2 berada di rute 1. X_{kl} pada rute ini adalah X_{32} yang berarti rak 3 berada di rute 2. W_{kl} pada rute ini adalah W_{23} yang berarti waktu tempuh dari rak 2 ke rak 3.

Model Shortest Path

Metode ini akan mencari rute dengan waktu tempuh terpendek dengan membandingkan rute mana yang memiliki waktu tempuh paling pendek.

Contohnya pada Gambar 3 rak nomor 2 akan diperiksa ke seluruh rute untuk mengetahui rute manakah yang memiliki waktu tempuh terpendek. Dari hasil pencarian, rak nomor 2 memilih rute menuju rak nomor 3 karena rute tersebut memiliki waktu tempuh terpendek. Langkah ini kemudian diulang hingga semua permintaan terpenuhi.

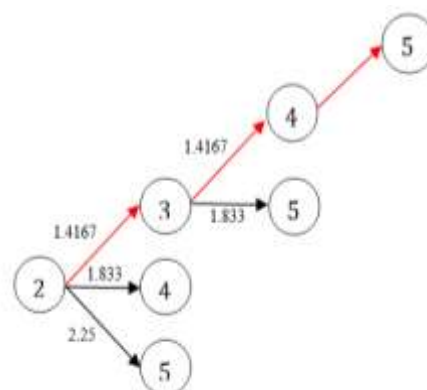
Algoritma Simulated Annealing (SA)

Simulated Annealing (SA) merupakan algoritma yang digunakan untuk mencari nilai optimal dengan mengontrol nilai dari temperatur (Atmaca dan Ozturk [1]). SA memiliki beberapa parameter yaitu temperatur awal, jumlah penurunan temperatur, derajat penurunan, dan jumlah replikasi setiap temperatur.

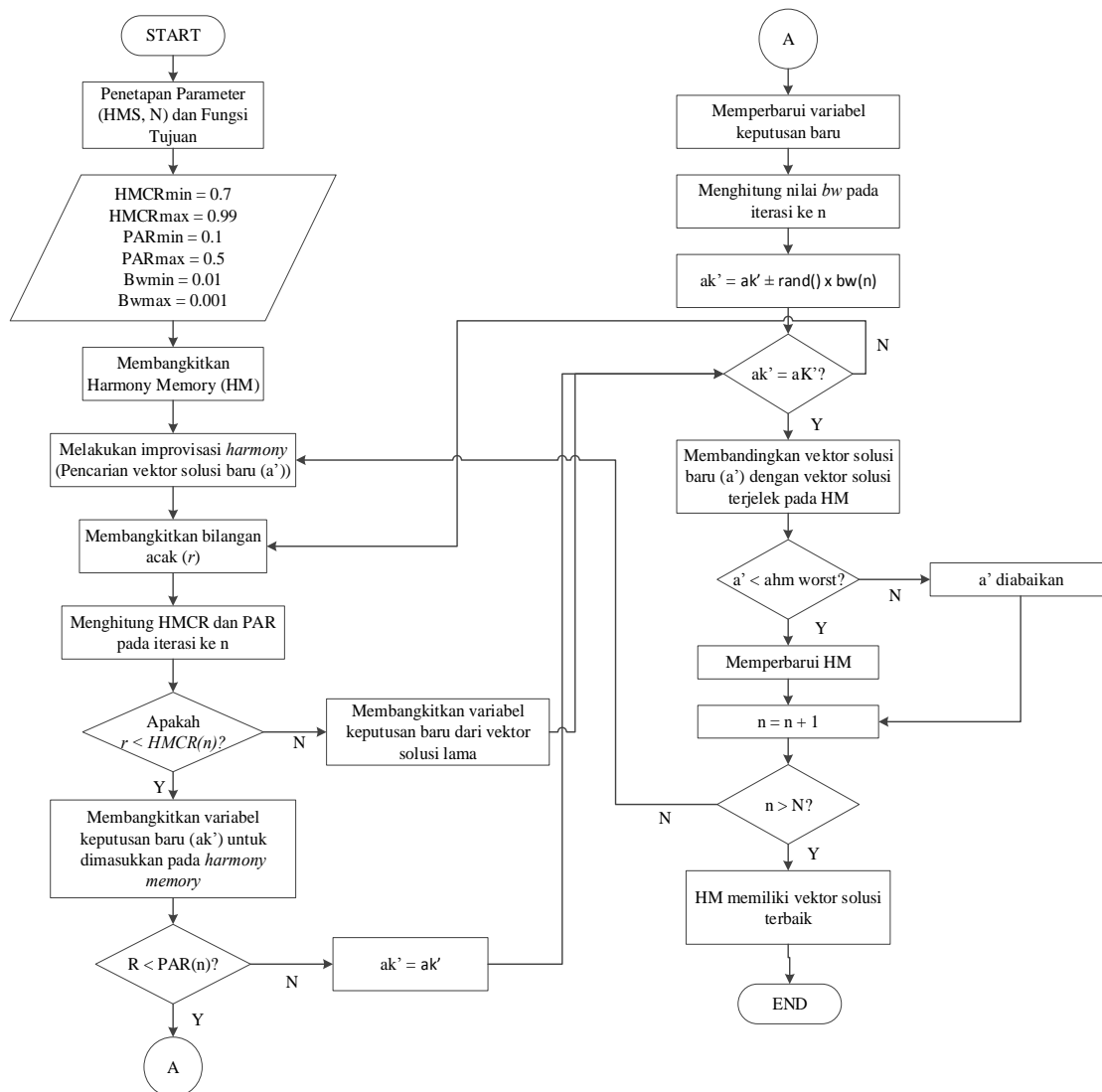
Adapun penetapan parameter pada penelitian ini mengacu pada Suman dan Kumar [2], Bosen et al [3], dan Yang [4]. Temperatur awal didapatkan dari persamaan berikut ini [4]:

$$T_0 = \frac{-\max \Delta f}{\ln P_0} \quad (5)$$

Nilai Δf didapatkan dari pengurangan antara solusi baru dan lama, dimana solusi baru didapatkan dari pengembangan solusi awal dengan menukar dua rute secara random. Misalnya pada solusi awal, hasil yang diperoleh adalah (2,3,4,5), pada solusi baru diperoleh (2,4,3,5). Maka hasil waktu tempuh solusi baru akan dikurangkan dengan hasil waktu tempuh dari solusi lama.



Gambar 3. Contoh penerapan metode *shortest path*



Gambar 4. Flowchart algoritma *harmony search*

SA akan membangkitkan solusi baru dengan dua metode yaitu *swap two* dan *push away*. Metode ini sebelumnya sudah dikembangkan oleh Bosen et al. [3]. Pada penelitian ini ditentukan metode *swap two* yang terjadi sebanyak 90% dan *push away* sebesar 10%. Contoh dari metode *swap two* adalah sebagai berikut. Ketika didapatkan solusi awal yaitu (2,4,-3,5) maka metode *swap two* akan menukar dua rute secara acak, sehingga solusi baru menjadi (2,3,5,4).

Contoh penggunaan dari metode *push away* adalah pada saat solusi awal didapatkan (2,3,5,4). Metode *push away* akan meletakkan salah satu rak ke rute paling terakhir sehingga solusi baru menjadi (2,5,4,3). Penentuan rak mana yang akan dipilih berdasarkan hasil *random data*.

Nilai dari P_0 menurut Suman dan Kumar [2] memiliki kisaran antara 0,5 – 0,95, sedangkan untuk penelitian nilai P_0 yang digunakan sebesar

0,95. Parameter selanjutnya yaitu derajat penurunan sebesar 0,995 (Bosen et al. [3]), jumlah penurunan temperatur sebanyak 1000 kali, dan jumlah replikasi setiap temperatur sebanyak 5 kali.

Algoritma *Harmony Search* (HS)

Harmony search (HS) adalah algoritma yang digunakan untuk mencari nilai optimal yang terinspirasi dari musisi yang mengimprovisasi *pitch instrument* untuk mendapatkan harmoni yang baik (Zong [5]). Beberapa penelitian dan pengembangan algoritma HS telah dilakukan baik dalam perbaikan parameter algoritma HS itu sendiri maupun perbaikan penerapan algoritma HS dengan algoritma meta-heuristik lain (Alia dan Mandava [6]), misalnya: Ant colony (Yun et al. [7]). Penerapan algoritma HS ini telah banyak dilakukan pada permasalahan *traveling salesman problem* (Yun et al.[7], Zong et al. [8]); *joshop problem* (Aulia [9]); maupun *four colour mapping* (Daham et al. [10]). Penerapan algoritma

HS pada ketiga permasalahan di atas telah memberikan hasil yang lebih baik.

Beberapa parameter yang dimiliki HS yaitu *harmony memory* (HM), *harmony memory consideration rate* (HMCR), *harmony memory size* (HMS), *pitch adjusting rate* (PAR), dan jumlah iterasi (N). Penetapan parameter pada penelitian ini berdasarkan dari Yang [4], Zong [5], dan Daham et al. [10].

$$HM = \begin{bmatrix} a_1^1 & a_2^1 & \dots & a_N^1 & f(a^1) \\ a_1^2 & a_2^2 & \dots & a_N^2 & f(a^2) \\ \vdots & \vdots & \dots & \vdots & \vdots \\ a_1^{HMS} & a_2^{HMS} & \dots & a_N^{HMS} & f(a^{HMS}) \end{bmatrix} \quad (6)$$

HM berisi vektor–vektor solusi sebanyak HMS, dimana vektor–vektor solusi pada penelitian ini didapatkan secara *random*. Parameter selanjutnya yaitu HMS sebesar 40, jumlah iterasi sebanyak 1000. Parameter HMCR dan PAR tidak ditetapkan secara pasti tetapi didapatkan melalui persamaan (7):

$$HMCR(min) = \left(\frac{HMCRmax - HMCRmin}{N} \right) \times n \quad (7)$$

$$PAR(min) = \left(\frac{PARmax - PARmin}{N} \right) \times n \quad (8)$$

Gambar 4 menunjukkan bagaimana algoritma HS dijalankan. HS akan membangkitkan vektor solusi baru berdasarkan dari vektor solusi yang berada di HMS.

Hasil dan Pembahasan

Simulasi dilakukan dengan menggunakan *visual basic application* pada aplikasi *Microsoft Excel*. Simulasi dijalankan menggunakan Laptop dengan spesifikasi prosesor Intel Gen-7 Core i7-7700HQ dan memori DDR4 16 GB 2400Mhz. Simulasi dilakukan pada kasus permasalahan dengan jumlah rak (10,20,30,40,50) dan jumlah *order* (25,50,100). Setiap kasus direplikasi sebanyak 50 kali.

Simulasi dilakukan pada kedua metode dengan mengganti parameter pada model sebanyak tiga kali. Pergantian parameter dilakukan untuk mengetahui pengaruh perubahan parameter terhadap hasil dari fungsi tujuan dan CPU time simulasi tersebut.

Parameter Pertama

Parameter ini adalah parameter awal untuk kedua algoritma. Parameter pertama SA adalah sebagai berikut:

Temperatur awal

Temperatur awal didapatkan dari persamaan (5). Perhitungan nilai Δf dilakukan untuk mengetahui nilai maksimum Δf dari permasalahan (lihat Tabel

Tabel 1. Nilai Δf setiap kasus

Rak/order	25	50	100
10	-7,6933	-11,861	-11,515
20	-34,186	-23,627	-43,358
30	-22,559	-34,921	-36,697
40	-39,922	-31,953	-79,021
50	-52,790	-62,564	-83,220

Tabel 2. Rata-rata selisih waktu terpendek algoritma SA dan HS dengan *shortest path* (SP) parameter pertama

Jumlah rak	Jumlah order	SA-SP	HS-SP
10	25	-6,8986	-13,535
	50	17,069	0
	100	18,211	-2,0783
20	25	47,791	18,795
	50	44,749	39,924
	100	-3,3398	-7,4702
30	25	-17,338	-35,935
	50	36,762	12,815
	100	36,461	-49,384
40	25	84,147	18,795
	50	-37,187	-37,237
	100	-8,8917	-26,021
50	25	-10,444	-30,821
	50	77,167	6,0613
	100	0	0

1). Nilai yang diambil adalah -7,6933 yaitu pada kasus jumlah rak 10 dengan jumlah order 25. Nilai ini akan digunakan pada perhitungan temperatur awal dengan P_0 sebesar 0,95. Temperatur awal yang didapatkan sebesar 149,99.

Parameter HS yang pertama adalah sebagai berikut:

Harmony Memory Size : 40

HMCR dan PAR didapatkan melalui persamaan (6) dan (7) yang dihitung pada setiap iterasi (n)

- Jumlah iterasi (N) : 1000
- Jumlah penurunan : 1000
- Derajat penurunan : 0,995
- Replikasi setiap temperatur: 5

Tabel 2 menunjukkan hasil antara kedua algoritma (SA dan HS) dengan metode *shortest path*. Hasil simulasi menunjukkan bahwa HS mampu mendapatkan hasil yang lebih baik daripada SA. Hal itu ditunjukkan dari 15 kasus yang dikembangkan, HS mampu lebih baik dari metode *shortest path* sebanyak 8 kasus. SA hanya mampu memiliki hasil yang lebih baik dari metode *shortest path* sebanyak 6 kasus.

Gambar 5 dan 6 menunjukkan bahwa algoritma SA lebih mampu menghasilkan hasil yang lebih konvergen dibandingkan algoritma HS. Hal ini dapat dilihat bahwa SA mulai menghasilkan solusi yang konvergen saat memasuki temperatur kelima atau memasuki iterasi ke 600. HS baru mulai menghasilkan solusi yang konvergen saat memasuki iterasi ke 900. Hal ini berarti SA mampu menghasilkan solusi dengan variasi yang lebih kecil dibandingkan HS.

Tabel 3 menunjukkan waktu pengerjaan masing-masing algoritma dalam satuan menit. SA memiliki rata-rata waktu pengerjaan selama 180,7826 menit, sedangkan HS selama 81,1056 menit. Hasil rata-rata ini menunjukkan bahwa HS dapat menghasilkan solusi yang lebih cepat dibandingkan SA.

Parameter Kedua

Parameter ini ditentukan dengan mengurangi jumlah iterasi untuk masing-masing algoritma. Parameter kedua algoritma SA adalah sebagai berikut :

- Temperatur awal : 149,99
- Jumlah penurunan : 500
- Derajat penurunan : 0,995
- Replikasi setiap temperatur: 5

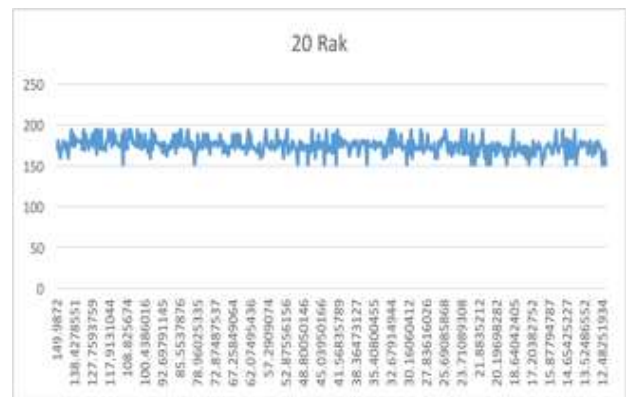
Perubahan temperatur SA ini berdasarkan dari hasil konvergensi yang menunjukkan bahwa SA sudah mampu menghasilkan solusi yang konvergen pada saat memasuki iterasi ke 600. Parameter HS yang kedua adalah sebagai berikut:

- Harmony memory size*: 40
- HMCR dan PAR didapatkan melalui persamaan (6) dan (7) yang dihitung setiap iterasi (n)
- Jumlah iterasi (N): 500

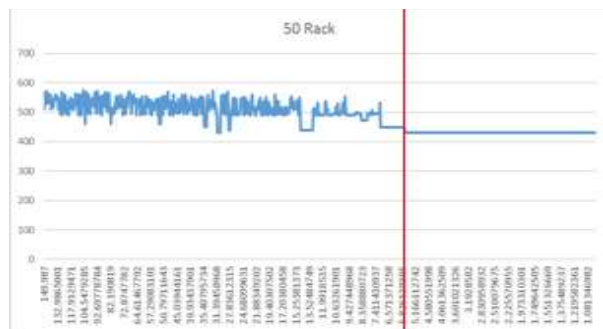
HS masih mendapatkan hasil yang lebih baik daripada SA. Hal itu ditunjukkan dari 15 kasus, HS memiliki hasil yang lebih baik dari metode *shortest path* sebanyak 9 kasus, sedangkan SA sebanyak 5 kasus.

Tabel 3. Hasil CPU *time* parameter pertama (dalam menit)

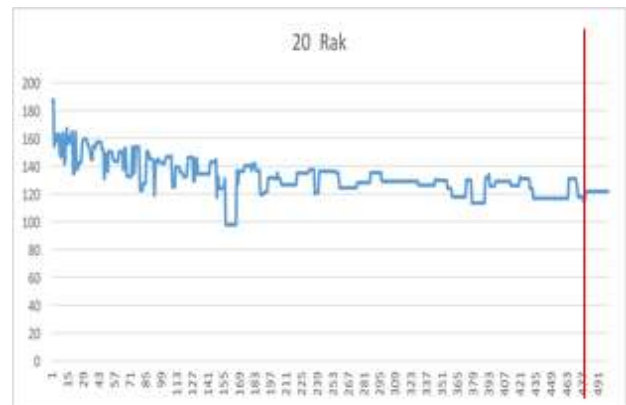
Jumlah rak	Jumlah order	SA	HS
10	25	110,1	37,433
	50	131,4	36,117
	100	134,783	42,883
20	25	145,25	36,117
	50	173,983	34,6
	100	181,917	30,9
30	25	142	31,9
	50	181,05	133,733
	100	221,067	134,167
40	25	147,983	41,85
	50	215,983	54
	100	248,04	144,05
50	25	214,083	125,367
	50	213,683	175,867
	100	250,417	157,6
Rata-rata		180,7826	81,1056



Gambar 7. Konvergensi *simulated annealing*



Gambar 5. Konvergensi *simulated annealing*



Gambar 8. Konvergensi *harmony search*



Gambar 6. Konvergensi *harmony search*

Hasil pergantian parameter menunjukkan bahwa hasilkan solusi yang lebih konvergen daripada parameter pertama. Hal itu dikarenakan jumlah iterasi yang semakin sedikit menunjukkan bahwa jumlah tersebut masih belum cukup untuk mendapatkan hasil yang konvergen. Parameter kedua ini HS lebih baik dalam menghasilkan solusi yang lebih konvergen, walaupun HS menghasilkan solusi yang konvergen pada saat memasuki iterasi ke 470.

Tabel 4. Rata-rata selisih waktu terpendek algoritma SA dan HS dengan *shortest path* (SP) parameter kedua

Jumlah rak	Jumlah order	SA-SP	HS-SP
10	25	-5,533536	-11,49738
	50	-12,42853	-10,4164
	100	-44,4382	-29,0162
20	25	15,076138	6,68741
	50	45,133386	-44,461196
	100	19,273	-27,4406
30	25	-42,02161	-31,48758
	50	71,38797	-25,184357
	100	14,1265	-21,4100053
40	25	56,7382	27,98565
	50	83,5331	11,201
	100	79,289318	28,1444
50	25	37,374504	-21,32427
	50	-15,01927	86,234
	100	2,140286	104,289925

Tabel 5. Hasil CPU time parameter kedua (dalam menit)

Jumlah rak	Jumlah order	SA	HS
10	25	56,133	22,7
	50	57,6167	32,133
	100	79	38,6333
20	25	70,433	18,783
	50	72,4	19,9167
	100	94,033	20,45
30	25	86,3833	21,2833
	50	95,8	25,733
	100	111,8	20,9667
40	25	95,433	34,3667
	50	100,433	27,6667
	100	130,417	38,5
50	25	106,75	35,9333
	50	111,217	29,5333
	100	155,717	41,2667
Rata-rata		94,9044	28,52438

Tabel 6. Rata-rata selisih waktu terpendek algoritma SA dan HS dengan *shortest path* (SP) parameter ketiga

Jumlah rak	Jumlah order	SA-SP	HS-SP
10	25	-6,2877	-11,49738
	50	-5,9812	-10,4164
	100	-26,613	-29,0162
20	25	19,8819	6,68741
	50	42,3377	-44,461196
	100	-45,152	-27,4406
30	25	12,5476	-31,48758
	50	34,4871	-25,184357
	100	-34,091	-21,4100053
40	25	-15	27,98565
	50	68,8062	11,201
	100	8,68817	28,1444
50	25	-11,008	-21,32427
	50	22,3898	86,234
	100	-24,371	104,289925

Tabel 5. menunjukkan SA memiliki rata-rata waktu pengerjaan selama 94,9044 menit, sedangkan HS selama 28,52438 menit. Hasil rata-rata ini menunjukkan bahwa HS masih lebih cepat dalam segi waktu pengerjaan dibandingkan SA.

Parameter Ketiga

Parameter ini ditentukan dengan mengganti derajat penurunan dari algoritma SA. Parameter kedua algoritma SA adalah sebagai berikut :

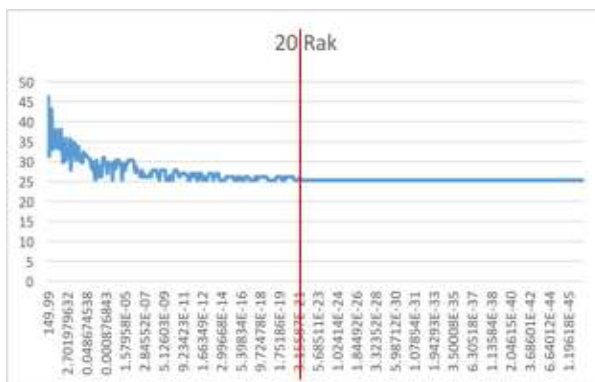
- Temperatur awal : 149,99
- Jumlah penurunan : 500
- Derajat penurunan : 0,8
- Replikasi setiap temperatur: 5

Pergantian derajat penurunan dilakukan untuk mengetahui penaruh derajat penurunan terhadap hasil yang didapatkan. Parameter HS yang kedua adalah sebagai berikut:

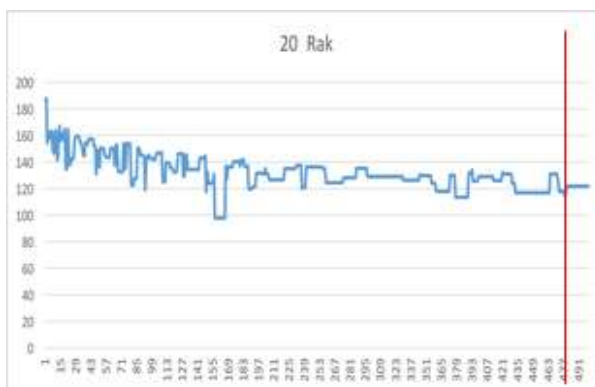
- Harmony Memory Size*: 40
- HMCR dan PAR didapatkan melalui persamaan (6) dan (7) yang dihitung setiap iterasi (n)
- Jumlah iterasi (N): 500

Hasil pergantian parameter ketiga menunjukkan bahwa HS masih mendapatkan hasil yang lebih baik daripada SA. Hal itu ditunjukkan dari 15 kasus, HS memiliki hasil yang lebih baik dari metode *shortest path* sebanyak 9 kasus, sedangkan SA sebanyak 8 kasus.

SA menghasilkan solusi yang paling baik pada saat parameter ketiga karena menghasilkan solusi yang lebih baik daripada metode *shortest path* paling banyak dibandingkan parameter lainnya.



Gambar 9. Konvergensi *simulated annealing*



Gambar 10. Konvergensi *harmony search*

Tabel 7. Hasil CPU *time* parameter ketiga (dalam menit)

Jumlah rak	Jumlah <i>order</i>	SA	HS
10	25	63,8167	22,7
	50	69,15	32,133
	100	70,2167	38,6333
20	25	63,8	18,783
	50	67,0667	19,9167
	100	72,95	20,45
30	25	63,4	21,2833
	50	94,9667	25,733
	100	105,8	20,9667
40	25	63,1667	34,3667
	50	112,5	27,6667
	100	127,4	38,5
50	25	95,6833	35,9333
	50	19,283	29,5333
	100	165	41,2667
Rata - rata		83,61332	28,52438

Gambar 9 dan 10 menunjukkan bahwa SA pada parameter ketiga mampu menghasilkan solusi yang paling konvergen diantara parameter-parameter sebelumnya. Hal itu ditunjukkan pada saat iterasi ke 235 atau pada saat temperatur $3.15E-21$ SA sudah menghasilkan solusi yang konvergen. SA mampu menghasilkan solusi yang lebih konvergen dibandingkan HS. Hal ini berarti SA parameter ketiga mampu menghasilkan solusi dengan varians yang kecil.

Tabel 7 menunjukkan SA memiliki rata-rata waktu pengerjaan selama 83,61332 menit, sedangkan HS selama 28,52438 menit. Hasil rata-rata ini menunjukkan bahwa HS membutuhkan waktu yang lebih cepat dalam memberikan hasil dibandingkan SA.

Simpulan

Hasil penelitian menunjukkan algoritma *simulated annealing* memberikan rata-rata waktu tempuh yang lebih singkat dibandingkan metode *shortest path* dengan rata-rata CPU *time* sebesar 180,78 menit untuk 8 dari 15 permasalahan. Hasil simulasi juga menunjukkan algoritma *harmony search* memberikan rata-rata waktu tempuh yang lebih pendek pada 9 dari 15 permasalahan dibandingkan metode *shortest path* dengan rata-rata CPU *time* sebesar 81,106 menit.

Algoritma *harmony search* menghasilkan solusi yang paling baik ketika jumlah iterasi 500 dan 40. Hasil penelitian juga menunjukkan algoritma *harmony search* memberikan solusi yang lebih baik dibandingkan algoritma *simulated annealing* dengan CPU *time* yang lebih rendah, akan tetapi, algoritma *simulated annealing* mampu menghasilkan solusi yang lebih konvergen dibandingkan *harmony search*.

Daftar Pustaka

1. Atmaca, E., Ozturk, A., Defining Order Picking Policy: A Storage Assignment Model and a Simulated Annealing Solution in AS/RS Systems, *Applied Mathematical Modelling*, 37, 2012, pp. 5069–5079.
2. Suman, B., and Kumar, P., A Survey of Simulated Annealing as a Tool for Single and Multiobjective Optimization, *Journal of the Operational Research Society*, 57, 2006, pp.1143–1160.
3. Boysen, N., Briskorn, D., and Emde, S., Sequencing of Picking Orders in Mobile Rack Warehouse, *European Journal of Operational Research*, 259(2017), 2016, pp. 293-307.
4. Yang, X. S., *Nature-Inspired Metaheuristic Algorithms*, United Kingdom: Luniver Press, 2010.
5. Zong, W. G., *Music Inspired Harmony Search Algorithm Theory and Applications*, New York: Springer, 2009.
6. Alia, O. M., Mandava, R., *The Variants of the Harmony Search Algorithm: An Overview*, Springer Science Business Media B.V., 2011.
7. Yun, H. Y., Jeong, S. J., Kim, K. S., Advanced Harmony Search with Ant Colony Optimization for Solving the Traveling Salesman Problem, *Journal of Applied Mathematics*, 2013, pp. 29-37.
8. Zong, W. G., Kim, J. H., Loganathan, G. V., A New Heuristic Optimization Algorithm: Harmony Search, *Simulation Council*, 75, 2001, pp. 60–68.
9. Aulia, I., Nababan, E. B., dan Muchtar, M. A. Penerapan Harmony Search Algorithm dalam Permasalahan Penjadwalan *Flow Shop*, *Jurnal Dunia Teknologi Informasi*, 1(1), 2012, pp. 1-7.
10. Daham, B. F., Mohammed, N. M., and Mohammed, S. K., Modified Harmony Search Algorithm for Solving the Four-Color Mapping Problem, *International Journal of Computer Application*, 91(6), 2014, pp. 975-8887.