

Algoritma Penjadwalan *Job Shop* Alternatif *Routing* Menggunakan *Greedy Randomized Adaptive Search Procedure With Fixed Threshold* Untuk Minimisasi *Makespan**

KHARIZA SYAILANDRA, ARIF IMRAN, EMSOSFI ZAINI

Jurusan Teknik Industri, Institut Teknologi Nasional (Itenas), Bandung

Email: kharizasyailandra@gmail.com

ABSTRAK

Penelitian ini membahas algoritma penjadwalan job shop dengan alternatif routing menggunakan greedy randomized adaptive search procedure with fixed threshold dengan fungsi tujuan minimisasi makespan. GRASP with fixed threshold adalah metode metaheuristik dua tahap untuk menyelesaikan masalah-masalah optimasi kombinatorial. Tahap pertama adalah tahap konstruksi jadwal inisial. Tahap kedua adalah tahap local search untuk memperbaiki jadwal inisial. Performansi algoritma usulan diuji melalui 2 skenario dengan menggunakan set data dari literatur. Hasil pengujian kedua skenario menunjukkan bahwa algoritma usulan memberikan solusi yang kompetitif jika dibandingkan dengan penelitian-penelitian sebelumnya.

Kata Kunci: *Penjadwalan Job Shop, Alternatif Routing, GRASP, Threshold Accepting*

ABSTRACT

In this paper we present algorithms job shop scheduling with alternative routing using a greedy randomized adaptive search procedu with a fixed threshold with makespan minimization objective function. GRASP with fixed threshold is a two-phase metaheuristic methods for solving combinatorial optimization problems. The first stage is the initial stage of construction. The second stage is the stage of local search to improve the initial schedule. Performance of the proposed algorithm was tested through two scenarios using data sets from the literature. Theresults indicate that the proposed algorithm provides a competitive solution when compared with previous studies.

Key words: *Job Shop Scheduling, Alternative Routing, GRASP, Threshold Accepting*

* Makalah ini merupakan ringkasan yang disusun oleh penulis pertama dengan pembimbingan penulis kedua dan ketiga. Makalah ini merupakan draft awal dan akan disempurnakan oleh para penulis untuk disajikan pada seminar nasional dan/atau jurnal nasional.

1. PENDAHULUAN

Masalah penjadwalan *job shop* klasik (*Classical Job Shop Scheduling Problem*, CJSSP) adalah masalah penugasan satu set job ($J_i = J_1, J_2, \dots, J_n$) pada satu set mesin ($M_m = M_1, M_2, \dots, M_m$). Setiap operasi dari suatu job ($O_{ij} = O_{i1}, O_{i2}, \dots, O_{ij}$) memiliki urutan pengerjaan (*routing*) yang berbeda-beda dan unik. Tujuannya adalah menghasilkan jadwal yang dapat meminimumkan waktu yang diperlukan untuk menyelesaikan seluruh *job* pada semua mesin (*makespan*). Seiring dengan berkembangnya teknologi pemesinan, maka suatu mesin dapat melakukan beberapa macam proses operasi. Seperti halnya mesin *milling* dengan mengganti mata pahat, maka mesin *milling* dapat melakukan proses yang sama dengan mesin bubut yaitu membuat lubang pada suatu material. Kemajuan teknologi pemesinan tersebut mengakibatkan sebuah operasi *job* dapat dikerjakan hanya dalam beberapa mesin sehingga sebuah operasi *job* akan memiliki beberapa alternatif mesin disebut juga sebagai alternatif *routing*.

Nasr dan Elsayed (1990) membahas permasalahan *job shop* alternatif mesin dengan menggunakan *greedy procedure* untuk meminimasi *mean flow time*. Penelitian ini hanya menghasilkan satu solusi algoritma yang digunakan dan hanya menghasilkan sebuah solusi. Feo dan Resende (1995) secara formal memperkenalkan GRASP sebagai pendekatan metaheuristik untuk memecahkan permasalahan *hard combinatorial optimization* (NP-Hard). *Greedy randomized adaptive search procedure* (GRASP) merupakan metode metaheuristik dua tahap berbasis *multi-start randomized search technique*. Selanjutnya Binato *et al*, (2001) mengusulkan GRASP untuk memecahkan masalah penjadwalan *job shop* klasik (CJSSP) dengan kriteria minimasi *makespan*. Model ini menggunakan dua tahap, yaitu tahap konstruksi dan tahap *local search*. Susanti (2010) telah membahas permasalahan *job shop* alternatif *routing* menggunakan GRASP dengan kriteria minimisasi *makespan*.

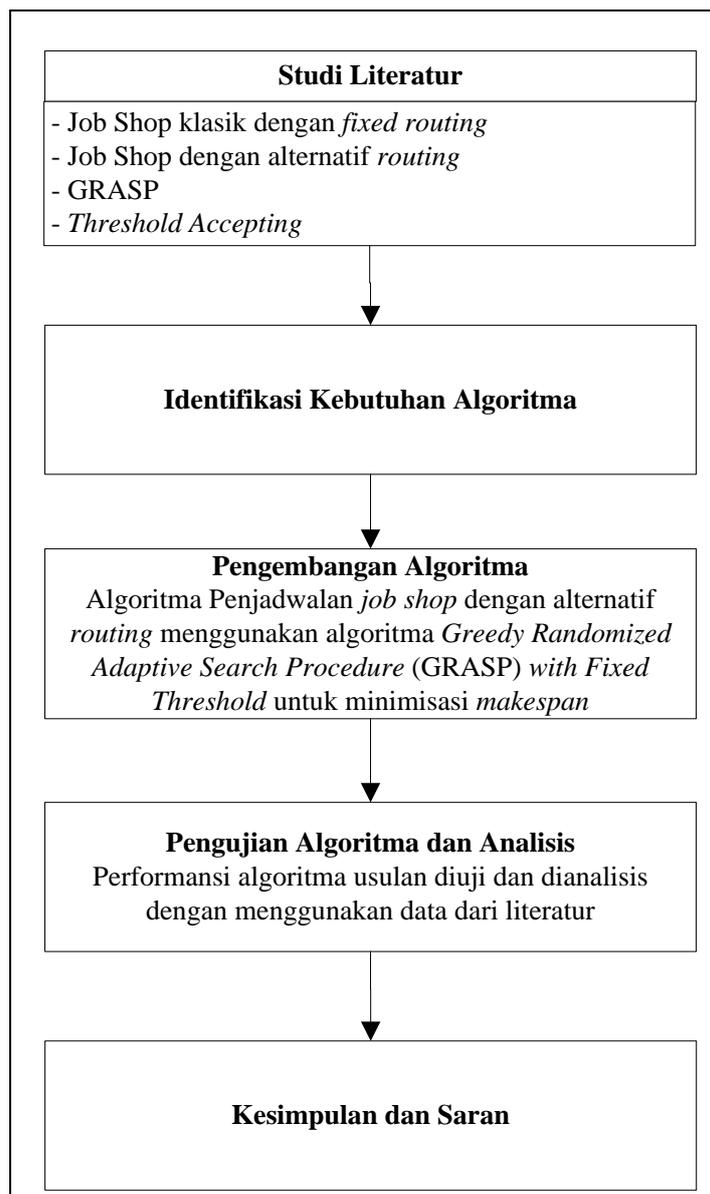
Adapun metode metaheuristik lainnya adalah *threshold accepting* yang diperkenalkan oleh Dueck dan Scheuer (1990) untuk menyelesaikan masalah optimisasi kombinatorial. Dalam metode ini, terdapat ambang batas toleransi untuk menerima suatu solusi dari solusi sebelumnya untuk di eksplorasi sehingga menemukan solusi yang lebih baik. Fungsi batas *threshold* tersebut dapat diadaptasikan ke algoritma yang lain. Berdasarkan penelitian-penelitian sebelumnya, penggabungan algoritma GRASP dengan fungsi *fixed threshold* belum dikembangkan sehingga perlu dilakukan penelitian pengembangan algoritma GRASP *with fixed threshold*. Penelitian ini bertujuan untuk menghasilkan algoritma GRASP *with fixed threshold* dengan kriteria minimisasi *makespan*.

Beberapa pembatas yang digunakan adalah:

1. Kemungkinan terjadinya kerusakan mesin selama proses pengerjaan tidak diperhitungkan.
2. Alternatif *routing* yang dibahas dalam model ini merupakan alternatif mesin.

2. METODOLOGI PENELITIAN

Metodologi penelitian disusun secara sistematis dan terarah yang digunakan sebagai suatu kerangka dalam sebuah penelitian ilmiah. Tahap dalam penelitian ini dapat dilihat pada Gambar 1.



Gambar 1. Metodologi Penelitian

Posisi penelitian ini terhadap penelitian sebelumnya dapat dilihat pada Gambar 2.

		Metaheuristik			
		<i>Greedy Procedure</i>	GRASP	<i>Threshold Accepting</i>	<i>GRASP With Fixed Threshold</i>
		Minimisasi Mean Flow Time	Minimisasi Makespan		Minimisasi Makespan
Job Shop	Klasik		Feo dan Resende (1995); Binato <i>et al.</i> (2001)	Dueck dan Scheuer (1990)	
	Alternatif Routing / Flexible Job Shop	Nasr dan Elsayed (1990)	Susanti (2010)		PENELITIAN

Gambar 2. Peta Posisi Penelitian

3. PENGEMBANGAN ALGORITMA

Notasi-notasi yang digunakan dalam algoritma GRASP *with fixed threshold* dalam menyelesaikan masalah penjadwalan *job shop* alternatif *routing* adalah:

- i = indeks *job* ($i = 1, 2, \dots, n$).
- j = indeks operasi *job* ($j = 1, 2, \dots, q$).
- k = indeks *job* ($k = 1, 2, \dots, m$).
- z = jumlah alternatif mesin.
- e = banyaknya iterasi.
- r_{ij} = waktu operasi *j* *job* *i* siap untuk dijadwalkan.
- R_k = *ready time* mesin k .
- t_{ijk} = waktu operasi *job* *i* operasi *j* di mesin k .
- Ω = himpunan operasi *job* *i* di mesin k untuk pemilihan alternatif.
- O_{ijk} = *job* *i* operasi *j* di mesin k .
- C_{ijk} = *completion time* operasi *job* *i* operasi *j* di mesin k .
- C_{ijk}^h = *completion time* operasi *job* *i* operasi *j* di mesin k untuk iterasi ke- h .
- RCL_T = *restricted candidate list* untuk pemilihan alternatif mesin operasi *j* *job* *i*.
- L = jumlah minimum operasi *job* *i* yang berada dalam RCL_T .
- \underline{h} = nilai *completion time* minimum dari penambahan operasi O_{ijk} pada operasi yang telah terjadwalkan.
- \bar{h} = nilai *completion time* maksimum dari penambahan operasi O_{ijk} pada operasi yang telah terjadwalkan.
- $\%_{Th}$ = persentase batas *threshold*
- MS_e = *makespan* pada iterasi ke- e .
- MS^h = *makespan* iterasi ke- h .
- MS_{Th} = batas atas nilai *makespan* berdasarkan $\%_{Th}$.
- L_k = operasi *job* yang menjadi lintasan kritis.
- h = banyaknya iterasi pada tahap *local search*.
- b = operasi terakhir pada operasi yang telah terjadwalkan.
- S = urutan jadwal yang dihasilkan pada tahap konstruksi.
- S_h = urutan jadwal yang dihasilkan pada iterasi ke- h .
- L_c = operasi *job* menjadi lintasan kritis untuk urutan penjadwalan yang terdapat dalam S_h .
- SO_L = operasi-operasi *job* terakhir yang berada pada lintasan kritis.

Langkah-langkah algoritma GRASP *with fixed threshold* dalam menyelesaikan masalah penjadwalan *job shop* alternatif *routing* dengan kriteria minimisasi *makespan* adalah:

a. Tahap 1 Konstruksi (Pemilihan Alternatif Mesin dan Penjadwalan Job)

Langkah 1

Input data matriks alternatif *routing*

Langkah 2

Set *ready time* job $r_{ij} [r_{ij} = 0 \forall i, j]$

Set *ready time* mesin $R_k [R_k = 0 \forall k]$

Set maxiter = $\max(2, \lceil \frac{n}{4} \rceil)$

Langkah 3

Set $e = 1$ dan $j = 1$

Langkah 4

Tentukan Ω ($\Omega = O_{ij}, \dots, O_{nq}$)

Langkah 5

Set $i = 1$

Langkah 6

Periksa apakah $z = 1$?

Jika ya, maka pilih O_{ijk} sebagai O^* dan lanjutkan ke Langkah 12.

Jika tidak, maka lanjutkan ke Langkah 7.

Langkah 7

Hitung $C_{ijk} = \text{Maks} [r_{ij}, R_k] + t_{ijk}$ (1)

Langkah 8

Set $L = \max(2, \lceil \frac{z}{2} \rceil)$ dan bangkitkan secara random parameter α dalam selang $]0,1[$

Langkah 9

Bentuk RCL_T dengan menggunakan persamaan

$RCL_T = \{ O_{ijk} \in \Omega \mid \underline{h} \leq h(O_{ijk}) \leq \underline{h} + \alpha(\bar{h} - \underline{h}) \}$ (2)

Langkah 10

Jika $|RCL_T| \geq L$, maka lanjutkan ke Langkah 11.

Lainnya, susun prospektif kandidat dari C_{ijk} terkecil ke C_{ijk} terbesar, pilih sebanyak L elemen untuk masuk ke dalam RCL_T dan lanjutkan ke Langkah 11.

Langkah 11

Pilih secara random satu operasi dalam RCL_T , nyatakan dalam O^*

Langkah 12

Hilangkan $job O^*$ dari Ω , ($\Omega = \Omega - \{O^*\}$). Perbaharui $r_{ij} \forall i$ dan $R_k \forall k$. dan jadwalkan.

Langkah 13

Set $i = i + 1$ dan periksa apakah $i > n$?

Jika ya, maka set $j = j + 1$ dan periksa apakah $j > q$?

Jika ya, Lanjutkan ke Langkah 14.

Jika tidak, maka kembali ke Langkah 4.

Jika tidak, kembali ke Langkah 6.

Langkah 14

Tampilkan *Gantt Chart* hasil penjadwalan dan *makespan*-nya (MSe).

Langkah 15

Tentukan nilai MS_{Th} dengan menggunakan persamaan:

$MS_{Th} = MSe + (MSe \times \%_{Th})$ (3)

b. Tahap 2 Local Search (Insert dan Exchange)

Langkah 16

Input S.

Langkah 17

Tentukan lintasan kritis, set operasi-operasi *job* terakhir yang berada pada lintasan kritis (SO_L).

Langkah 18

Set $h = 1$

Langkah 19

Tentukan operasi terakhir yang berada pada lintasan kritis (b).

Langkah 20

Periksa b apakah dapat dilakukan proses *insert*.

Jika ya, lakukan proses *insert* sebanyak *job* yang berada pada posisi b sampai $b - 1$ pada operasi-operasi yang terdapat pada lintasan kritis (SO_L), set S_h dan lanjutkan ke Langkah 21.

Jika tidak, set $b = b - 1$ kembali ke Langkah 19.

Langkah 21

Hitung *completion time* tiap operasi *job* dan *makespan*.

$C_{ijk}^n = \text{Maks} [r_{ij}, R_k] + t_{ijk}; \forall i = (1, 2, \dots, n)$ (4)

Set L_c dan SO_L .

Langkah 22

Apakah $MS^h \leq MS_{Th}$?

Jika ya, lanjut ke Langkah 23.

Jika tidak, kembali ke Langkah 19.

Langkah 23

Set : $MS_e = \min [MS_e, MS^h]$ dan $h = h+1$.

Langkah 24

Jika SO_L masih dapat dilakukan proses *insert*, maka kembali ke Langkah 19 dan lakukan proses *insert* sebanyak q kali. Lainnya, pilih *makespan* terkecil dari proses *insert* dan lanjutkan ke Langkah 25.

Langkah 25

Apakah SO_L dapat dilakukan proses *exchange*?

Jika ya, maka lanjutkan ke Langkah 26.

Jika tidak, lanjutkan ke Langkah 31.

Langkah 26

Tentukan operasi terakhir yang berada pada lintasan kritis (b).

Langkah 27

Periksa b apakah dapat dilakukan proses *exchange*.

Jika ya, lakukan proses *exchange* pada operasi *job* yang terdapat dalam SO_L dan lanjut ke Langkah 28.

Jika tidak, set $b = b - 1$ dan kembali ke Langkah 26.

Langkah 28

Hitung *completion time* tiap operasi *job* dan *makespan*.

$$C_{ijk}^n = \text{Maks} [r_{ij}, R_k] + t_{ijk}; \forall i = (1, 2, \dots, n) \quad (5)$$

Langkah 29

Apakah $MS^h \leq MS_{Th}$?

Jika ya, lanjut ke Langkah 30.

Jika tidak, kembali ke Langkah 26.

Langkah 30

Jika $MS^h > MS_e$, maka set $h = h+1$ dan kembali ke Langkah 26. Lainnya set $MS_e = MS^h$ dan lanjutkan ke Langkah 31.

Langkah 31

Set $e = e + 1$ dan periksa apakah $e > \text{maxiter}$?

Jika ya, maka set $MS = [MS_1, \dots, MS_e]$ dan tampilkan *Gantt Chart* MS terbaik. Lainnya, tampilkan *Gantt Chart* MS_e dan kembali ke Langkah 4.

4. PENGUJIAN ALGORITMA

Algoritma usulan diuji dengan menggunakan dua skenario. Skenario 1 menggunakan data penelitian Nasr dan Elsayed (1990) dan skenario 2 menggunakan data penelitian Brandimarte (1993). Algoritma usulan juga diuji menggunakan persentase *threshold* yang berbeda, yaitu 5%, 10%, dan 15%.

a. Skenario 1

Set data yang digunakan pada Skenario 1 dapat dilihat pada Gambar 3.

		Mesin					
		1	2	3	4	5	6
Job 1	O ₁₁	2	3	4			
	O ₁₂		3		2	4	
	O ₁₃	1	4	5			
Job 2	O ₂₁	3		5		2	
	O ₂₂	4	3			6	
	O ₂₃			4		7	11
Job 3	O ₃₁	5	6				
	O ₃₂		4		3	5	
	O ₃₃			13		9	12
Job 4	O ₄₁	9		7	9		
	O ₄₂		6		4		5
	O ₄₃	1		3			3

Gambar 3. Set Data Skenario 1

b. Skenario 2

Set data yang digunakan pada Skenario 2 dapat dilihat pada Gambar 4.

10	6	2
6	2153435335212346236526111313663643	
5	126131112226463652611	
5	126234623652611334266621155	
5	3652611126131353352123462	
6	3533521365261112621534226463342666	
6	234621123342666126365261121342	
5	1612134233426663265116131	
5	234623342666365261112622646	
6	161211553663643112334266622646	
6	23462334266635335211612264621342	

Gambar 4. Set Data Skenario 2

Hasil pengujian algoritma menggunakan kedua skenario dan perbandingan hasil dengan penelitian sebelumnya dapat dilihat pada Tabel 1.

Tabel 1. Perbandingan Nilai *Makespan*

Set Data	<i>n</i>	<i>m</i>	Nasr & Elsayed (1990)	Moon & Lee (2000)	Prasetyo <i>et al.</i> (2004)	Pezella <i>et al.</i> (2008)	Gao <i>et al.</i> (2008)	Susanti (2010)	Algoritma Usulan
NE	4	6	18	17	17			17	17
MK-01	10	6				40	40	40	40

5. ANALISIS

Dari Tabel 1 dapat diketahui bahwa algoritma GRASP with *Fixed Threshold* memiliki keandalan dalam menyelesaikan permasalahan penjadwalan *job shop* alternatif *routing*. Selain menggunakan nilai *threshold* sebesar 10%, digunakan juga nilai *threshold* 5% dan 15%. Semua memberikan nilai *makespan* yang sama untuk kedua set data, yang membedakannya adalah pada tahap *local search*. Semakin kecil nilai *threshold* maka alternatif solusi yang diberikan semakin sedikit juga.

Algoritma GRASP with *Fixed Threshold* menghasilkan nilai *makespan* yang sama pada set data NE untuk penelitian Moon dan Lee (2000), Prasetyo *et al.* (2004), dan Susanti (2010), Algoritma GRASP with *Fixed Threshold* menghasilkan nilai *makespan* yang lebih baik sebesar 1 satuan waktu dibandingkan dengan penelitian Nasr dan Elsayed (1990).

Untuk set data MK-01, Algoritma GRASP with *Fixed Threshold* menghasilkan nilai *makespan* yang sama dengan penelitian Pezella *et al.* (2008), Gao *et al.* (2008), dan Susanti

(2010). berdasarkan hasil Skenario 2, maka algoritma usulan memiliki keandalan untuk menyelesaikan permasalahan *job shop* alternatif *routing*.

6. KESIMPULAN

Kesimpulan dari hasil penelitian ini adalah sebagai berikut:

1. Pengembangan algoritma yang diusulkan adalah algoritma *greedy randomized adaptive search procedure with fixed threshold* yang diujikan untuk menyelesaikan masalah penjadwalan *job shop* alternatif *routing* dengan kriteria minimisasi *makespan*.
2. Pengujian skenario menunjukkan bahwa algoritma usulan dapat digunakan untuk menyelesaikan penjadwalan *job shop* alternatif *routing* dengan hasil yang kompetitif.

REFERENSI

Binato, S., Hery, W., Loewestern, D., dan Resende, M. G. C. (2001). *A GRASP for job shop scheduling*, In C. Ribeiro dan P. Hansen (editor), *Essays and Survery on Metaheuristics*, Kluwer Academic Publishers, p. 59-79.

Brandimarte, R. (1993). *Routing and Scheduling in a flexible Job Shop By Tabu Search*. *Annals of Operation Research* 41, p. 157-183.

Dueck, G. dan Scheuer, T. (1990). *Threshold Accepting: A General Purpose Optimization Algorithm Appearing Superior to Simulated Annealing*, *Journal of Computational Physics* 90, p. 161-175.

Feo, T. A. dan Resende, M. G. C. (1995). *Greedy randomized adaptive search procedures*, *Journal of Global Optimization*, 6, p. 109-133.

Gao, Jie, Sun, L. dan Gen, M. (2008). *A Hybrid Genetic and Variable Neighborhood Descent Algorithm for Flexible Job Shop Scheduling Problems*, *Computers & Operations Research* 35, p. 2892-2897.

Nasr, N., dan Elsayed, E. A. (1990). *Job Shop Scheduling with Alternative Machines*, *International Journal of Production Research* 28, No. 9, p. 1595-1609.

Moon, I. dan Lee, J. (2000). *Genetic Algorithm Application to The Job Shop Scheduling Problem with Alternative Routings*, *Brain Korea 21 Logistics Team, Industrial Engineering*, I Pusan National University.

Pezzella, F., Morganti, G. dan Ciaschetti, G. (2008). *A Genetic Algorithm for The Flexible Job shop Scheduling Problem*, *Computer & Operation Research* 35, p. 3202-3212.

Prasetyo, H., Zaini, E., Luis, M. dan Irawan, D. (2004). *Model Penjadwalan Job Shop Alternatif Routing Menggunakan Algoritma Ant Colony System Untuk Meminimisasi Makespan*, *Prosiding Seminar Nasional II Peningkatan Kualitas Sistem Manufaktur dan Jasa*, Yogyakarta, 267-275.

Susanti, N. (2010). *Model Penjadwalan Job Shop dengan Alternatif Routing Menggunakan Algoritma Greedy Randomized Adaptive Search Procedure untuk Minimisasi Makespan*. Itenas.