MODEL PENJADWALAN *NO-WAIT JOB SHOP* MENGGUNAKAN ALGORITMA *VARIABLE NEIGHBOURHOOD DESCENT* DENGAN *THRESHOLD* UNTUK MEMINIMISASI *MAKESPAN*[*]

YONI A. RESPATI, EMSOSFI ZAINI, ARIF IMRAN

Jurusan Teknik Industri
Institut Teknologi Nasional (Itenas) Bandung

Email: adityorespati@gmail.com

ABSTRAK

Permasalahan no-wait job shop didefinisikan sebagai masalah penjadwalan job shop dengan batasan no-wait didalamnya. Batasan no-wait yaitu kondisi dimana antar dua operasi pada satu job harus dikerjakan secara kontinu tanpa adanya jeda waktu. Pada penelitian ini digunakan algoritma variable neighborhood descent dengan threshold untuk meminimisasi makespan. Algoritma usulan menggunakan threshold sebagai batas untuk mendapatkan current solution dimana nilai ini diambil dari makespan terbaik dari setiap iterasi. Set data dari literatur digunakan untuk menguji algoritma. Hasil pengujian menunjukkan bahwa algoritma memberikan hasil yang sama baiknya dengan hasil yang telah dipublikasikan.

***Kata kunci**: penjadwalan, job shop, variable neighborhood descent, threshold, neighbor*

ABSTRACT

No-wait job shop scheduling problems is defined as a job shop scheduling problems with no-wait constraints therein . No-wait constraint is a condition in which between two operations in one job should be done continuously without any lag time. In this study will be used variable neighborhood descent with threshold algorithm to minimize the makespan. Proposed algorithm uses a threshold as a limit to get current solution where the value is taken from the best makespan of each iteration. Several data set from literature are used to test the algorithm. The results show that the algorithm gives equally good results with the results that have been published .

***Keywords**: scheduling, job shop, variable neighborhood descent, threshold, neighbor*

[*] *Makalah ini merupakan ringkasan dari Tugas Akhir 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

1.1 Pengantar

Permasalahan penjadwalan *job shop* yaitu terdapat sejumlah *n* job pada *m* mesin mesin dengan setiap *job* dapat memiliki urutan proses yang berbeda dengan *job* yang lainnya. Setiap *job* memiliki urutan proses dan setiap operasi tersebut harus diproses tanpa adanya gangguan pada mesin-mesin yang telah ditentukan dalam selang waktu operasi tersebut. Tujuan dari teknik penjadwalan *job shop* adalah untuk mendapatkan satu set jadwal dengan waktu penyelesaian yang lebih singkat. Untuk menyelesaikan permasalahan penjadwalan telah dikembangkan beberapa metode, yaitu metode optimisasi yang digunakan untuk mendapatkan solusi yang optimal. Metode optimisasi yang telah digunakan diantaranya oleh Carlier & Pinson (1989) yang mengajukan metode optimisasi berbasis *Branch* & *Bound*. Metode-metode ini sangat efektif untuk mencari solusi jadwal yang optimal namun membutuhkan waktu komputasi yang cenderung lama. Untuk permasalahan *job shop* yang lebih kompleks diperlukan sebuah metode yang lebih cepat dalam memberikan solusi, kemudian dikembangkanlah beberapa metode heuristik.

Beberapa penelitian yang membahas mengenai permasalahan *job shop* dengan metode heuristik, adalah Applegate dan Cook (1991) membahas permasalahan *job shop* dengan menggunakan algoritma *constructive heuristic* berbasis *Branch* & *Bound* dan prosedur *shifting bottleneck* untuk meminimasi *makespan*. Chu et. al. (1998) mengusulkan heuristik berbasis *disjunctive graph* dalam menyelesaikan permasalahan *job shop*. Adapun penelitian-penelitian lain yang mengunakan teknik *global search heuristic* (metaheuristik) untuk menyelesaikan permasalahan *job shop*, salah satunya adalah Dell'Amico dan Trubian (1993) yang mengembangkan algoritma *tabu search* untuk menyelesaikan masalah *job shop* dengan kriteria minimasi *makespan*.

Penelitian-penelitian yang disebutkan di atas mengizinkan adanya waktu tunggu antar operasi dalam satu *job*. Kondisi *no-wait* merupakan suatu kondisi dimana operasi-operasi yang terdapat dalam satu *job* harus dikerjakan terus-menerus tanpa adanya jeda atau interupsi. Kondisi ini biasanya dikarenakan oleh proses lingkungan ataupun karakteristik dari *job* tersebut.

Batasan *no-wait* membuat permasalahan menjadi rumit, dimana ketidaktepatan penjadwalan dapat mengakibatkan mundurnya *makespan*. Beberapa penelitian untuk mendapatkan penyelesaian NWJSS tersebut telah dilakukan, misalnya dengan menggunakan *Genetic-Algorithm-Simulated Annealing* (Schuster & Framinan, 2003) dan *Hybrid Tabu Search Algorithm* (Bożejko dan Makuchowski, 2008). Kemudian Anggraeni (2010) mengusulkan algoritma *constructive heuristic* untuk meminimisasi *makespan*. Metode diatas mendapatkan hasil *makespan* yang tidak optimal, namun ada pula yang mendapatkan hasil *makespan* yang optimal namun dengan waktu komputasi yang relatif lama.

1.2 Identifikasi Masalah

Penelitian ini membahas masalah penjadwalan NWJSS. Sistem NWJS dapat didefinisikan sebagai penjadwalan *n* job yang terdiri dari beberapa operasi dimana tiap operasi dalam satu *job* harus dilakukan terus-menerus tanpa diperbolehkan adanya interupsi. Metode yang coba dikembangkan dalam penelitian ini adalah algoritma *variable neighborhood descent* dengan *threshold*. Pembangkitan solusi inisial dilakukan dengan aturan SPT (*Shortest Proccesing Time*) dan kriteria performansi yang digunakan adalah *makespan*. Pada penelitian ini

digunakan metode *variable neighborhood descent*. *Variable neighborhood descent* menelusuri secara sistematis perubahan yang terjadi pada struktur *neighborhood*, dalam penjadwalan *neighbor* adalah solusi jadwal atau set jadwal lain dalam satu permasalahan penjadwalan. Tujuan dari penelitian ini adalah untuk menghasilkan sebuah algoritma penjadwalan *no-wait job shop* dengan menggunakan algoritma *variable neighborhood descent* dengan *threshold* untuk meminimisasi *makespan.*

2. STUDI LITERATUR

2.1 Konsep Dasar Penjadwalan

Menurut Baker (1974), tujuan penjadwalan adalah sebagai berikut:

1. Meningkatkan produktifitas mesin, yaitu dengan mengurangi waktu mesin menganggur.
2. Mengurangi persediaan barang setengah jadi dengan jalan mengurangi jumlah rata-rata pekerjaan yang menunggu dalam antrian suatu mesin karena mesin tersebut sibuk.
3. Mengurangi keterlambatan suatu pekerjaan. Dengan metoda penjadwalan maka keterlambatan ini dapat dikurangi, baik waktu maupun frekuensi.

2.1.1 Klasifikasi Penjadwalan

1. Berdasarkan jumlah mesin yang digunakan, dapat berupa penjadwalan pada mesin tunggal atau penjadwalan pada mesin jamak.
2. Pola kedatangan job, dapat berupa pola kedatangan statis atau pola kedatangan dinamis. Pola kedatangan statis yaitu dimana sejumlah pekerjaan datang secara teratur/periodik. Sedangkan pola kedatangan dinamis yaitu dimana job datang secara terus-menerus (kontinu) dan tidak menentu.
3. Sifat informasi yang diterima, dapat berupa deterministik atau stokastik.
4. Pola aliran proses, dapat berupa pola aliran proses *flow shop* atau pola aliran proses *job shop*. Pola aliran *flow shop* ditandai dengan semua pekerjaan mengikuti lintas yang sama dari satu mesin ke mesin yang lain melalui urutan mesin yang sama. Pada pola aliran *job shop*, aliran proses dalam shop tidak mempunyai pola yang umum.

2.1.2 Istilah dalam Penjadwalan

1. Waktu proses (t_i)
2. *Ready Time* (R_i)
3. *Due Date* (d_i)
4. *Completion Time* (C_i)
5. *Flow Time* (F_i)
6. *Makespan* (MS)
7. *Lateness* (L_i)
8. *Tardiness* (T_i)
9. *Slack Time* (S_i)

2.2 Sistem Penjadwalan *Job Shop*

Dalam permasalahan *job shop*, *n job* harus diproses pada *m* mesin. Setiap *job* mempunyai beberapa operasi yang harus diproses pada mesin yang berbeda. Kriteria performansi yang tepat sangat bergantung pada tujuan dan kebijakan manajemen, sehingga dalam penjadwalan *job shop* diperlukan *input* berupa jumlah *job*, jumlah operasi dalam tiap *job*, dan urutan proses beserta mesin yang akan digunakan.

Tabel 1. Matriks Routing *Routing Job Shop* dan Matriks Waktu Proses; (a) Matriks *Routing*; (b) Matriks Waktu Proses

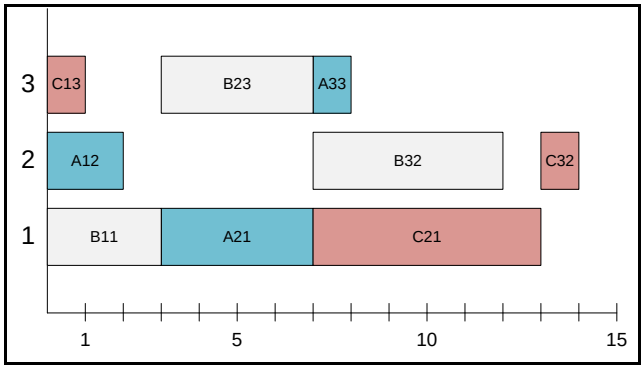
(a)

Job	Operasi		
	1	**2**	**3**
A	M2	M1	M3
B	M1	M3	M2
C	M3	M1	M2

(b)

Job	Operasi		
	1	**2**	**3**
A	2	4	1
B	3	4	5
C	1	6	1

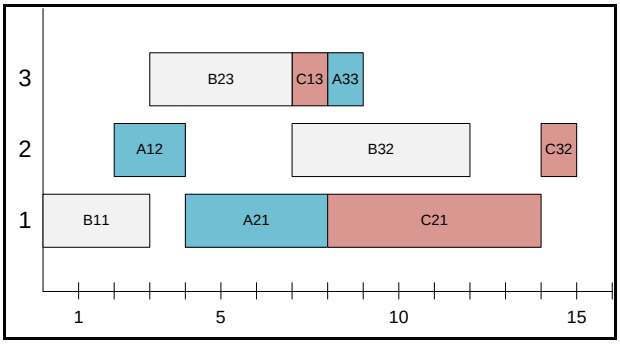
Setelah *input* masing-masing *job* didefinisikan, proses selanjutnya adalah penugasan operasi dari *job* pada tiap mesin. Suatu jadwal umumnya direpresentasikan ke dalam *Gantt-chart*. *Gantt-chart* digambarkan dalam bentuk susunan blok-blok batang yang analog dengan waktu penyelesaian pekerjaan-pekerjaan tersebut. Contoh *gantt-chart* untuk penjadwalan 3 job dan 3 mesin dapat dilihat pada Gambar 1.



Gambar 1. *Gantt-chart* Hasil Penjadwalan *Job Shop* untuk 3 Job dan 3 Mesin

2.3 Sistem Penjadwalan *No-Wait Job Shop* (NWJS)

Perbedaan *no-wait job shop* dengan *job shop* adalah pada *no-wait job shop* setiap operasi (*Oij*) pada masing-masing *job* harus diproses secara terus menerus tanpa adanya interupsi. Data yang dibutuhkan dalam penjadwalan kondisi *no-wait* tidak berbeda dengan permasalahan *job shop*. Sebagai contoh Tabel 1 akan digunakan untuk melakukan penjadwalan dalam kondisi no-wait. Hasil penjadwalan kondisi *no-wait* dengan waktu proses dan *routing* pada Tabel 1 dapat dilihat pada Gambar 2.



Gambar 2. *Gantt-chart* Hasil Penjadwalan *No-Wait Job Shop* untuk 3 Job dan 3 Mesin

Gambar 1 dan Gambar 2 merupakan hasil penjadwalan dengan menggunakan set data yang sama, namun memiliki gantt chart yang berbeda. Hal ini dapat dilihat pada pengerjaan *Job C*. Pada kasus *job shop*, proses pertama *Job C* pada mesin 3 di kerjakan pada t = 0,

sedangkan pada permasalahan *No-wait job shop* dilakukan pada $t = 0$. Sedangkan pada *no-wait job shop*, dikerjakan pada $t = 7$. Hal ini menjelaskan bahwa ketika operasi pertama dikerjakan pada $t = 0$, maka akan menimbulkan jeda antara operasi pertama dan kedua. Pada kondisi *no-wait* jeda antar operasi ini tidak diperkenankan.

2.4 *Heuristic* dan *Metaheuristic*

Istilah heuristik berasal dari bahasa Yunani yang berarti mencari atau menemukan. Metode heuristik didapatkan melalui dasar akal sehat, logika, dan pengalaman. Pengertian heuristik menurut Reeves dan Beasley (1993) adalah suatu teknik yang digunakan untuk mencari solusi mendekati solusi optimal dan tidak dapat dijamin kelayakan dan keoptimalannya atau dapat digunakan untuk menyatakan seberapa dekat keoptimalan tertentu dari solusi yang optimal.

Menurut Osman dan Kelly (1996) metaheuristik adalah proses pengulangan generasi yang memadu heuristik sebelumnya dengan mengkombinasikan konsep cerdas yang berbeda untuk menyelidiki dan memanfaatkan ruang pencarian dengan menggunakan strategi belajar untuk mendapatkan struktur informasi dalam rangka secara efisien menemukan solusi yang mendekati optimal.

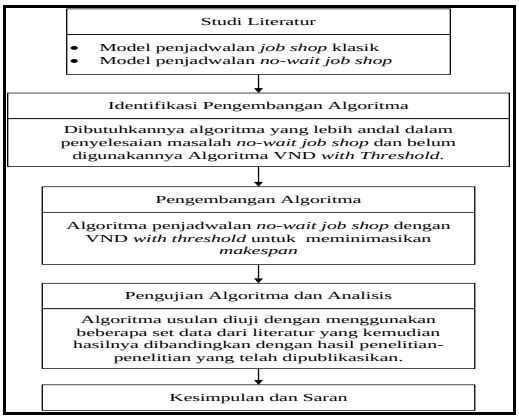
2.5 *Variable Neighborhood Search* (VNS)

Ide dasar dari VNS ini ialah memanfaatkan perubahan struktur yang terjadi dalam *neighborhood*. Hal tersebut berfungsi untuk melakukan pencarian solusi ketika pencarian solusi terjebak dalam minimum lokal. *Variable neighborhood descent* (VND) merupakan salah satu pengembangan konsep dasar yang diambil dari VNS. Perbedaan antara VNS dan VND terletak pada perubahan neighborhood, VND melakukan perubahan *neighborhood* secara deterministik sedangkan VNS melakukan perubahan *neighborhood* secara acak (random).

Pada tahap awal VND dipilih satu set struktur *neighborhood* yang didefinisikan sebagai jadwal inisial melalui tahap konstruksi. Kemudian struktur ini di-set sebagai *current solution*. Kemudian dilakukan pencarian solusi baru melalui proses *local search*. Jika pada saat pencarian ini ditemukan solusi yang lebih baik dari *current solution*, maka solusi tersebut akan menggantikan *current solution*. Proses ini terus berulang sampai semua kemungkinan *neighborhood* dan tidak ditemukan lagi solusi yang lebih baik daripada *current solution*.

3. METODOLOGI PENELITIAN

Urutan proses dan langkah-langkah yang dilakukan pada penelitian ini dapat dilihat pada Gambar 3.



Gambar 3. Langkah-Langkah Penelitian

3.1 Studi Literatur

Langkah pertama yang dilakukan adalah mempelajari literatur yang akan digunakan sebagai dasar ilmu untuk melakukan penelitian ini.

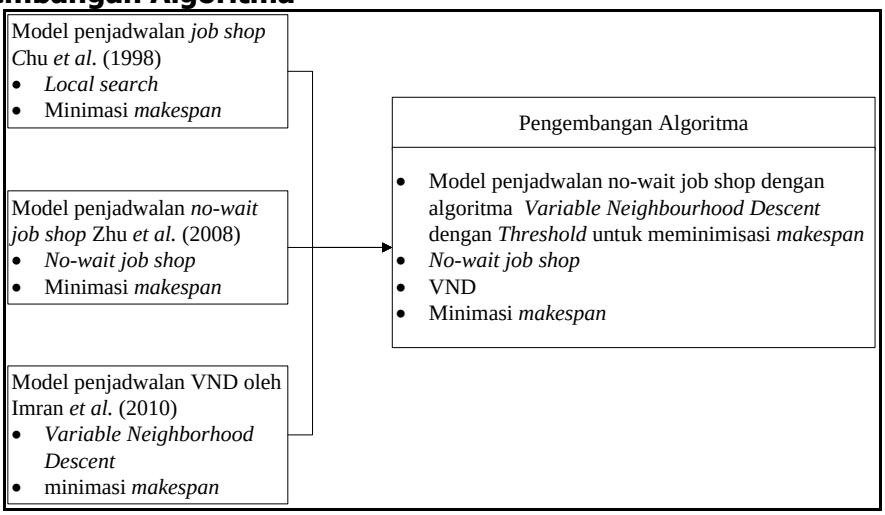
1. Model Penjadwalan *Job Shop*
2. Model Penjadwalan *No-Wait Job Shop* (NWJS)

3.2 Identifikasi Kebutuhan Algoritma

Algoritma *variable neighborhood descent* (VND) menelusuri secara sistematis perubahan struktur *neighborhood*. Penggunaan algoritma VND untuk permasalahan *job shop* dengan alternatif *routing* untuk minimasi *makespan* telah dikembangkan oleh Imran *et al*. (2010).

Pada kondisi *no-wait* terdapat batasan dalam algoritma. Zhu *et al*. (2008) mengembangkan algoritma *Complete Local Search With Limited Memory* untuk menyelesaikan penjadwalan NWJS.
Berdasarkan hal-hal tersebut, dianggap perlu untuk dilakukan pengembangan algoritma *variable neighborhood descent* dengan *threshold* didasarkan pada penelitian-penelitian diatas. Digunakan beberapa model dasar yang akan dintegrasikan untuk membentuk sebuah algoritma baru dalam menyelesaikan permasalahan NWJS.

3.3 Pengembangan Algoritma



Gambar 4. Skema Pengembangan Algoritma

3.4 Pengujian Algoritma dan Analisis

Pengujian algoritma dilakukan dengan menggunakan set data dari literatur yang ada. Set data yang digunakan pada tahap pengujian algoritma adalah set data dari penelitian Framinan dan Schuster (2006), Zhu *et al*. (2008), Pan & Huang (2009), dan Fisher dan Thompson (1963).

Berdasarkan hasil dari pengujian set data di atas kemudian dilakukan analisis mengenai solusi serta proses komputasinya.

3.5 Kesimpulan

Setelah didapatkan hasil dari seluruh rangkaian penelitian kemudian ditarik kesimpulan serta saran yang diajukan kepada peneliti-peneliti selanjutnya untuk pengembangan yang lebih lanjut.

4. PENGEMBANGAN ALGORITMA

4.1 Karakteristik Perusahaan

Kondisi dimana tidak diperbolehkan adanya jeda antar dua operasi dalam satu job (no-wait) sudah banyak terdapat pada beberapa perusahaan. Pada beberapa perusahaan diharuskan untuk meminimisasi atau bahkan menghilangkan transportasi diantara dua operasi. Dengan menciptakan sebuah sistem perpindahan barang setengah jadi maka waktu transportasi dapat diabaikan. Algoritma yang dikembangkan dirancang untuk beberapa kasus dimana waktu transportasi dapat diabaikan karena pada algoritma yang dikembangkan waktu

4.2 Pengembangan Algoritma

Notasi-notasi yang digunakan pada pengembangan algoritma penjadwalan no-wait job shop dengan menggunakan algoritma variable neighborhood descent dengan threshold adalah:

i = indeks operasi (i = 1, 2, 3, …, q)
j = indeks mesin (j = 1, 2, 3, …, m)
O_{ij} = operasi ke-i mesin ke-m (A12 = operasi ke-1 *job* A di mesin 2)
P_i = Total waktu proses job ke-i
MS_{Th} = batas *makespan threshold*
S = himpunan jadwal fisibel (MS ≤ MSTh)
Y = jumlah jadwal fisibel (yang terdapat pada himpunan S)
LK = lintasan kritis
W = himpunan pasangan *job* yang akan dipertukarkan
X = jumlah pasangan *job* yang akan dipertukarkan
q = jumlah operasi di setiap *job*

Algoritma algoritma *variable neighborhood descent* dengan *threshold* untuk meminimasi *makespan* adalah sebagai berikut:

Tahap Konstruksi (pembentukan jadwal inisial):

Langkah 1:
Input data:
1. Jumlah *job* (n)
2. Jumlah mesin (m)
3. Matriks *routing*
4. Matriks waktu proses

Langkah 2:
Hitung total waktu proses Pi
Pi = $\sum_{j=1}^{1} t_{ij}$
Langkah 3:
Jadwalkan *job* dengan aturan SPT.
Langkah 4:
Lakukan proses *left shift* lalu tampilkan *gantt chart* dan catat *makespan*-nya sebagai acuan untuk dijadikan batas makespan *threshold*.

Tahap *Local Search*:

Langkah 5:
Hitung nilai MS_{Th} dengan persamaan MS_{Th} = $MS_{terbaik}$ + ($MS_{terbaik}$ x % *Threshold*).
Langkah 6:
Tentukan S, S = himpunan jadwal fisibel atau jadwal dengan makespan di bawah batas *makespan threshold*.
Periksa apakah S = {} ?

Jika ya, maka solusi jadwal terbaik adalah semua jadwal dengan nilai *makespan* di bawah batas *makespan threshold* sebelumnya.
Jika tidak, maka lanjutkan ke langkah 7.
Langkah 7:
Ambil salah satu jadwal yang ada di dalam Himpunan *S* lalu tentukan lintasan kritis kemudian pilih salah satu lintasan yang memberikan pertukaran *job* terbanyak. Nyatakan lintasan kritis sebagai LK.
Langkah 8:
Set $X' = 0$
Langkah 9:
Tentukan himpunan pasangan *job* yang dipertukarkan sebagai *W*, hitung jumlah pasangan *job* yang dipertukarkan sebagai *X*.
Langkah 10:
Tentukan dua *job* yang akan dipertukarkan, misal: *Job A* dan *Job B*, dengan kriteria kedua *job* saling bersebelahan satu sama lain, lalu tampilkan *gantt chart* proses *exchange*.
$X' = X' + 1$
Langkah 11:
Periksa apakah urutan sudah pernah terjadi ?
Jika ya, maka lanjutkan ke langkah 15
Jika tidak, maka lanjutkan ke Langkah 12
Langkah 12:
Bila memungkinkan lakukan proses *insert* lalu tampilkan *gantt chart* dan catat *makespan*.
Langkah 13:
Periksa apakah $MS \leq MS_{Th}$?
Jika ya, lalu lanjutkan ke langkah 14
Jika tidak, lanjutkan ke langkah 15
Langkah 14:
Masukkan jadwal ke dalam Himpunan *S*.
Langkah 15:
Periksa apakah $X' = X$?
Jika ya, keluarkan jadwal yang telah dilakukan pertukaran dari *S* lalu kembali ke langkah 5.
Jika tidak, kembali ke langkah 10.

5. PENGUJIAN ALGORITMA DAN ANALISIS

5.1 Tahapan Pengujian Algoritma

Set data yang digunakan yaitu:
1. Framinan dan Schuster (2006) yang selanjutnya disebut Set Data 1.
2. Zhu *et al.* (2008) yang selanjutnya disebut Set Data 2.
3. Pan & Huang (2009) yang selanjutnya disebut Set Data 3.
4. Fisher dan Thompson (1963) yang selanjutnya disebut Set Data 4.

5.2 Pengujian Algoritma

Keempat set data yang digunakan seperti yang telah disebutkan sebelumnya ditampilkan pada Tabel 2, Tabel 3, Tabel 4, dan Tabel 5 berikut.

Tabel 2. Matriks *Routing* dan Matriks Waktu Proses Set Data 2; (a) Matriks *Routing*; (b) Matriks Waktu Proses

(a)

Job	**Operasi**				
	1	**2**	**3**	**4**	**5**
A	M4	M2	M3	M5	M1
B	M5	M4	M2	M3	M1
C	M1	M2	M3	M4	M5

(b)

Job	**Operasi**				
	1	**2**	**3**	**4**	**5**
A	5	6	4	1	2
B	8	8	7	5	4
C	1	6	5	7	4

Tabel 3. Matriks *Routing* dan Matriks Waktu Proses Set Data 3; (a) Matriks *Routing*; (b) Matriks Waktu Proses

(a)

Job	**Operasi**			
	1	**2**	**3**	**4**
A	M1	M4	M3	M2
B	M1	M3	M4	M2
C	M2	M4	M3	M1
D	M4	M2	M1	M3
E	M3	M2	M4	M1
F	M3	M4	M2	M1

(b)

Job	**Operasi**			
	1	**2**	**3**	**4**
A	7	10	7	4
B	10	8	5	7
C	3	2	1	3
D	8	1	8	1
E	3	4	7	3
F	1	6	6	10

Tabel 4. Matriks *Routing* dan Matriks Waktu Proses Set Data 4; (a) Matriks *Routing*; (b) Matriks Waktu Proses

(a)

Job	**Operasi**					
	1	**2**	**3**	**4**	**5**	**6**
A	M3	M1	M2	M4	M6	M5
B	M2	M3	M5	M6	M1	M4
C	M3	M4	M6	M1	M2	M5
D	M2	M1	M3	M4	M5	M6
E	M3	M2	M5	M6	M1	M4
F	M2	M4	M6	M1	M5	M3

(b)

Job	**Operasi**					
	1	**2**	**3**	**4**	**5**	**6**
A	1	3	6	7	3	6
B	8	5	10	10	10	4
C	5	4	8	9	1	7
D	5	5	5	3	8	9
E	9	3	5	4	3	1
F	3	3	9	10	4	1

Hasil dari pengujian algoritma dapat dilihat pada Tabel 5.

Tabel 5. Perbandingan *Makespan* Algoritma Usulan

Set Data	*N*	*m*	Makespan				
			Framinan & Schuster (2006)	Zhu et al. (2008)	Pan Huang (2009)	Anggraeni, W. (2010)	Algoritma Usulan
Set Data 1	3	3	16			15	15
Set Data 2	3	3		33		33	33
Set Data 3	6	4			56	49	49
Set Data 4	6	6	73	73	73	73	73

Hasil dari tahap pengujian algoritma dengan menggunakan keempat set data dari penelitian yang telah dipublikasikan menghasilkan *makespan* yang sama dibandingkan penelitian terakhir yang dilakukan oleh Anggraeni (2010).

6. KESIMPULAN

1. Algoritma *variable neighborhood descent* dengan *threshold* untuk meminimisasi *makespan* dapat digunakan untuk mendapatkan set jadwal pada permasalahan *no-wait job shop.*

2. Algoritma *variable neighborhood descent* dengan *threshold* untuk meminimisasi *makespan* mendapatkan hasil *makespan* sebesar 15 dan 49 satuan waktu pada set data 1 dan 3. Kedua hasil ini lebih baik dibandingkan dengan algoritma awal pada masing-masing penelitian yaitu sebesar 16 dan 56 satuan waktu. Sedangkan pada set data 2 dan 4 algoritma usulan mendapatkan hasil *makespan* yang sama yaitu sebesar 33 dan 73 satuan waktu.
3. Untuk membandingkan keandalan algoritma dengan penelitian terakhir dapat digunakan set data lainnya.
4. Proses perhitungan manual dengan menggunakan algoritma usulan memakan waktu komputasi yang cukup lama sehingga dibutuhkan bantuan aplikasi *software* untuk membantu proses perhitungan.

REFERENSI

Anggraeni, W., 2010, Model Penjadwalan *No-Wait Job Shop* Menggunakan Algoritma *Constructive Heuristic* Untuk Meminimisasi Makespan: Tugas Sarjana – Program Studi Teknik Industri, Institut Teknoogi Nasional, Bandung.

Applegate, D. dan Cook, W., 1991, A Computational Study of the Job-Shop Scheduling Problem, ORSA 3, No. 2, 149-156.

Baker, K. R., 1974, Introduction to Sequencing and Scheduling, John Wiley & Son, New York.

Bozejko, W. dan Makuchowski, M., 2008, A Fast Hybrid Tabu Search Algorithm for the No-Wait Job Shop Problem, *Computer & Industrial Engineering*, doi:10.1016/j.cie.2008.19.023, 1-8.

Chu, C., Porth,, M. J., dan Wang, O., 1998, Improving Job Shop Schedules Through Critical Pairwise Exchange, International Journal of Production Research 36, No. 3, 683-694.

Dell' Amico, M., dan Trubian, M., 1993, Applying Tabu Search for the Job Shop Scheduling Problem, *Annals of Operation Research* 41, 231-252.

Fisher, H., dan Thompson, G. L., 1963, Probabilistic Learning Combinations of Local Job Shop Sched:uling Rules, In: Industrial Scheduling, Englewood Cliffs, NJ: Prentice-Hall, 225-251

Framinan, M. J., dan Schuster, C., 2006, An Enhanced Timetabling Procedure for the No-Wait Job Shop Problem: A Complete Local Search Approach, Computers and Operations Research 331, 1200-1213

Imran, A., Luis, M., Zaini, E., dan Beny, B., Job Shop Scheduling Problem with Alternative Routings using Variable Neighbourhood Descent to Minimize Makespan. Proceedings of the International Conference on Industrial Engineering and Business Management. Yogyakarta. 2010.

Mladenovic, N., dan Hansen, P., 1997, Variable Neighborhood Search, *Computer Operation Research* 24, 1097-1100.

Osman, L. H. dan Kelly, J. P., 1996, Meta-Heuristic: An Overview. In Osman, L. H., Kely, J. P., editors, *Meta-Heuristic*: *Theory and Application*. Kluwer Academic Publishers, Massachusetts, 1-21.

Pan, H., dan Huang, C., 2009, A Hybrid Genetic Algorithm for No-Wait Job Shop Scheduling Problems, Experts System with Applications 36, 3800-5806
Reeves, C. R., dan Beasley, J. E., 1993, Introduction. In Reeves, C. R., editor, *Modern Heuristic Techniques for Combinatorial Problems*. Blackwell Scientific Publications, Oxford, 1-19.

Schuster, J. C., dan Framinan, M. J., 2003, Approximative Procedure for No-Wait Job Shop Scheduling, *Operation Research Letters* 31, 308-318.

Zhu, J. Li, X., dan Wang, O., 2008, Complete Local Search With Limited Memory Algorithm for No-Wait Job Shops to Minimize Makespan, European Journal of Operational Research, doi: 10.1016/j.ejor.2008.09.015, 1-9.