PERBANDINGAN PERFORMANSI ALGORITMA *CROSS ENTROPY* (*CE*) DAN ALGORITMA *PARTICLE SWARM OPTIMIZATION* (*PSO*) PADA PENYELESAIAN PERMASALAHAN *FLOWSHOP SCHEDULING*

DINAMIKA TEKNIK
Vol. V, No. 2
Juli 2011
Hal 53 - 63

Maria Krisnawati
Dosen Fakultas Teknik Universitas Stikubank Semarang
Email: mariakrisnawati@gmail.com

Abstrak

Penjadwalan flowshop yaitu penjadwalan proses produksi dari n-job yang memiliki urutan proses produksi yang sama. Terdapat berbagai macam teknik penyelesaian pada masalah ini, salah satunya adalah dengan algoritma cross entropy. Modifikasi algoritma cross entropy dengan menurunkan jumlah sample untuk penyelesaian NP-hard problem pada permasalahan crew scheduling menunjukkan performansi yang relatif bagus. Untuk itu, modifikasi algoritma cross entropy dapat memberikan alternatif baru dalam penyelesaian masalah penjadwalan flowshop. Penelitian ini ditujukan untuk menilai performansi cross entropy dengan pengurangan jumlah sampel untuk menyelesaikan masalah penjadwalan flowshop m-mesin untuk n-job Hasil dari uji coba dengan jumlah job yang berbeda – beda dan jumlah mesin yang sama , menghasilkan bahwa solusi yang dihasilkan dari algoritma cross entropy dengan penurunan jumlah sampel memberikan hasil yang relatif baik pada problem yang yang berukuran besar bila dibandingkan dengan algoritma PSO dan waktu komputasi yang diperlukan untuk penyelesaian masalah lebih singkat jika dibandingkan dengan algoritma cross entropy.

Keywords *: Flowshop Scheduling, tardiness, particle swarm optimization, cross entropy.*

Pendahuluan

Penjadwalan produksi *n*-job dengan menggunakan *m*-mesin merupakan masalah optimasi yang tergolong *NP-hard (Non Polynomial-hard) problem* dan *combinatorial* problem yang dihadapi sebagian besar perusahaan. Mereka akan cenderung mencari waktu produksi yang minimum supaya produktivitas perusahaan menjadi semakin besar. Untuk mencapai waktu produksi minimum sekaligus bisa memenuhi batasan penggunaan mesin yang tidak boleh melakukan job bersamaan, diperlukan teknik solusi yang efektif dan efisien untuk penjadwalan produksi. (Lathifi, 2010). Penjadwalan produksi flowshop adalah satu jenis permasalahan dalam scheduling problem seperti yang disebutkan diatas. Permasalahan *flowshop scheduling* telah dikelompokkan oleh banyak peneliti ke dengan asumsi klasik yang beragam dan

berbeda fungsi dan tujuan dengan menerapkan berbagai teknik optimasi. Masalah flowshop reguler terdiri dari dua elemen utama: (1) kelompok *m*-mesin dan (2) seperangkat pekerjaan (*job*) *n* untuk diproses pada kelompok mesin. Permasalahan penjadwalan umumnya adalah menentukan urutan dan waktu pemrosesan satu pekerjaan pada mesin, dengan tujuan yang telah ditetapkan. (Hejazi, 2005).

Beberapa metode atau pendekatan sudah telah dilakukan sebelumnya sebagai alternatif pemecahan masalah flowshop scheduling untuk mendapatkan waktu produksi minimum. Berbagai penelitian itu antara lain : *Lagrangian relaxation algorithms* (Tang, 2006), *Genetic algorithma* (Chen, 2000), *Particle swarm optimization* (Tasgetiren, 2007), *Ant Colony Optimization* (Rajendran, 2004), *Hybrid Tabu search* dan *Cross Entropy* (Lathifi, 2010), dsb. *Cross Entropy* (CE) diketahui memiliki performansi yang cukup baik untuk menyelesaikan *NP-Hard Problem*. Penelitian yang menggunakan *Cross Entropy* antara lain Rubinstein menggunakan *cross entropy* untuk menyelesaikan permasalahan *combinatorial* dan *continuous optimization* (1999), Laguna menggunakan *cross entropy* pada *max-cut problem* (2009), Caserta menggunakan *cross entropy* dalam *multi-item capacitated lot-sizing problem with setup times* (2009), Santosa menggunakan *cross entropy* dalam *no-wait job-shop scheduling* (2011), Krisnawati (2011) menggunakan *cross entropy with decreasing sample* dalam penjadwalan kru pada maskapai penerbangan.

Pada penelitian yang dilakukan Krisnawati (2011), *Cross entropy with decreasing sample* menunjukkan performansi yang baik dalam menyelesaikan permasalahan *NP Hard problem* dari segi solusi yang dihasilkan dan waktu komputasi yang relatif singkat. Untuk itu, *cross entropy with decreasing sample* dapat menjadi alternatif baru untuk penyelesaian permasalahan *flowshop scheduling* yang juga merupakan permasalahan *NP Hard problem*. Sebagai *benchmarks*, dilakukan penyelesaian masalah dengan algoritma lain, yaitu dengan *Cross Entropy* (tanpa penurunan jumlah sampel) dan Algoritma *Particle Swarm Optimization* (PSO) untuk melihat seberapa efektif penurunan jumlah sampel yang dilakukan dan waktu penyelesaian permasalahan *flow shop scheduling*.

Pada penelitian ini, ketiga algoritma dibandingkan dari parameter total waktu produksi (makespan) minimum yang dihasilkan dan waktu komputasi yang dibutuhkan. Pada bab 2 akan dibahas mengenai tinjauan pustaka yang berkaitan dengan penelitian. Pada bab 3 akan dibahas mengenai algoritma penyelesaian permasalahan *flowshop scheduling* menggunakan algoritma *cross entropy,* algoritma *cross entropy with decreasing sample* dan algoritma *Particle Swarm Optimization*. Analisa dan percobaan diberikan pada bab 4. Kesimpulan dan saran untuk penelitian selanjutnya akan dibahas pada bab 5. Pada penelitian ini kita membandingkan penyelesaian permasalahan *flowshop scheduling* dengan algoritma *cross entropy,* algoritma *cross entropy with decreasing sample* dan algoritma *Particle Swarm Optimization* dari segi waktu penyelesaian dan solusi optimal yang dihasilkan.

Tinjauan Pustaka

Penjadwalan *Flowshop*

Menurut Ponnambalam (2001), penjadwalan produksi dapat diartikan sebagai pengalokasian sumber daya untuk mengerjakan operasi-operasi tertentu dengan tujuan memperoleh jadwal produksi yang optimal. Dalam penjadwalan produksi yang dimaksud sebagai operasi adalah job, sedangkan yang dimaksud dengan sumber daya adalah mesin. Sehingga permasalahan penjadwalan produksi dapat diartikan sebagai proses mengurutkan job-job pada mesin-mesin yang berbeda dalam suatu unit produksi untuk mencapai kondisi yang optimal.

Penjadwalan flowshop merupakan penjadwalan yang dilakukan berdasarkan suatu aliran produksi, yang mana mesin – mesin yang ada disusun sesuai dengan urutan proses produksinya (seri) dan setiap job harus memenuhi urutan mesin yang sama. Penjadwalan flowshop dapat disebut dengan *permutation flowshop* apabila urutan job yang ada pada tiap mesin tidak berubah.

Proses penjadwalan *flowshop* ini memiliki beberapa asumsi antara lain :

- Proses produksi dari masing – masing job sudah diketahui
- Tidak terdapat *pre – emption* (interupsi untuk pengerjaan produk lain di tengah – tengah pengerjaan suatu produk)

- Setiap *job* memerlukan m mesin dan setiap proses memerlukan mesin yang berbeda.
- Waktu *set-up* bersifat independent dan termasuk dalam waktu proses
- Semua *job* mempunyai *ready time* yang sama. (Laha, 2008)

Umumnya pada sistem produksi yang bersifat *flowshop*, terdiri dari beberapa mesin (*m*) dan mempunyai sejumlah job yang harus dikerjakan (*n*) serta waktu proses per unit job *j* pada mesin *i*, $t(i,j)$ untuk $i = 1,\ldots, m$; $j = 1, \ldots n$, maka :

$$t_p(i,j) = t(i,j) \times d_j \tag{1}$$

Dengan d_j adalah jumlah permintaan job *j* dan $t_p(i,j)$ adalah waktu total proses (sesuai demand) job *j* pada mesin *i*

$t_p(i,j) =$ total waktu untuk job j pada mesin i
$t_c(i,j) =$ total waktu proses untuk job j pada mesin i
J_j = Job ke $-j$ M_i = mesin ke $-i$

Proses komputasi untuk totak waktu semua job:

For $i = 1,2, \ldots, m$ and $j = 1,2, \ldots, n$

$$\begin{aligned} t_c(M_1,J_1) &= t_p(M_1,J_1) \\ t_c(M_i,J_1) &= t_c(M_{i-1},J_1)+t_p(M_i,J_1) \\ t_c(M_1,J_j) &= t_c(M_1,J_i)+t_p(M_1,J_j) \\ t_c(M_i,J_j) &= max\{t_c(M_{i-1},J_j), t_c(M_i,J_{j-1})\}+t_p(M_i,J_j) \end{aligned} \tag{2}$$

Sehingga didapat :

$$\text{Makespan} = t_c(M_m, J_n) \tag{3}$$

$$\text{total flow time} = \sum_{j=1}^{n} t_c(M_m,J_j) \tag{4}$$

Algoritma *Cross Entropy*

Algoritma yang diusulkan dalam menyelesaikan permasalahan *flowshop scheduling* pada penelitian ini adalah algoritma *cross entropy* dan algoritma *particle swarm optimization*. Algoritma *Cross Entropy* (CE) pada prinsipnya adalah menggunakan data sampel elite untuk menentukan parameter baru yang akan digunakan untuk membangkitkan populasi baru yang lebih mendekati solusi. Sampel elite adalah berapa persen dari sampel yang kita pilih untuk memperbaiki atau

mengupdate parameter *p* yang digunakan. Pada penelitian ini, problem yang ada diselesaikan dengan tiga algoritma. Salah satu algoritma tersebut adalah modifikasi algoritma *cross entropy* dengan *decreasing sample*. Algoritma ini sama dengan algoritma *cross entropy*, hanya saja pada setiap iterasi terjadi pengurangan jumlah sampel yang dibangkitkan untuk iterasi selanjutnya (Krisnawati, 2011).

Particle Swarm Optimization

PSO adalah salah satu teknik optimasi yang didasarkan pada metafora sosial interaksi dan komunikasi seperti kelompok burung atau ikan. Model ini akan disimulasikan dalam ruang dengan dimensi tertentu dengan sejumlah iterasi sehingga di setiap iterasi, posisi partikel akan semakin mengarah ke target yang dituju (minimasi atau maksimasi fungsi). Ini dilakukan hingga maksimum iterasi dicapai atau bisa juga digunakan kriteria penghentian yang lain.

Algoritma penyelesaian permasalahan *flowshop Scheduling*

Algoritma *Cross Entropy*

Algoritma *cross entropy* untuk permasalahan *flowshop scheduling* adalah :

(i) Set initial parameter :

- *Set percentile of elite sample*, ρ ;
- *number of population npop*;
- *generation number*, *t*;
- *maximum generation*, t_{max}

(ii) Pembangkitan sampel

a. Untuk iterasi awal ($t = 0$)

Untuk iterasi awal dibangkitkan sejumlah solusi yang berupa urutan job *n* sejumlah *npop* berdasarkan parameter $\widehat{p_0}$. Setiap elemen dari matrik parameter $\widehat{p_0}$ yang berukuran *m* x *m* mempunyai probabilitas sukses masing – masing sebesar 1/*m*, dengan *m* adalah jumlah mesin. Pada iterasi awal semua *job* mempunyai probabilitas sukses yang sama untuk ditempatkan pada suatu urutan tertentu. Solusi ukuran job *n* yang

berukuran 1 x *m* dibangkitkan sebanyak *npop* (untuk *npop* = 1000, 2000, 3000 dan 4000).

b. Untuk iterasi selanjutnya ($t \neq 0$)

Pembangkitan sampel untuk iterasi selanjutnya bukan lagi berdasarkan $\widehat{p_0}$ melainkan berdasarkan probabilitas sukses $\widehat{p_t}$, dimana $\widehat{p_t}$ adalah parameter yang telah diupdate pada setiap iterasi ke – *t* menggunakan data sample elite. Solusi ukuran job *n* yang berukuran 1 x *m* dibangkitkan sebanyak *npop* (untuk *npop* = 1000, 2000, 3000 dan 4000).

(iii) Pembaharuan Parameter *p0*

Hitung nilai *fitness* $\left(\frac{1}{(1+makespan)}\right)$ dan urutkan dari yang terkecil hingga yang terbesar. Seelanjutnya tentukan sampel elite. Parameter *p0* diperbaharui berdasarkan data sampel elite untuk mendapatkan sampel yang lebih baik dari iterasi sebelumnya dengan :

$$\widehat{p_t} = \propto \widetilde{\omega_t} + (1-\alpha)\hat{p}_{t-1} \quad (5)$$

dimana,

$\widetilde{\omega_t}$ adalah rata – rata sampel elite

α adalah parameter *smooting* yang bernilai ($0 < \alpha < 1$), pada penelitian ini menggunakan alpha $(\alpha) = 0.9$.

(iv) Pengecekan terhadap Syarat Pemberhentian

Syarat pemberhentian pada penelitian ini adalah *maxit* (*maxit* = 100 iterasi). Jika syarat pemberhentian ini terpenuhi, maka hentikan iterasi dan jika tidak kembali ke langkah (ii).

Algoritma *Cross Entropy with decreasing sample*

Sama halnya dengan algoritma sebelumnya, Algoritma *cross entropy with decreasing sample* untuk permasalahan *flowshop scheduling* juga melalui keempat tahap *cross entropy* pada umumnya. Sesuai dengan namanya pengurangan jumlah sampel, yang berarti terjadi pengurangan secara bertahap

terhadap jumlah solusi yang dibangkitkan pada setiap iterasi ($t \neq 1$). Pengurangan jumlah *npop* setiap iterasi sesuai dengan persamaan :

$$npop(t+1) = decreasing\ sample\ x\ npop(t) \quad (6)$$

Penurunan sampel yang digunakan pada penelitian ini adalah 3%, sehingga *decreasing sample* = (100% - 3%) = 97%. Jumlah sampel untuk setiap iterasi yang dilakukan akan berkurang sebanyak 3% dari iterasi sebelumnya.

Algoritma *Particle Swarm Optimization*

Algoritma *Particle Swarm Optimization* untuk *flowshop Scheduling :*

a. Pembangkitan solusi awal

   Pembangkitan solusi awal X_0 dilakukan dengan membangkitkan bilangan random antara 0 dan 1 berukuran *n x m x npop* (*npop* = 1000, 2000, 3000, 4000). Kemudian dilakukan pengurutan bilangan random untuk memperoleh urutan *job*.

b. Evaluasi *fitness* $\left(\frac{1}{(1+makespan)}\right)$ dari setiap partikel (baris solusi).

c. Tentukan partikel dengan *fitness* terbaik, dan tetapkan sebagai *Gbest*. Untuk setiap partikel, tentukan *Pbest* dengan membandingkan posisi sekarang dengan *Pbest* dari iterasi sebelumnya. Nilai yang diambil untuk *Pbest, Gbest* bukanlah nilai integer pada urutan *job*, melainkan bilangan random yang dibangkitkan pada tahap sebelumnya dan belum diurutkan.

d. Menggunakan *Pbest* dan *Gbest* yang ada, perbarui kecepatan setiap partikel menggunakan persamaan (7) dan tentukan solusi baru untuk iterasi selanjutnya X_{t+1} dengan persamaan (8).

$$V_i(t) = V_i(t-1) + c_1 r_1 \left(X_i^L - X_i(t-1)\right) + c_2 r_2 \left(X^G - X_i(t-1)\right) \quad (7)$$

   dimana :

   X = posisi partikel

   V = kecepatan partikel

   i = indeks partikel t = iterasi ke-t

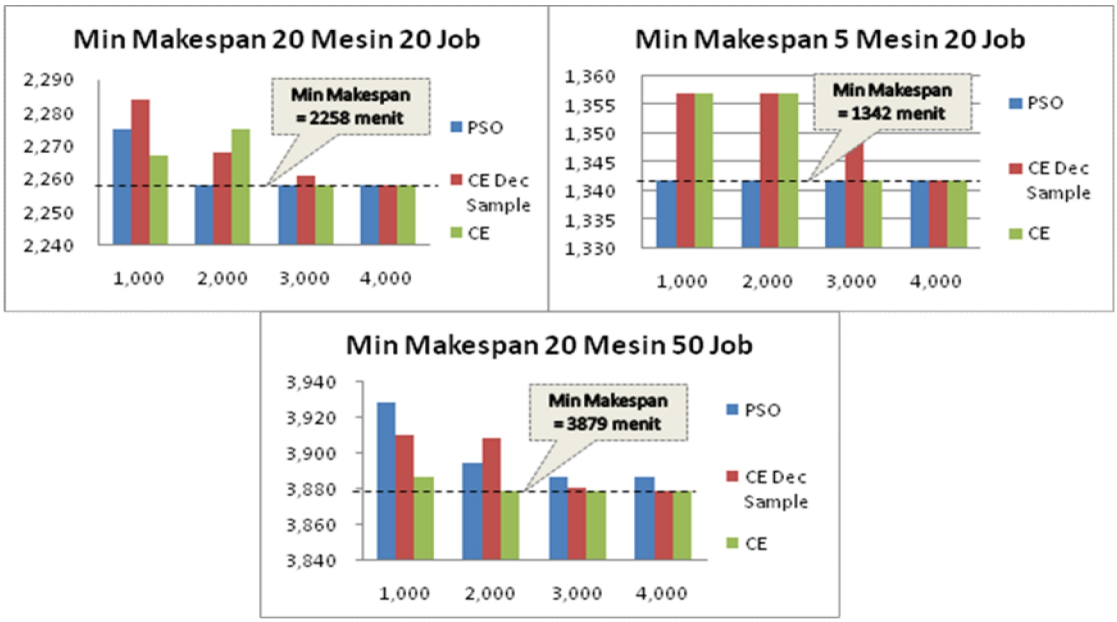
   N = ukuran dimensi ruang

$X_i^L = x_{i1}^L, x_{i2}^L, \ldots, x_{iN}^L$ merepresentasikan *local best* dari partikel ke-*i*. Sedangkan $X^G = x_{i1}^G, x_{i2}^G, \ldots, x_{iN}^G$ merepresentasikan global best dari seluruh kawanan. Sedangkan c_1 dan c_2 adalah suatu konstanta yang bernilai positif yang biasanya disebut sebagai *learning factor*. Kemudian r_1 dan r_2 adalah suatu bilangan random yang bernilai antara 0 sampai 1. Lalu dengan kecepatan baru yang didapat, perbarui posisi setiap partikel menggunakan persamaan :

$$X_i(t) = V_i(t) + X_i(t-1) \tag{8}$$

Analisa dan Hasil Percobaan

Percobaan dilakukan dengan untuk tiga data variasi pasangan *n*-job dan *m*-mesin, yaitu 20 job untuk 5 mesin, 20 job untuk 20 mesin dan 20 job untuk 50 mesin. Adapun data yang digunakan untuk eksekusi program adalah data yang dibangkitkan secara random. Pada setiap penyelesaian permasalahan *flowshop* dieksekusi dengan jumlah sampel yang sama yaitu 1000, 2000, 3000, dan 4000 sampel dengan jumlah iterasi yang sama yaitu 100 iterasi. Hasil Eksekusi Program untuk ketiga jenis data tersebut adalah sebagai berikut :



Gambar 1. Minimum Makespan untuk ketiga skenario

Pada gambar 1 dapat dilihat bahwa algoritma PSO memberikan hasil yang relatif baik untuk pencarian solusi dengan jumlah permasalahan yang relatif kecil (m = 20, n = 20 dan m = 5, n = 20). Hal ini terlihat dari pencapain hasil optimal dengan jumlah sampel minimum 1000 untuk permasalahan berukuran n = 20 dan m = 5. Sedangkan hasil optimal untuk m = 20, n = 20 dicapai ketika jumlah sampel minimum 2000. Namun, pada saat permasalahan menjadi lebih besar (m = 50, n = 20) algoritma PSO memberikan hasil yang kurang baik walaupun sampel yang digunakan cukup besar. Hasil perhitungan minimum makespan algoritma PSO lebih besar daripada yang dihasilkan algoritma CE baik dengan penurunan sampel maupun tidak. Hal ini karena, kelemahan algoritma PSO kelemahan terjebak dalam local optima. Perhitungan *cross entropy* sangat berpengaruh dengan jumlah sampel yang digunakan. Semakin besar jumlah sampel yang dihasilkan akan semakin baik juga solusi yang dihasilkan. Dan semakin besar jumlah sampel yang digunakan semakin panjang pula waktu komputasi yang digunakan (tabel 1).

Tabel 1. *Computation Time* Eksekusi Program (detik)

Jumlah		Sampel				Keterangan
mesin	Job	1,000	2,000	3,000	4,000	
5	20	3.33	5.72	8.11	10.50	PSO
		64.53	132.77	201.00	269.23	CE Decreasing Sample
		243.43	398.42	553.41	708.40	CE
20	20	7.31	15.13	22.95	30.77	PSO
		65.06	134.15	203.25	272.34	CE Decreasing Sample
		242.91	395.56	548.21	700.86	CE
50	20	12.52	27.23	41.95	56.67	PSO
		295.61	586.77	877.92	1,169.08	CE Decreasing Sample
		911.95	1,910.40	2,908.85	3,907.29	CE

Waktu eksekusi program (*computation time*) yang diperlukan untuk menyelesaikan permasalahan dengan algoritma *PSO* paling kecil bila dibandingkan dengan kedua algoritma lainnya (*CE* dan *CE decreasing sample*). Penurunan jumlah sampel pada algoritma *Cross Entropy* membutuhkan waktu perhitungan yang relatif singkat dibandingkan dengan algoritma *CE* tanpa penurunan jumlah sampel.

Kesimpulan dan Saran

Kesimpulan pada penelitian ini adalah : Algoritma *cross entropy* sangat tergantung dengan jumlah sampel yang digunakan. Semakin besar sampel yang digunakan semakin baik solusi yang dihasilkan. Oleh karena itu penurunan sampel pada *cross entropy* dapat mempercepat waktu perolehan solusi untuk permasalahan tersebut. Penurunan sampel pada algoritma *cross entropy* memberikan solusi yang cukup baik bila dibandingkan dengan algoritma PSO untuk permasalahan yang cukup besar. Namun, algoritma *cross entropy* memberikan solusi yang tidak cukup baik jika digunakan untuk permasalahan dengan ukuran kecil. Algoritma *cross entropy decreasing sample* sesuai untuk problem berukuran besar dalam perolehan solusi optimal, hanya saja jumlah sampel yang digunakan untuk *cross entropy decreasing sample* harus cukup besar.

Daftar Pustaka

- Budi Santosa, Muhammad Arif Budiman, Stefanus Eko Wiratno. (2011). *A Cross Entropy-Genetic Algorithm for m-Machines No-Wait Job-ShopScheduling Problem.* JILSA 3(3) : 171-180.
- Caserta, M.; Rico, E. Quiñonez; dan Uribe, A. Márquez. 2008. *A Cross Entropy Algorithm for the Knapsack Problem with Setups*. Computers & Operations Research 35: 241 – 252
- Hejazi, Reza S., S. Shaghafian. (2005). *Flowshop-scheduling problems with makespan criterion: a review*. International Journal of Production Research, Vol. 43, No. 14, 15 : 2895–2929.
- Krisnawati, Maria, Budi Santosa, Ahmad Rusdiansyah. (2011). *Comparison Of Cross Entropy And Differential Evolution To Solve Crew Rostering Problem.* International Engineering and Service Science (IESS) Conference, Surakarta, Indonesia.
- Lathifi, Muhammad Fahmi, Budi Santosa, Stefanus Eko W. 2005. *Pengembangan Metode Hybrid TS-CE untuk Penjadwalan Flowshop*. Tugas Akhir : Institut Teknologi Sepuluh Nopember**.**
- Laha, Dipak. 2008. *Heuristics and Metaheuristics for Solving Scheduling Problems*. India : Jadavpur University.

- Ponnambalam,S.G.;Aravindan,P.;Chandrasekaran,S. 2001. *Constructive and Improvement Flow Shop Scheduling heuristics : an extensive evaluation.* Production Planning & Control Journal,Vol.12,N0.4,335-344

- Rajendran, C., & Ziegler, H. (2004). *Ant-Colony Algorithms for Permutation Flowshop Scheduling to Minimize Makespan/Total Flowtime of Jobs,* Journal of Computers and Industrial Engineering.

- Rubinstein, Reuven Y., dan Kroese, Dirk P. (2004). *The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation, and Machine Learning*. New York: Springer Science+Business Media, Inc.

- Tang, L., Xuan, H. and Liu, J. (2006). *A new Lagrangian relaxation algorithm for hybrid flowshop scheduling to minimize total weighted completion time.* Computers & Operations Research, 33: 3344 – 3359.

- Tagestiren, M. Fatih, Yun-Chia Liang, Mehmet Sevklic, Gunes Gencyilmazd. (2007). *A particle swarm optimization algorithm for makespan and total flowtime minimization in the permutation flowshop sequencing problem*. European Journal of Operational Research Volume 177, Issue 3 : 1930–1947