University of Thi-Qar Journal Of Science (UTsci)
Website: http://jsci.utq.edu.iq Email: utjsci@utq.edu.iq
Volume 6, Number 3, December 2017

Ant Colony Method to Minimize Single Machine Scheduling Problem

Sami .M Araibi

Department of Mathematics- College of Computer Science and Mathematics- Thi-Qar University

E-mail:sami.mezal@utq.edu.iq

Abstract:

The study deal with a single machine scheduling problem where the objective is to find the sequence which it give the optimal or efficient solution for the objective function the sum of discounted weighted completion time and number of tardy jobs. The optimal solution was found for some special cases. Ant colony optimization (ACO) using to found an approximate solution. Results of extensive computational tests show that proposed (ACO) is effective in solving problems up to (1000) jobs at a time less than or equal to (10) minutes.

Keywords: single machine, scheduling, ant colony, discounted total weighted completion time, number of tardy jobs.

1. Introduction:

A set $N = \{1, 2, \dots, n\}$ of n independent jobs has to be scheduled on a single machine to minimize a bi-criterion. This study concerns the one machine scheduling problem with multiple objectives function denoted by $1// \sum_{j=1}^{n}(w_j\,(1 - e^{-rC_j}) + \mathrm{U_j}\,)$. Clearly that the objective function consist from two P-type problems. The first one $1// \sum_{j=1}^{n} w_j\,(1 - e^{-rC_j})$ which it solve by WDSPT rule (**Pinedo, M.L (2008)**), and the second problem $1// \sum_{j=1}^{n} \mathrm{U_j}$ which it solve by moor's algorithm (**Moore J.M (1968)**). The composed problem is NP-hard, because the problem $1// \sum_{j=1}^{n} \mathrm{C_j} + \sum_{j=1}^{n} \mathrm{U_j}$ is NP-hard (**T.S. Abdul-Razaq et al. (2000)**), because $\sum_{j=1}^{n} \mathrm{w_j C_j}$ is generalize to $\sum_{j=1}^{n} \mathrm{C_j}$ and $\sum_{j=1}^{n} w_j\,(1 - e^{-rC_j})$ is extension to $\sum_{j=1}^{n} \mathrm{w_j C_j}$ (**Pinedo, M.L (2008)**). In this problem, preemption is not allowed, no precedence relation among jobs is assumed, only one job j can be processed at a time. Each job j needs p_j time units to be processed on the machine, and ideally should be completed at its due date d_j. The completion time is C_j ($C_j = C_{j-1} + p_j$, $j = 1, \dots, n$ and $C_0 = 0$). If a job j is completed after it's due date ($C_j > d_j$ then $U_j = 1$), this job is said to be late or tardy job, otherwise ($C_j \le d_j$ then $U_j = 0$), and this job is said to be early or on time. Our objective is to find a schedule to minimize the sum discounted total weighted completion time ($\sum_{j=1}^{n} w_j\,(1 - e^{-rC_j})$) and the number of tardy jobs ($\sum_{j=1}^{n} U_j$).

2. Problem Formulation:

We can be stated the problem $1// \sum_{j=1}^{n}(w_j\,(1 - e^{-rC_j}) + \sum_{j=1}^{n} \mathrm{U_j}\,)$ as follows:

A set of n independent jobs $N = \{1, 2, \dots, n\}$ are available for processing at time zero, job j ($j \in N$) is to be processed without interruption on a single machine that can be handled only one job at a time. For a given sequence σ of the jobs, completion time $C_{\sigma(j)}$ and number of tardy jobs $\sum_{j=1}^{n} U_j$ are given respectively by:

$$C_{\sigma(j)} = \sum_{i=1}^{j} p_i \qquad \dots (1)$$

$$U_{\sigma(j)} = \begin{cases} 1 & \text{if} \quad C_{\sigma(j)} > d_{\sigma(1)} \\ 0 & \text{otherwise} \end{cases} \quad , \; j = 1, \dots, n \qquad \dots (2)$$

mathematically as:

$$\left.\begin{aligned} & Z = \min_{\pi \in \delta} \left\{ \sum_{j=1}^{n} w_{\pi(j)} \left(1 - e^{-rC_{\pi(j)}}\right) + \mathrm{U}_{\pi(j)} \right\} \\ & \text{Subject to} \\ & C_{\pi(j)} \ge p_{\pi(j)} \qquad , \; j = 1, \dots, n \\ & C_{\pi(j)} \ge C_{\pi(j-1)} + p_{\pi(j)} \qquad , \; j = 2, \dots, n \\ & 0 < r < 1 \end{aligned}\right\} \dots (P)$$

University of Thi-Qar Journal Of Science (UTsci)
Website: http://jsci.utq.edu.iq **Email: utjsci@utq.edu.iq**
Volume 6, Number 3, December 2017

Where $\pi \in \delta$, δ represent the set of all feasible

solutions, $|\delta| = n!$ and $\pi(j)$ is the jth positon in schedule π.

3. Optimal Solution:

In this section, we shall give optimal solution for our problem (P) when the data of problem satisfy some conditions.

Theorem 1: If $p_j = p$ and $w_j = w$ $\forall j, (j = 1,2,\dots,n)$, then (MA) gives an optimal solution, for $1/p_j = p,\ w_j = w/\ \sum_{j=1}^{n}(w_j\,(1-e^{-rC_j}) + \mathrm{U_j})$ problem.

Proof:

Since $p_j = p$ and $w_j = w$ $\forall j, (j = 1,2,\dots,n)$, then any order gives (WDSPT) rule. Then the problem $1/p_j = p,\ w_j = w/\ \sum_{j=1}^{n}(w_j\,(1-e^{-rC_j}) + \mathrm{U_j})$ is depend on $1/p_j = p,\ w_j = w/\ \sum_{\mathrm{j}=1}^{n}\mathrm{U_j}$. So (MA) gives an optimal solution for $1/p_j = p,\ w_j = w/\ \sum_{j=1}^{n}(w_j\,(1-e^{-rC_j}) + \mathrm{U_j})$ problem.

Theorem 2: If $p_j = p$ and $d_j = d$ $\forall j, (j = 1,2,\dots,n)$, then the sequence which is sequencing the jobs in decreasing order of their weight, gives an optimal solution for $1/p_j = p,\ d_j = d/\ \sum_{j=1}^{n}(w_j\,(1-e^{-rC_j}) + \mathrm{U_j})$ problem.

Proof:

Since $p_j = p$ and $d_j = d$ $\forall j, (j = 1,2,\dots,n)$, then any order gives the optimal for the problem $1/p_j = p,\ d_j = d/\ \sum_{\mathrm{j}=1}^{n}\mathrm{U_j}$. Then the problem $1/p_j = p,\ d_j = d/\ \sum_{j=1}^{n}(w_j\,(1-e^{-rC_j}) + \mathrm{U_j})$ is depend on $1/p_j = p,\ d_j = d/\ \sum_{j=1}^{n} w_j\,(1-e^{-rC_j})$, and it was solved by sequencing the jobs in decreasing order of their weight, (because $p_j = p$).

Theorem 3: If $p_j = p$, $d_j = d$ and $w_j = w$ $\forall j, (j = 1,2,\dots,n)$, then any order gives an optimal solution, for $1/p_j = p, d_j = d, w_j = w/\ \sum_{j=1}^{n}(w_j\,(1-e^{-rC_j}) + \mathrm{U_j})$ problem.

Proof:

Since $p_j = p, d_j = d, w_j = w$ $\forall j, (j = 1,2,\dots,n)$, then any order gives (WDSPT) rule and (MA). Then any order gives an optimal solution, for $1/p_j = p, d_j = d, w_j = w/\ \sum_{j=1}^{n}(w_j\,(1-e^{-rC_j}) + \mathrm{U_j})$ problem.

Theorem 4: If $\sum_{j=1}^{n} w_j\,(1-e^{-rC_j})(\mathrm{WDSPT}) = \sum_{j=1}^{n} w_j\,(1-e^{-rC_j})(MA)$, then (MA) gives an optimal solution, for $1//\ \sum_{j=1}^{n}(w_j\,(1-e^{-rC_j}) + \mathrm{U_j})$ problem.

Proof:

Since $1//\ \sum_{j=1}^{n} w_j\,(1-e^{-rC_j})$ is minimized by(WDSPT).And $\sum_{j=1}^{n} w_j\,(1-e^{-rC_j})(\mathrm{WDSPT}) = \sum_{j=1}^{n} w_j\,(1-e^{-rC_j})(MA)$. Then (MA) rule is optimal for both criteria $(\sum_{j=1}^{n} w_j\,(1-e^{-rC_j})$ and $\sum_{\mathrm{j}=1}^{n}\mathrm{U_j})$. Hence (MA) rule is optimal for $1//\ \sum_{j=1}^{n}(w_j\,(1-e^{-rC_j}) + \mathrm{U_j})$ problem.

Theorem 5: If $\sum_{\mathrm{j}=1}^{n}\mathrm{U_j}\,(MA) = \sum_{\mathrm{j}=1}^{n}\mathrm{U_j}\,(\mathrm{WDSPT})$, then (WDSPT) gives an optimal solution, for $1//\ \sum_{j=1}^{n}(w_j\,(1-e^{-rC_j}) + \mathrm{U_j})$ problem.

Proof:

Since $1//\ \sum_{\mathrm{j}=1}^{n}\mathrm{U_j}$ is minimized by (MA). And $\sum_{\mathrm{j}=1}^{n}\mathrm{U_j}\,(MA) = \sum_{\mathrm{j}=1}^{n}\mathrm{U_j}\,(\mathrm{WDSPT})$. Then (WDSPT) is optimal for both criteria $(\sum_{j=1}^{n} w_j\,(1-e^{-rC_j})$ and $\sum_{\mathrm{j}=1}^{n}\mathrm{U_j})$. Hence (WDSPT) is optimal for $1//\ \sum_{j=1}^{n}(w_j\,(1-e^{-rC_j}) + \mathrm{U_j})$ problem.

4. Heuristic:

The following algorithm (heuristic) has been proposed to obtain the upper bound for the problem (P).

Step (1): Let $\sigma = (\sigma_{(1)}, \sigma_{(2)}, \dots, \sigma_{(n)})$ be a sequence which obtained by applying (MA).

Step (2): Let Q be a subsequence of tardy jobs which is obtained from σ.

Step (3): Let $\pi = (\pi_{(1)}, \pi_{(2)}, \dots, \pi_{(n)})$ be a new sequence, which is obtained from σ after ordered the jobs of Q according to (WDSPT) rule.

Step(4): Compute $|Q|$, $\sum_{j=1}^{n} w_{\pi(j)} \left(1 - e^{-rC_{\pi(j)}}\right)$.

Step (5): Stop.

5. Ant Colony Optimization:

The Ant Colony Optimization (ACO) algorithm, originally introduced by (**Dorigo et al. (1991)**), is a cooperative heuristic searching algorithm inspired by the ethological study on the behavior of ants. It was observed that ants – who lack sophisticated vision – could manage to establish the optimal path between their colony and the food source within a very short period of time. This is done by an indirect communication via the chemical substance, or pheromone, left by the ants on the paths. Though any single ant moves essentially at random, it will make a decision on its direction based by the "strength" of the pheromone trails that lie before it, where a higher amount of pheromone hints a better path. As an ant traverses a path, it reinforces that path with its own pheromone, which in turn creates an even larger amount of pheromone on those short trails, which makes those short trails more likely to be chosen by future ants (**Gang Wang et al. (2005)**).

5.1.Historical development of ant colony optimization:

Ant algorithms are a population-based approach, which has been successfully applied to several NP-hard combinatorial optimization problems. As the name suggests, ant algorithms have been inspired by the behavior of real ant colonies. One of the main ideas of ant algorithms is the indirect communication of a colony of agents, called (artificial) ants, based on pheromone trails (pheromones are also used by real ants for communication). The (artificial) pheromone trails are a kind of distributed numeric information which is modified by the ants to reflect their experience while solving a particular problem. (**Dorigo et al. (1997)**) have applied the first ACO algorithm, called ant system (AS) to the traveling salesman problem (TSP). In spite of hopeful results, the algorithm results were not comparable to the other advanced algorithms that were already applied to solve this problem. Despite the fact, this algorithm built important principles in creating algorithms that are more advanced. At the present time, many algorithms have been suggested based on the improvement of AS algorithm and used for solving various problems (**Keivan G et al. (2006)**).

5.2. Basic definition of (ACO):

The Main idea of the (ACO) is to keep a population or colony of (n) artificial ants that iteratively builds solution by continually applying a probabilistic decision policy (n) times until a solution is found. Ants that found a good solution mark their path through the decision space by putting some amount of pheromone on the edges of the path. Ants of the next iteration are attracted to the pheromone resulting in a higher probability to follow the already traversed good paths. In addition to the pheromone values, the ants will usually be guided by some problem specific heuristic for evaluating possible decisions regarding which direction to take along the way. In ACO algorithm, ants have a memory that stores visited components of their current path. Apart from the construction of solutions and depositing of pheromone, the ACO incorporates two other methods, pheromone evaporation and optional daemon activities. Pheromone evaporation causes the amount of pheromone on each edge to decrease over time. The important property of evaporation is that it prevents premature convergence to a sub-optimal solution. In this manner, the ACO has the capability of "forgetting" bad solutions, which favors the exploration of the search space. Daemon activities can be any action such as a local search technique over the solutions of ants. Off-line pheromone updating occurs if the daemon is given the responsibility of laying pheromone on the edges (**Ventresca M et al. (2004)**).

5.3.Ant Colony Optimization (ACO) Algorithm:

ACO was suggested as a new heuristic method to solve optimization problems by (**Dorigo et al. (1999)**). The reformed form of the (AS) algorithm and functions is shown as follows. Each ant generates a complete solution by choosing the nodes according to a probabilistic state transition rule. The state transition rule is given in (3) is called a pseudorandom-proportional rule:

$$P^k(ij) = \frac{(t_{ij})^{\alpha}(\eta_{ij})^{\beta}}{\sum_{j\in N_i^k}(t_{ij})^{\alpha}(\eta_{ij})^{\beta}} \qquad \dots(3)$$

Where t_{ij} is the amount of pheromone in edge ij, $\eta_{ij} = \frac{1}{\delta_{ij}}$ where δ_{ij} is the cost of edge ij, α and β are parameters that determine the relative importance of η versus t, and N_i^k is the remaining node set of ant k based on moving from node i to build a feasible solution (**Keivan G et al. (2006)**).

The parameters α, β are user defined parameters that determine the degree to which the pheromone is used versus the heuristic distance in deciding where to move. Setting $\beta = 0$ will result in only the pheromone information being used whereas if $\alpha = 0$, only the heuristic information will be used (**Ventresca M et al. (2004)**).

In either case in ACO, only the globally best ant that has built the best solution deposits pheromone in the graph. At the end of an iteration of the algorithm, once all the ants have built a solution, pheromone is added to the arcs used by the ant that found the best tour from the beginning of the trial. This updating rule is called the global updating rule of pheromone:

$$t_{ij} = (1-p)t_{ij} + p\,\Delta t_{ij} \qquad \dots(4)$$

Where $0 < p < 1$ is a pheromone decay parameter and Δt_{ij} equal to

$$\Delta t_{ij} = \begin{cases} \frac{1}{\text{best cost}} & \text{if } i,j \in \text{best sequence} \\ 0 & \text{otherwise} \end{cases} \qquad \dots(5)$$

In ACO, ants perform step-by-step pheromone updates using local updating rule of pheromone. These updates are performed to favor the emergence of other solutions than the best so far. The updates result in step-by-step reduction of the pheromone level of the visiting edges by each ant.

The local updating rule of pheromone is performed by applying the rule:

$$t_{ij} = (1-\zeta)t_{ij} + \zeta\, t_0 \qquad \dots(6)$$

t_0 is a small fixed value and $0 < \zeta < 1$ is the local evaporation of pheromone (**Keivan G et al. (2006)**).

6. Computational Experience:

The ACO algorithm is tested on problem (P) for finding the near optimal solution by coding it in Matlab R2010a and running on a personal computer Lenovo with Ram 8 GB. Test problems are generated as follows: for each job j, an integer processing time p_j and an integer weights w_j is generated from the discrete uniform distribution [1, 10]. Also, for each job j, an integer due date d_j is generated from the discrete uniform distribution in the interval [q(1-TF-RDD/2), q(1-TF+RDD/2)], $q = \sum_{j=1}^{n} p_j$. Depending on the relative range of due date (RDD) and on the average tardiness factor (TF). For both parameters, the values 0.2, 0.4, 0.6, 0.8 and 1.0 are considered. For each selected value of n where n is the number of jobs, five problems were generated.

In contracts of the local search algorithms, the ACO algorithm does not start from initial solution as we see, this will make ACO algorithm occasionally weaker than the other local search algorithms. In this paper, we modified the ACO algorithm by making it start from a good heuristic method, doing this modification for ACO algorithm (the new procedure) is as follows: Updating (global updating) the path (solution), found by the considered heuristic before starting the ACO algorithm procedure (old procedure). Particularly for our problem, we use heuristic (which introduced in section four) as an initial solution. The following table (1) shows that the two solutions for each example found by both of the new and old procedure and also the number of best solution for each procedure.

Where

n = the number of jobs (problem size).

New pro. = the value of ACO that use the new procedure.

Old pro. = the value of ACO that use the old procedure.

No. best = number of best solution of examples with associated procedure.

Table (1): compare between new and old ACO algorithm

n	50		100		200	
Ex	Old pro.	New pro.	Old pro.	New pro.	Old pro.	New pro.
1	327.1251	323.2586	601.5956	602.4574	1291.9610	1291.1064
2	309.0778	306.3866	691.4574	686.7921	1338.3985	1337.7604
3	333.2480	328.8397	635.2236	632.5082	1283.0341	1280.8887
4	344.1539	341.9369	609.0000	609.9973	1289.5307	1288.5307
5	299.2595	294.7718	654.0527	646.8098	1287.8565	1277.6845
No. best	0	5	2	3	0	5

n	300		400		500	
Ex	Old pro.	New pro.	Old pro.	New pro.	Old pro.	New pro.
1	1922.4884	1922.3067	2651.5917	2650.8391	3215.9898	3212.2160
2	1988.4345	1987.9501	2568.3807	2565.9524	3232.0000	3230.9998
3	1904.9386	1904.0000	2622.5048	2622.7293	3214.0000	3214.0000
4	1897.0000	1891.2792	2648.9996	2648.9999	3253.9998	3251.1829
5	1955.9940	1938.3639	2515.0000	2479.1266	3252.0000	3240.2889
No. best	0	5	2	3	1	5

n	1000					
Ex	Old pro.	New pro.				
1	6515.6374	6514.5821				
2	6540.3212	6540.3212				
3	6559.5067	6559.0000				
4	6426.0000	6427.0000				
5	6431.0000	6433.0000				
No. best	3	3				

It is clear that the algorithm that uses new procedure (start from a good initial solution) is better than the algorithm that uses old procedure (without initial solution).

7.Conclusions:

In this paper, we consider a single machine scheduling problem to minimize Multiple Objective Function (MOF), we assume that a job *i* is a valuable to processing at time zero. ACO presented for problem. Several results concerning optimality of five solvable special cases are presented. The computational result shows that uses new procedure (start from a good initial solution) is better than the algorithm that uses old procedure (without initial solution)

8.References:

Dorigo M., Maniezzo V., and Colorni A., “A positive feedback as a search strategy”, Milan, Italy. tech. Rep. 91-016. (1991).

Dorigo M., and Gambardlla, L.M., “Ant Colony System: A cooperative learning approach to the traveling salesman problem”, IEEE Transactions on Evolutionary computation, 1(1), 53-66 (1997).

Dorigo M., and Gambardlla, L.M., "Ant algorithms for discrete optimization", Massachusetts Institute of technology, artificial life 5: 137-172 (1999).

Gang Wang, Wenrui Gong, Ryan Kastner, “Instruction Scheduling Using (MAX-MIN) Ant System Optimization”, Department of Electrical and Computer Engineering, University of California at Santa Barbara, Chicago, Illinois, USA. 17-19, April 2005.

Keivan G. and Fahimeh M., “ACS-TS: train scheduling using Ant Colony system”, Journal of applied mathematics and design sciences. Article ID (95060) 1-28, (2006).

Moore J.M.,"Sequencing *n* Jobs on one Machine to Minimize the Number of Tardy Jobs", Management Science, 17, NO.1 (September, 1968).

Pinedo, M.L., "Scheduling theory, algorithms, and systems", Spring+er Science +Business Media, LLC., New York (2008).

T.S. Abdul-Razaq and M.K. Zghair,"One Machine Scheduling to minimize Total Cost of Completion Time and Number of Tardy Jobs", Journal of Basrah Research 25, pp. 83-93 (2000).

Ventresca M., and Ombuki B.M., “Ant Colony Optimization for Job Shop Scheduling Problem”, Brock University Canada, L25, 3A1, (2004).