RESEARCH ARTICLE OPEN ACCESS

Ant Colony Optimization for Job Shop Scheduling Problem Using Priority Rules

B.SASIKALA
Assistant Professor, Dept of Computer Science,
Bharathidasan University Constituent College(W),
Orathanadu, Thanjavur,Tamilnadu, INDIA,

Dr.V.P. ESWARAMURTHY
Assistant Professor, Dept of Computer Science,
Government Arts and Science College,
Komarapalayam, Namakkal, Tamilnadu, INDIA.

Abstract:

Scheduling problems have a vital role in recent years due to the growing consumer demand for variety, reduced product life cycles, changing markets with global competition and rapid development of new technologies. The Job Shop Scheduling Problem is one of the most popular scheduling models existing in practice, which is among the hardest combinatorial optimization problems. The Ant Colony Optimization is a technique of swarm intelligence, which is applied to combinatorial optimization problems as JSSP. This paper presents ACO meta-heuristic approach with new strategies in solving JSSP. Priority rules play a major role during the construction of a solution. Different priority rules are analyzed and the best one is found. Experiments using well-known benchmark problems show that this approach improves the performance obtained by the basic ant colony system..

Keywords — **Job Shop Scheduling, Ant Colony Optimization, meta-heuristic.**

I. INTRODUCTION

In spite of globalization and rapidly decreasing product life cycle, manufacturing companies are trying different means to improve productivity through the management of machine utilization and product cycle-time. Effective scheduling is an essential activity in manufacturing industry which leads to improvement in the efficiency and utilization of resources. The job shop scheduling problem is one of the most difficult combinatorial problems in classical scheduling theory and considered as closed and static.[1]. It has attracted many researchers due to its wide applicability and inherent difficulty[2]. In JSSP, a finite number of jobs are available and these jobs are processed by a finite number of machines. For each job, there is a sequence of operations, which needs to be processed without interruption for a given period of time on a given machine. An operation of a job cannot be started until its previous operations of the same job are completed. The completion time of all jobs is known as makespan.The objective is to find a feasible schedule with minimum makespan. Feasible schedules are obtained by permuting the processing order of operations on machines without violating the technological constraints.

A recent adaptive algorithm, named Ant System, is introduced and used to solve the problem of job shop scheduling. The algorithm was first introduced by Dorigo, Maniezzo and Colorni in 1991[3] and is derived from the foraging and recruiting behavior observed in an ant colony[4][5][6][7]. This describes how ants explore the world in search of food sources, then find their way back to the nest and indicate the food source to the other ants of the colony. To do so, ants initially take a random walk in search of food. On the way, each ant deposits a fraction of pheromone back to the nest so as to indicate the source to the others. This pheromone acting as memory preservation for ants in order to come up with the shortest path so this pheromone is useful for increasing the probability of other ants following the same path which is proportioned to the density of the pheromone. Through this mechanism, ants will find the shortest path.

This paper is structured as follows.In Section 2, JSSP is explained and is properly described. In Section 3, Ant colony optimization is described. In Section 4, the literature review is given. The Priority rules and results are given in Section 5 and Finally, in Section 6 the conclusion is given.

II. JOB SHOP SCHEDULING

Job Shop Scheduling problem is one of the widely studied and most complex combinatorial optimization problems. The JSSP can be described as a set of n jobs denoted by J_i where i =1,2…n which have to be processed on a set of m machines denoted by M_k where k=1,2….m. Operation of an i^{th} job on the k^{th} machine will be denoted by O_{ik} with the processing time p_{jk}. For each job, the machine order of operations is prescribed and is known as technological constraints which are static to a problem instance.The completion time of all jobs is known as makespan.

The time when an operation begins is not specified, so work can begin at any point in time as long as the required machine is available. Each job must go through a particular sequence of operations that is predefined, so that operations cannot begin until the end of its predecessor, preventing the processing of two operations of the same job concurrently. A machine performs only one job at a time. Once an operation is initiated for processing, it will not be interrupted until its completion. More than one operations of a job cannot be processed on a single machine. Jobs must wait for the next machine to be available.In addition, to the above constraints, we have determined that all operations have the equal priority of processing, and all machines are the same and can be idle at any time.

Table 1. A 3×3 instance of JSSP

Jobs (J)	**Machine (Time)**		
J1(O1,O2,O3)	3(4)	2(3)	1(3)
J2(O4,O5,O6)	2(1)	3(2)	1(4)
J3(O7,O8,O9)	2(3)	1(2)	3(3)

2.1 Disjunctive graph

The JSSP is usually represented as a disjunctive graph $G = (V, C \cup D)$, where *V* is a set of nodes representing operations of the jobs together with two special nodes, a node I and node F, representing the start and end of the schedule, respectively. *C* is a set of conjunctive arcs representing technological sequences of the operations(linking operations corresponding to the same job) and *D* is a set of disjunctive arcs representing pairs of operations that must be performed on the same machines.In addition the processing time of each operation(weighted value) is placed in the upper part of the node.

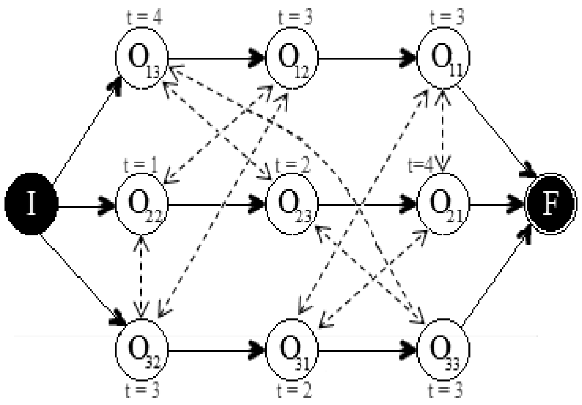

Figure 1. Disjunctive Graph Representation for 3X3 Problems in Table 1.

Ant Colony Optimization (ACO) is an evolutionary metaheuristic to solve combinatorial optimization problems by using principles of communicative behavior found in original ant colonies. The intellectual group manners characterize the whole colony of social insects, such as ants; examples of that emergent behavior include foraging and nest building. This collective behavior can be viewed as a powerful problem-solving system that can solve problems such as scheduling and load balancing. Properties associated with their group behavior, such as self-organization, flexibility and robustness, can be shown as characteristics that should exist in the complex system for control, optimization and problem-solving techniques [8][9][10].

The ACO heuristic has been developed based on a behavioral pattern of ants foraging for food. The ants search for food more or less follows a certain pattern. Initially, ants roam randomly looking for a food source. Once the food source is found, ants return to their colony. On their way back, these ants lay a chemical called ‘pheromone’. Consequently, a pheromone trail is formed, which can be sensed by other ants. Once other ants wandering for food sense a pheromone trail, they no longer move randomly rather follow the pheromone trail. While returning from the food source, these ants reinforce the pheromone trail. However, pheromone trail also evaporates with time resulting in a decrease of attractive strength, and on a longer path pheromones strength decrease due to evaporation. If a shorter path is detected, then more ants follow this shorter path.

This result increases the pheromone level on this shorter path. This positive feedback eventually

leads to following a single short path. The advantage of pheromone evaporation is that longer paths are avoided due to decreased pheromone level.

IV. LITERATURE REVIEW

The JSP has been proven to be NP-hard. Therefore, only small size instances of the JSP can be solved optimally with good computational time using exact solution methods[11][12]. When the problem size increases, the computational time of exact algorithms grows exponentially. Heuristic algorithms have generally acceptable time and memory requirements to obtain a near-optimal or optimal solution. During the past few decades, most researchers on the JSP have been concentrated on developing heuristic algorithms.

Kacem et al.[13] developed an assignment and scheduling procedure, called approach by localization, which was used to assign each operation to the suitable machine considering the processing times and workloads of the machines. In order to find better results for many real problems, they hybridized the GA with an approach by localization to combine the advantages of the methods. Pezzella et al.[14] integrated di_erent strategies to improve the performance of GAs for the FJSP. They used approach by localization to generate the initial solution and dispatching rules to obtain the sequencing of the initial assignments.

Variants of the ACO algorithm generally differ in the applied pheromone update rule. Dorgio and Blumb[15] pointed out the three main types of ACO algorithm: ant system (AS), max-min ant system (MMAS), ant colony system (ACS). ACS and MMAS are regarded as the most successful ACO variants in practice. AS was introduced by Dorigo et al. [16]. In this algorithm, each ant reinforces the value of the pheromone on their path. There are three methods for calculating the pheromone update: ant cycle, ant density, and ant quantity. MMAS was introduced in [17] and differs from AS in several important aspects. Only the best solution from a population is used to update the pheromone values and a mechanism is added to limit the strengths of pheromones in order to avoid premature convergence. In the case of ACS, the state transition rule provides a direct way to balance the exploration of new edges and the exploitation of accumulated knowledge. ACS uses a global updating rule and local pheromone updating rule.

E. Taillard [1989] has proposed a paper about Benchmarks' for Basic Scheduling Problems about 260 scheduling problems whose rare examples were published. Those kinds of problems correspond to real dimensions of industrial problems. In this paper, he solved the flow shop, the job shop and the open shop scheduling problems and provides all benchmark results [18].

V. IMPLEMENTATION

During the construction of a new solution, the state transition rule is the phase where each ant decides which is the next state to move to.Starting in an initial node, every ant chooses the next node in its path according to the state transition rule by using a probability of transition. Let S be the set of nodes at decision point i. The transition probability for choosing the edge from node i to node j by the ant k at the time t is calculated as in equation (1). $\tau(i.\ j)$ is the quantity of the pheromone on the edge between node i and node j. $\eta(j)$ is the inverse of the operation time of the node j. α and β tune the relative importance in the probability of the amount of the pheromone versus the heuristic distance.

$$P_k(i,j) = \begin{cases} \dfrac{\tau(i,\ j)]^{\alpha}.\ [\eta(\ j)]^{\beta}}{\sum_{j\in S}[\tau(i,j)]^{\alpha}.[\eta(j)]^{\beta}} & \text{if } j\in S \\ 0 & \text{Otherwise} \end{cases} \tag{1}$$

Ants use state transition rule to select the next state that is to be added to a partial solution. ACS employs a transition rule called *pseudo-random-proportional*, which is a balance between *pseudo-random* state choice rule used in Q-learning (Watkins and Dayan 1992) and *random-proportional* action choice rule used in AS. In ACS, an ant selects a state using the biased random choice as in AS during some of the time, whereas the best state is selected during the rest of the time based on the heuristic information and the pheromone level. *Pseudo-random-proportional* rule selects the best state with a probability q_0 and selects a random state with a probability 1-q_0 where q_0 is a constant given as input ranging from 0 to 1. However, all the time, the r*andom-proportional* rule used in AS selects the next state randomly with a probability distribution, which depends on the heuristic information and the pheromone level. *Pseudo-random-proportional* state transition rule in ACS provides a way to compromise between exploration of new states and exploitation of the heuristic information and the pheromone level. Hence, the *pseudo-random-proportional* rule uses a state transition rule given in Equation (2).

$$s = \begin{cases} \max\limits_{j \in S}\{[\tau(i,j)]^{\alpha}.[\eta(j)]^{\beta}\} & \text{if } q \le q_0 \\ r & \text{otherwise} \end{cases} \tag{2}$$

where $q \in [0,1]$ is a uniform random number and r is a component, which is chosen randomly according to the probability distribution defined by Equation(1). The random number q is selected each time an ant moves from a state i to another state j. If the value of q is less than or equal to the value of q_0, the ant will select the best state. Otherwise, the ant will select a biased random state.

5.1 Priority Rules and their Comparisons

Priority rules used in state transition have more influence in solving JSSP instances. Three variations of priority rules are discussed and are evaluated in this section by using well-known benchmark instances of JSSP. The different rules discussed in this section are

(1) 1/Operation time
(2) Job time/Number of Operations
(3) 1/(Job remaining time/Operation time).

One of the rules can be applied in state transition rule given in Equation(2). In the first rule, the operation with minimum time will have higher the probability to be scheduled in the partial solution. In the second rule, the time of a job, in which the operation belongs, is divided by the number of operations and hence, operation with minimum time in a long job will have the higher probability for the schedule. In the third rule, remaining job time is divided by the operation time then compute 1/(job remaining time/operation time). The remaining job time denotes the total time of operations yet to be scheduled in that job and hence, the probability of the operation with minimum time belonging to longer remaining time of a job will be high. Different priority rules are tested using well-known JSSP instances with α = 0.7, β = 0.9, ρ = 0.001 and Q = 100. Table 2 shows a number of iterations required to reach the upper bound values for several instances LA01-LA15 (Lawrence 1984), FT06 and FT10 (Fisher and Thompson 1963) and ABZ5 and ABZ6 (Adams et al 1988) by using different priority rules and average value of five runs are recorded for each problem under each rule.

The variation parameter is not applied now to test the algorithm because the purpose of this section is to know the result of three priority rules. The number of ants used for testing is 2 × n × m. It is clear that second rule results in more iterations required to reach the optimal solution. A lesser number of iterations are required to get the optimal value by using the third rule. In the first rule, the required number of iterations to produce the best result is moderate level compared to the second rule. Hence, the third rule is superior to all rules used in the method.

Table 2. Performance of Different Priority Rules with α = 0.7, β = 0.9, ρ = 0.001, q_0 = 0.0 and Q = 100

Problem Instance	Problem Size	Opt	**Upper Bound**		
			Rule1	**Rule 2**	**Rule 3**
LA01	10 × 5	666	677	672	666*
LA02	10 × 5	655	683	675	661
LA03	10 × 5	597	626	619	603
LA04	10 × 5	590	637	628	614
LA05	10 × 5	593	593*	593*	593*
LA06	15 × 5	926	931	927	926*
LA07	15 × 5	890	922	910	905
LA08	15 × 5	863	887	882	871
LA09	15 × 5	951	957	955	951*
LA10	15 × 5	958	958*	958*	958*
LA11	20 × 5	1222	1245	1235	1233
LA12	20 × 5	1039	1052	1044	1039*
LA13	20 × 5	1150	1186	1180	1165

LA14	20 × 5	1292	1292*	1292*	1292*
LA15	20 × 5	1207	1250	1240	1234
FT06	6 × 6	55	55*	55*	55*
FT10	10 × 10	930	955	942	933
ABZ5	10 × 10	1234	1264	1257	1241
ABZ6	10 × 10	943	987	970	958

VI. CONCLUSION

The Ant Colony Optimization is a technique of swarm intelligence, which is applied to combinatorial optimization problems as JSSP. This paper presented the application of ant colony optimization system to solve job shop scheduling problems. The goal of the work was to gain some insight into the influence of the priority rules which seems to play an important role in the construction of good solutions. Different priority rules are analyzed and the best one is found.

REFERENCES

[1] M. R. Garey "*The complexity of Flowshop and Job Shop Scheduling*," Mathematics of Operations Research, vol. 1, no. 2, pp. 117–129, 1976.

[2] A. Jain and S. Meeran, "Deterministic job-shop scheduling: Past, present and future," *European Journal of Operational Research*, vol. 113, pp. 390–434, 1999.

[3] M.Dorigo, V.Maniezzo, and A. Colorni, The ant system: an autocatalytic optimizing process, Technical Report TR91-016, Politecnico di Milano, 1991.

[4] M. Dorigo, V. Maniezzo, A. Colorni, "The Ant System: Optimization by a colony of cooperating agents"" *IEEE Trans. Systems, Man,Cybernetics,*. Vol.26, no.2, pp.29-41, 1996.

[5] M. Dorigo, L. M. Gambardella, "Ant colony System: A Cooperative Learning Approach to the Travelling Salesman Problem", *IEEE Trans. On Evolutionary Computation,* vol.1,no.1, pp.53-66, April 1997.

[6] M. Dorigo, V. Maniezzo, A. Colorni, "Distributed Optimization by Ant Colonies", *Proceedings of ECAL91 – European Conference on Artificial Life*, Elsevier Publishing, pp:134-142, 1991.

[7] M. Dorigo, V. Maniezzo, A. Colorni, "An Investigation of some properties of an Ant Algorithm", *Proceedings of the Parallel Problem Solving From Nature Conference (PPSN92)*, Brussels, Belgium, Elsevier Publishing, 509-520, 1992.

[8] P.Tarasewich, and P.R. McMullen, 2002. Swarm intelligence: Powers in numbers. Commun. ACM, 45: 62-67.

[9] J. Krohn,2001. Ant algorithms and the swarm intelligence model of problem solving. Proceedings of the Conference on the UMM Computer Science Discipline Seminar, (CCSDS'01), UMM University of Minesota Morris, Morris, pp: 1-6.

[10] D. Merkle., M. Middendorf., H. Schmeck, 2000. Pheromone evaluation in ant colony optimization. Proceedings of the 26th Annual IEEE International Conference on Industrial Electronics, Control and Instrumentation, Oct. 22-28, IEEE Xplore Press, Nagoya, Japan, pp: 2726-2731.

[11] J.Carlier.,& E. Pinson, E. (1989). An algorithm for solving the job-shop problem, *Management Science*,35, 164–176.

[12] Lenstra, J.K. (1976). *Sequencing by enumerative methods*. Tech. Rep. Mathematical Centre Tract 69,Mathematisch Centrum, Amsterdam.

[13] I. Kacem, S. Hammadi and P. Borne, Approach by localization and multiobjective evolutionary optimization for flexible job- shop scheduling problems, Ieee Transactions on Systems Man and Cybernetics Part C-Applications and Reviews, 32 (2002), 1-13.

[14] F. Pezzella, G. Morganti and G. Ciaschetti, A genetic algorithm for the flexible job-shop scheduling problem, Computers and Operations Research, 35 (2008), 3202-3212.

[15] Dorigo M, Blumb C (2005) Ant colony optimization theory: a survey. Theory Comput Sci 344:243–278.
[16] Dorigo M, Maniezzo V, Colorni A (1996) Ant system: optimization by a colony of cooperating agents. IEEE Trans Systems Man Cybernet-Part B 26:29–41.
[17] Stützle T, Hoos HH (2000) MAX-MIN ant system. Future Gen Comput Systems 16(8):889–914.
[18] E. Taillard, "BenchMarks For Basic Scheduling Problems," European Journal of Operations Research, 64, 1993, pp. 278-285.