Contents lists available at ScienceDirect

Robotics and Computer-Integrated Manufacturing

journal homepage: www.elsevier.com/locate/rcim

Application of ant colony, genetic algorithm and data mining-based techniques for scheduling

Surendra Kumar [a,*], C.S.P. Rao [b]

[a] ARDE, Pune, Maharastra, India
[b] Mechanical Engineering Department, NIT, Warangal, AP, India

A R T I C L E I N F O

Article history:
Received 23 December 2008
Received in revised form
25 February 2009
Accepted 9 April 2009

Keywords:
Batch processing flow shop
Ant colony optimization
Genetic algorithm operators
Chimerge algorithm
Data mining
See5

A B S T R A C T

In this paper, we have proposed a novel use of data mining algorithms for the extraction of knowledge from a large set of flow shop schedules. The purposes of this work is to apply data mining methodologies to explore the patterns in data generated by an ant colony algorithm performing a scheduling operation and to develop a rule set scheduler which approximates the ant colony algorithm's scheduler. Ant colony optimization (ACO) is a paradigm for designing metaheuristic algorithms for combinatorial optimization problems. The natural metaphor on which ant algorithms are based is that of ant colonies. Fascinated by the ability of the almost blind ants to establish the shortest route from their nests to the food source and back, researchers found out that these ants secrete a substance called 'pheromone' and use its trails as a medium for communicating information among each other. The ant algorithm is simple to implement and results of the case studies show its ability to provide speedy and accurate solutions. Further, we employed the genetic algorithm operators such as crossover and mutation to generate the new regions of solution. The data mining tool we have used is Decision Tree, which is produced by the See5 software after the instances are classified. The data mining is for mining the knowledge of job scheduling about the objective of minimization of makespan in a flow shop environment. Data mining systems typically uses conditional relationships represented by IF-THEN rules and allowing the production managers to easily take the decisions regarding the flow shop scheduling based on various objective functions and the constraints.

1. Introduction

In this paper, we propose an approximate method to resolve a multi-product batch flow shop schedule. Our paper is based on the Koonce's work [1] that uses data mining techniques to extract forms from genetic algorithm solution, but we propose an ant colony algorithm that is fast and easy to implement for generating a learning population. Further we propose another method of knowledge extraction that will generate decision rules finding the affection order of operation on all machines. We studied an 8×4 benchmark problem of Taillard [2] to compare our results.

This paper is divided into two sections. The first section describes the ant colony algorithm (AC) and the method to generate a population of the optimal sequences. The second section deals with mining the solutions given by AC to extract from them the decision rules. These rules are based on several attributes like processing time (PT), position in the job, remaining time of the job, machine loading (ML), etc. to find the affection order of operation on every machine.

2. Ant colony optimization (ACO)

Ants are social insects. They live in colonies and all their actions are towards the survival of the colony as a whole, rather than the benefit of a single individual of the society. The individual ants have no special abilities. They communicate between each other using chemical substances, the pheromones. This indirect communication allows the entire colony to perform complex tasks, such as establishing the shortest route paths from their nests to feeding sources. An optimization algorithm was proposed that tries to mimic the foraging behavior of real ants, i.e. the behavior of wandering in the search for food [3]. This algorithm has already been successfully used to solve the TSP and other NP hard optimization problems (Fig. 1).

When an ant is searching for the nearest food source and comes across with several possible trails, it tends to choose the trail with the largest concentration of pheromone τ, with a certain probability ρ. After choosing the trail, it deposits a certain quantity of pheromone, increasing the concentration of pheromones in this trail. The ants return to the nest using always the same path, depositing other portion of pheromone in the way back. Imagine then, that two ants at the same location choose two different trails at the same time. The pheromone concentration on

* Corresponding author.
E-mail address: surendra.kumar137@gmail.com (S. Kumar).

0736-5845/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.rcim.2009.04.015

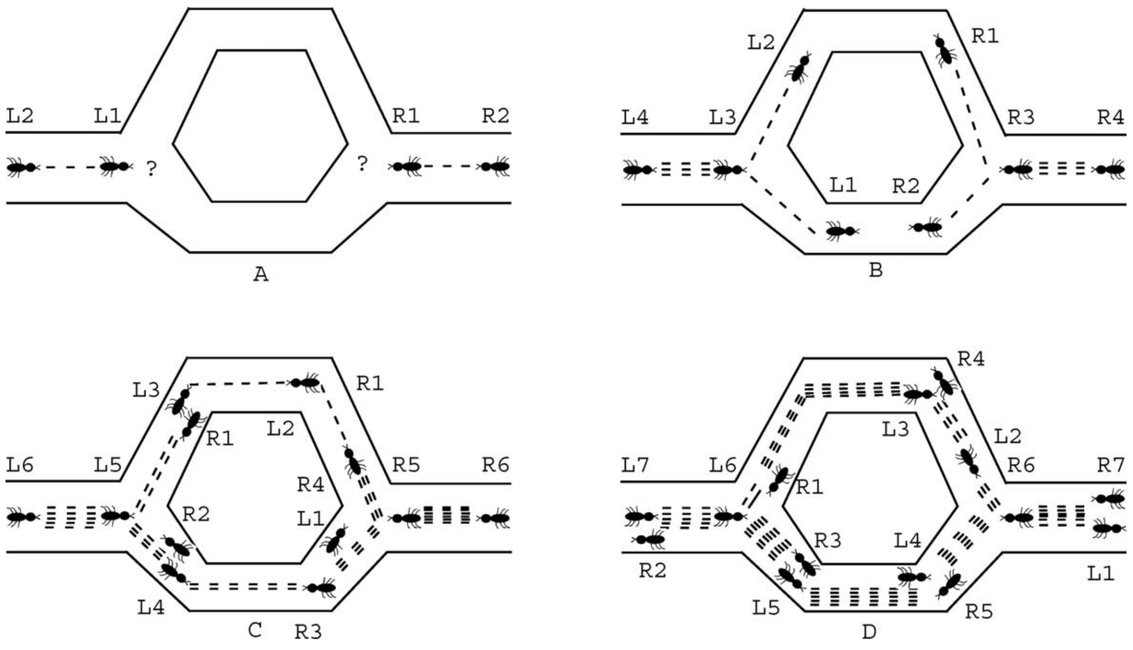

Fig. 1. How real ants find a shortest path. (A) Ants arrive at a decision point. (B) Some ants choose the upper path and some the lower path. The choice is random. (C) Since ants move at approximately constant speed, the ants which choose the lower, shorter, path reach the opposite decision point faster than those which choose the upper, longer, path. (D) Pheromone accumulates at a higher rate on the shorter path. The number of dashed lines is approximately proportional to the amount of pheromone deposited by ants.

the shortest way will increase faster than the other: the ant that chooses this way, will deposit more pheromones in a smaller period, because it returns earlier. If a whole colony of thousands of ants follows this behavior, soon the concentration of pheromone in the shortest path will be much higher than the concentration in other paths. Then the probability of choosing any other way will be very small, and only very few ants among the colony will fail to follow the shortest path. There is another phenomenon related with the pheromone concentration. Since it is a chemical substance, it tends to evaporate in the air, so the concentration of pheromones vanishes along the time. In this way, the concentration of the less used paths will be much lower than that on the most used ones, not only because the concentration increases in the other paths, but also because their own concentration decreases.

3. Batch processing flow line

The problem of scheduling flow shops with three or more stages has been considered to be NP-complete in strong sense [4]. Niem [5] provide overviews of the role of scheduling by defining what the scheduling problem involves and the various network structures, i.e., the layout of machines and flow of material that are prevalent to the batch production industry. The batch processing flow line is a kind of popularly implemented production lines in the metalworking and semiconductor industries. In some stages of a batch processing flow line, a limited number of jobs with the same production demand can be processed simultaneously as a batch by a set of the batch processing machines, examples include diffusion and thermal annealing operations in semiconductor wafer fabrication, burn-in operations in semiconductor testing, heat treatment in metalworking and the printed circuit board (PCB) SMT flow line [6]. Further batch process manufacturing finds its usefulness in the pharmaceutical, polymer, food and especially chemical industries, because it provides the necessary flexibility to accommodate various production requirements using the same processing facility.

The makespan minimization of the serial multi-product batch problem comprises of two tasks: (1) determination of completion times dealing with the details of operation schedule for a given sequence, and (2) determination of the optimal sequence. This batch scheduling problem comprises of several different subcategories depending on the way of handling the intermediate products between successive stages. In this paper, we specifically consider that each of N ($i = 1,\ldots, N$) jobs follow the same job order on M ($j = 1,\ldots, M$) machines (flow shop scheduling) and further make the following assumptions:

a. Processing time T_{ij} for each job on each machine is known a *priori*.
b. Use of suitable intermediate policy.
c. Each job can be processed on one machine at a time.
d. Jobs are not preemptable.

For batch processes, different intermediate storage policies are considered in literature, namely, the unlimited intermediate storage, finite intermediate storage, no intermediate storage and zero-wait processing (ZW). Heuristic and MILP formulations for makespan minimization for all different storage policies are available in literature [7–11]. Here, we have considered the ZW policy, which is widely followed in industries.

3.1. ZW policy

In batch processing, some products cannot be stored at all between some stages in the processing sequence like in chemical or heat treatment processes. These batches of products are usually unstable reaction intermediates that must be processed by the next processing unit immediately upon completion of the current processing unit. For such product stages, zero-wait policy must be adopted [12]. The completion time of product i at unit j, C_{ij} for ZW policy can be given by

$$C_{ij} = C_{iM} - \sum_{k=j+1}^{M} t_{ik} \tag{1}$$

$$\begin{aligned} C_{iM} &= \max[(C_{(i-1)M} + t_{iM}), \ldots, (C_{(i-2)2} + t_{iM} + t_{i(M-1)} + \cdots + t_{i2}), \\ &\quad (C_{(i-1)1} + t_{iM} + t_{i(M-1)} + \cdots + t_{i1})] \\ &= \max\left(C_{(i-1)1} + \sum_{k=1}^{M} t_{ik}, C_{(i-1)2} + \sum_{k=2}^{M} t_{ik}, \ldots, C_{(i-1)M} + \sum_{k=M}^{M} t_{ik}\right) \end{aligned}$$

4. ACO algorithm for flow shop scheduling problem

In general, the ant colony behavior can be described formally using the following mathematical framework. Let the nest and the food source be connected by several different paths, connecting n intermediate nodes. The ant k in node i chooses one of the possible trails (i, j) connecting the actual node to one of other possible positions $j \in \{1, \ldots, n\}$ with probability

$$p_{ij}^{k} = f(\tau_{ij}) \tag{2}$$

where τ_{ij} is the pheromone concentration on the path connecting i to j, in the way to the food source. The pheromone in this trail will vary in time according to

$$\tau_{ij}(t+1) = \tau_{ij}(t) \times \rho + \delta_{ij}^{k} \tag{3}$$

where ${\delta_{ij}}^{k}$ is the pheromone released by the ant k on the trail (i, j) and $\rho \in [0, 1]$ is the evaporation coefficient. The system is continuous, so the time acts as the performance index, since the shortest paths will have the pheromone concentration increased in a shorter period. This is the mathematical description of a real colony of ants. However, the artificial ants that mimic this behavior, can be uploaded with more characteristics, e.g. memory and ability to see. If the pheromone expresses the *experience* of the colony in the job of finding the shortest path, memory and ability to see, express useful *knowledge* about the problem the ants are solving. In this way, (2) can be extended to

$$p_{ij}^{k}(t) = \begin{cases} \dfrac{\tau_{ij}^{\alpha}\eta_{ij}^{\beta}}{\sum_{r\notin\Gamma}^{n}\tau_{ir}^{\alpha}\eta_{ir}^{\beta}} & \text{if } j \notin \Gamma \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

where η_{ij} is a *visibility function* and Γ is a *tabu list*. In this case, the visibility expresses the capability of seeing which is the nearest node j to travel towards the food source. Γ is a list that contains all the trails that the ant has already passed and must not be chosen again (artificial ants can go back before achieving the food source). This acts as the memory of an ant. The parameters α and β express the relative weight between the importance's of pheromone concentration and the visibility η. Finally, each ant deposits a pheromone ${\delta_{ij}}^{k}$ on the chosen trail

$$\delta_{ij}^{k} = \tau_c \tag{5}$$

where τ_c is a constant.

In the artificial ant's framework, Eq. (3) is not sufficient to mimic the increasing pheromone concentration in the shortest path. With real ants, time acts as a performance index, but the artificial ants use all the same time to perform the task, whether they choose a short path or not. For the artificial ants, the length l of the paths they have passed will determine if the solution is good or not. Thus, the best solution should increase even more the pheromone concentration on the shortest trail. To do so, (3) is changed to

$$\tau_{ij}(t+1) = \tau_{ij}(t) \times \rho + \Delta\tau_{ij} \tag{6}$$

where $\Delta\tau_{ij}$ are pheromones deposited in the trails (i, j) followed by all the q ants,

$${}^{q}\Delta\tau_{ij} = \sum_{K=1} \delta_{ij}^{k} \times f(1/z_k) \tag{7}$$

z_k is the performance index. The paths followed by the ants that achieved the shortest paths have their pheromone concentration increased. Notice that the time interval taken by the q ants to do a complete tour is *t+n* iterations. A *tour* is a complete route between the nest and the food source and an *iteration* is a step from i to j done by all the ants. The algorithm runs $N_{\max}$ times, where in every Nth tour, a new *ant colony* is released. The total number of iterations is $N_{\max\,x}\, n$. The general algorithm for the ant colonies is described in Fig. 2

4.1. Ant colony algorithm with genetic algorithm operators for flow shop

Let us now illustrate the ant algorithm for optimal scheduling of batch production with $N = 6$ products each being processed on $M = 4$ machines using processing times t_{ij} given in Table 1.

For this problem, $A = 5$ software ants (agents) are deployed to find an optimal schedule that minimizes the makespan given by Eq. (2). The ants decide the position of product number in the schedule by either one of the following processes:

(a) Chooses the site with maximum pheromone trail using certain probability q_0 (q_0 is a priori defined number in the range (0, 1), we used $q_0 = 0.75$ for illustrative example).
(b) Chooses one of the N sites using a stochastic distribution with a probability $(1-q_0)$, denoted as, p_{ij}.

```
Initialization:
 Set for every pair (i, j): $\tau_{ij} = \tau 0$
  Set $N = 1$ and define a $Nmax$
                 Place the $q$ants
   While $N <= Nmax$
    Build a complete tour
    For i = 1 to n
      For k = 1 to q
      Choose the next node using $p_{ij}^{k}$ = (3)
                 Update locally $\tau_{ij}$ (t) using (4)
                   Update the tabu list $\Gamma$
      Analyze solutions
        For k = 1 to q
           Compute performance index z
           Update globally $\tau_{ij}$ (t + n) using (5)
```

Fig. 2. Ant colonies optimization algorithm.

Table 1
Process data for example.

Products	Units			
	U1	U2	U3	U4
P1	10	20	5	30
P2	15	8	12	10
P3	20	7	9	5
P4	14	6	15	10
P5	6	11	5	15
P6	13	7	17	10

Table 2
Sequence generated.

Product	P1	P2	P3	P4	P5	P6
Position	4	1	6	3	2	5

The first process is known as exploitation whereas the latter is termed as biased exploration. To decide one of the above mechanisms a random number between zero and one must be drawn. Let the chosen random number be 0.69. As this value is less than $q_0 = 0.75$, the first process will be chosen to decide position of product number. From the pheromone matrix, which product number has the highest affinity towards position number one (i.e. a position chosen with highest pheromone concentration) is decided. Let the ant choose element one next for deciding its position in the schedule. The algorithm generates a random number say, $r = 0.86$. The random number generated is greater than q_0 hence; the algorithm uses the mechanism (b) for positioning element one in the current partial schedule. The ant chooses position j for placing element i by exploitation using Eq. (4). In this way, the ant develops its complete schedule. As soon as an ant constructs the solution, the pheromone trail τ_{ij} is updated locally using the following Eq. (5). Where, ρ is the evaporation rate generally lies in the range (0, 0.1) and $(1-\rho)$ is the persistence of trail. Thus, higher value of evaporation rate results into greater loss of information gathered in the past. Thus, the complete schedule generated by the first ant is given in Table 2.

The remaining four ants construct their solutions and modify the pheromone trail matrix using local updating rule employing the above-mentioned procedure. The fitness of solutions generated by different ants is now computed using Eq. (2). The solution with highest fitness value is allowed to update the pheromone trail, which is called global trail updating. The global pheromone trail updating is intended to strengthen the edges belonging to the solution having minimum makespan value in the current iteration. The global updating is done by Eq. (6)

Thus, the algorithm essentially works as follows: *A* number of ants start with empty solution strings. Each ant builds a solution by repeatedly applying a stochastic greedy rule (process (a)) and/or random probabilistic rule (process (b)). While constructing its solution, an ant also modifies the amount of pheromone on the chosen edges by applying the local updating rule. Once all the ants have build their solution, the amount of pheromone on edges is modified again by applying global updating rule.

4.2. *Use of genetic operators*

The algorithm continuously develops new solutions in order to find an optimal/near-optimal sequence of jobs with minimum makespan. In order to generate new region for ant tour, we had used the crossover and mutation genetic operators. We had performed corresponding GA using PMX and OX operators as advocated by Jung et al. The crossover operation is done as follows. Select a parent randomly and set the first variable of the child's position vector (which for the present problem is the vessel size for the first stage) same as that of the first element of the parent's positions vector. The subsequent values of the variables of the child are set to the corresponding values of a randomly chosen parent with a probability equal to the crossover probability (CP). A cross over probability of one means that each element of the child's position vector has a different parent. A CP of zero means that the child region has all the elements same as the chosen single parent. After the crossover step, mutation is carried out by randomly adding or subtracting a value to each and every variable of the newly created region with a probability equal to a suitably defined mutation probability. The mutation step size is reduced as per the relation:

$$\Delta(T,R) = R(1-\Upsilon^{(1-T)^b}) \tag{8}$$

where Υ is a random number from [0, 1], R is the maximum step size, T is the ratio of the current iteration number to that of the total number of iterations and b is a positive parameter controlling the degree of nonlinearity. The nonlinear reduction in mutation span reflects the decreased need for global search as the iteration proceeds and enhances the probability of locating maximum by concentrated search around a reduced search radius. In trail diffusion, which is another element in global search carried out by the ant algorithm, two parents are selected at random from the parent regions. The variables of the child's position vector can have either (1) the value of the corresponding variable from first parent; (2) the corresponding value of the variable from second parent, or (3) a combination arrived from a weighted average of the above

$$x(\text{child}) = (\alpha)x_i(\text{parent 1}) + (1-\alpha)x_i(\text{parent 2}) \tag{9}$$

where α is a uniform random number in the range [0, 1]. The probability of selecting third option is set equal to the mutation probability while allotting equal probability of selecting the first two steps. Thus, if mutation probability is 0.6 the probability of selecting third option is 0.6 while the probability of selecting option 1 or 2 is 0.2 each. The trail value of the newly created child regions is assigned a trail value lying between the values of the original parent regions. The pheromone values are decreased after each iteration by

$$\tau_i(t+1) = \rho\tau_i(t) \tag{10}$$

The ant colony was run for the problem $N = 8$ and $M = 4$ with the process time as shown in Table 3 for 150 iterations and we got different sequences which satisfy the criterion. The Gantt chart for one of the solution which gives the optimal completion time 142 with the sequence 5, 4, 1, 6, 3, 7, 2, 8 is shown in Fig. 3.

Table 4 shows the pheromone matrix for the sequence generated by ants for the solution shown in Fig. 4. The bold numbers in Table 4 shows the affection of a job towards a position in the sequence. More the concentration of pheromone value more the possibility of passing the ants through that position.

Table 3
Process data.

Products	Units			
	U1	U2	U3	U4
P1	10	20	5	30
P2	15	8	12	10
P3	20	7	9	5
P4	13	7	17	10
P5	8	3	16	7
P6	6	9	22	7
P7	7	5	15	12
P8	14	13	6	4

5. Data mining

The different sequences obtained from the ant colony algorithm for 8 × 4 flow shop problem should be properly structured in order to mine. The classification of the data was done on some attributes. These attributes contribute mainly in the scheduling operation. For this we use the characteristics processing time,

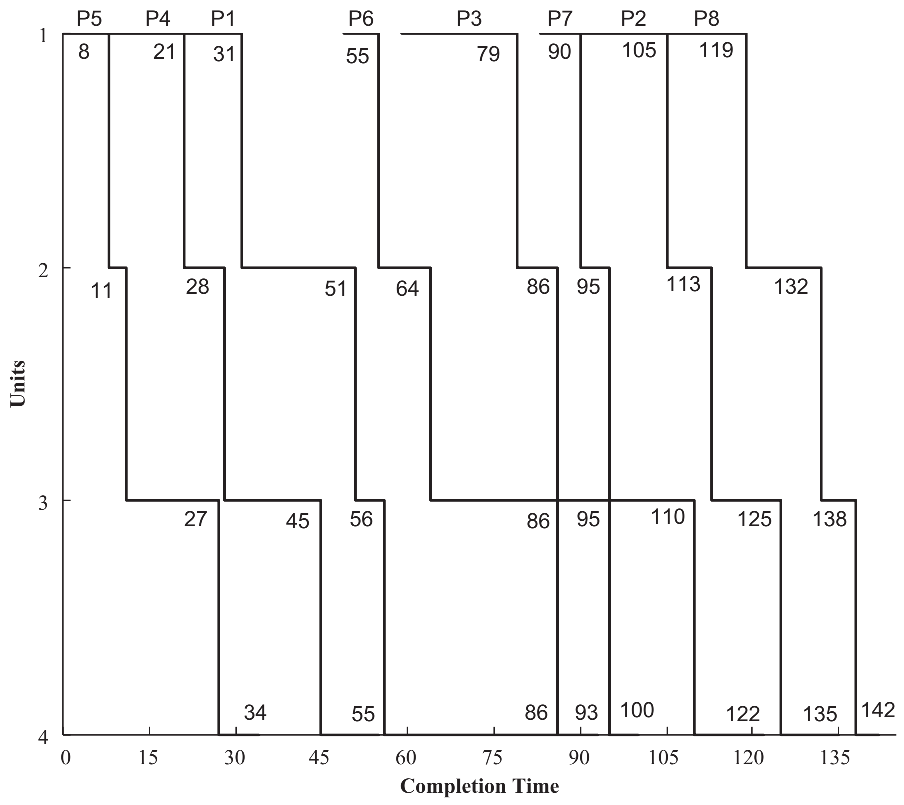

Fig. 3. Gantt chart for ZW policy for problem shown in Table 3.

Table 4
Pheromone matrix for solution for the 8 × 4 flow shop scheduling problem for the sequence shown in the Gantt chart shown in Fig. 3.

Job no.	Pheromone value for each position							
	P1	P2	P3	P4	P5	P6	P7	P8
J1	0.00566385	0.0337077	**0.171359**	0.00515377	0.0113062	0.00580029	0.00832053	0.00515377
J2	0.00582081	0.00588679	0.00572423	0.00823283	0.00566385	0.0404619	**0.16911**	0.00556427
J3	0.00691272	0.00768597	0.0143749	0.0401588	**0.354684**	0.00734112	0.00588679	0.128636
J4	0.00515377	**0.126699**	0.00515377	0.00515377	0.0058497	0.0516028	0.0416981	0.00515377
J5	**0.199369**	0.00515377	0.00515377	0.00549667	0.00515377	0.00778495	0.00515377	0.0131995
J6	0.00515377	0.016125	0.0337077	**0.169428**	0.00618129	0.00515377	0.00515377	0.00556164
J7	0.0128565	0.0460527	0.00553477	0.00768729	0.0410516	**0.121881**	0.00515377	0.00624755
J8	0.00553477	0.00515377	0.00545694	0.00515377	0.0013579	0.00643934	0.00598799	**0.0769481**

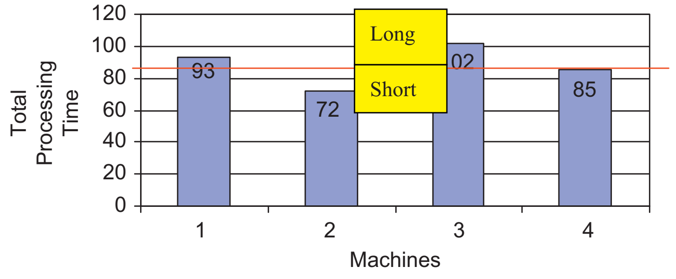

Fig. 4. Machine loading for 8 × 4 problem.

remaining process time (RPT), position of operation in its job (POJ), time length of a job (TLJ) and machine loading. Data mining is an automated process of extracting structured knowledge from databases, which is often referred to as a particular step in the overall process of discovering useful knowledge from data, called knowledge discovery from database (KDD) [13]. Data mining is a step of a long process chain of data analyzing that involves the evaluation and interpretation of pattern to determine what constitutes knowledge. Data mining is an interdisciplinary field, whose core is at the intersection of machine learning, statistics and database. There are several data mining tasks, including classification, regression, clustering, dependence modeling, etc. Among these tasks, the goal of classification is to assign each case (object, record, or instance) to one class, out of a set of predefined classes, based on the values of some attributes (called predictor attributes) for the case. In the classification task, the discovered knowledge is often expressed in the form of IF-THEN rules, which have the advantage of being a high-level and symbolic knowledge representation contributing towards the comprehensibility of the discovered knowledge (Fig. 5).

The main purpose of applying the data mining in this paper is to predict for each operation of the multi-product batch processing flow shop its affection order on its machine according to the

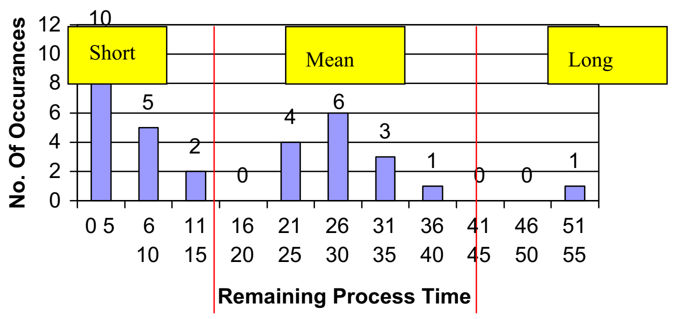

Fig. 5. Remaining time classifications.

Table 5
Different sequences from ant system.

No.	Optimal sequence of jobs	Completion time
1	5,4,1,6,3,7,2,8	142
2	5,2,4,1,6,3,7,8	142
3	7,1,5,6,4,2,8,3	142
4	7,5,6,4,2,8,1,3	142

various characteristics. We use in this paper the supervised learning process called also concept learning from example, consists in characterizing pre-classified objects by a supervisor in one or several classes under some attributes. The process of classification amounts in three stages and these are creation of a classifier from data set, the classification of the new instances via the classifier and evaluation of the performance of this last one. The ant colony was run for the problem $N=8$ and $M=4$ with the process time as shown in Table 3 for 150 iterations and 16 distinct sequences are found which satisfy the criterion. Out of these some are shown in Table 5.

The properly structured attributes are transferred from continue attribute to discreet attribute in order to obtain a general model which will be applicable to a vast set of scheduling flow shop problems. For discretization the chi2 algorithm [14–16] was used. The chi2 algorithm is based on the X^2 statistic and consists of two phases. In the first phase it begins with a high significance level say 0.5 (sigLevel) for all numeric attributes for discretization. Each attribute is sorted according to its value. Then X^2 value is calculated by Eq. (11) given below for every pair of adjacent intervals and the merging is done with the pair of adjacent interval with lowest X^2 value. This merging continues until all pair of adjacent intervals has X^2 values exceeding the parameter determined by sigLevel.

$$\chi^2 = \sum_{i=1}^{m}\sum_{j=1}^{k}\frac{(A_{ij}-E_{ij})^2}{E_{ij}} \tag{11}$$

where k is the number of classes; $m=2$ (the intervals being compared); A_{ij} is the no. of patterns in the ith interval, jth class; R_i is the no. of patterns in the ith interval; C_j is the no. of patterns in the jth class; N is the total no. of patterns; E_{ij} is the expected frequency of $A_{ij}=R_i{*}C_j/N$.

Tables 6 and 7 show the discretization results obtained by chi2 algorithm. Further with chi2 algorithm the four classes of operation position are classified as

- First: the first position of affection order in the machines.
- Middle: the second and the third position of affection order in the machine.
- Last: the fourth position of affection order in the machine.

Table 6
Discretization attributes with Chi2 algorithm.

Attributes	Short	Mean	Long
PT	[1,8]	[9,18]	[19,30]
RPT	[0,18]	[19,37]	[38,56]
UPT	[32,40]	[41,52]	[53,65]
ML	[30,87]	–	[88,110]

This structured data was manipulated by See5 classifier system, which produced a decision tree or a decision rule. The induced decision rules are all of the same if-then structure and will be background of a new metaheuristic of flow shop scheduling. We give here the obtained rules given by See5 software [17] when the rate given in the end of each rule is the rate of good classification.

5.1. Rule induction

The data mining was performed for the problem 8×4 which has optimum completion time 142. The rules generated from the See5 data mining tool are given below. The program was run with 50 boosting trials and 90% of the training data was used. Further the in each trial cross validating was done with 90 folds. The winnow attribute option was used and thresholds probability selected with total 374 training cases. See5 has taken total 8.5 s to generate rules. The rules inducted were,

Rule 01: (93/7, lift 3.7)
  Unit process time $\Leftarrow$ long
  Remain process time $\Leftarrow$ long
  Machine load $\Leftarrow$ short
    Job $>$ 3
    Job $\Leftarrow$ 5
    $\rightarrow$ Class first [0.916]
Rule 02: (48, lift 3.8)
  Unit process time $\Leftarrow$ long
  Remain process time $\Leftarrow$ mean
  Machine load $\Leftarrow$ short
    Job $\Leftarrow$ 1
    $\rightarrow$ Class middle [0.980]
Rule 03: (45, lift 4.0)
  Unit process time $\Leftarrow$ short
  Remain process time $\Leftarrow$ Long
  Machine load $>$ short
    Job $>$ 7
    $\rightarrow$ Class last [0.979]
Rule 04: (44, lift 3.9)
  Unit process time $\Leftarrow$ short
  Remain process time $\Leftarrow$ mean
  Machine load $>$ short
    Job $>$ 2
    Job $\Leftarrow$ 3
    $\rightarrow$ Class later [0.978]
Rule 05: (47/7, lift 3.4)
  Unit process time $\Leftarrow$ mean
  Remain process time $\Leftarrow$ mean
  Machine load $>$ short
    Job $>$ 6
    Job $\Leftarrow$ 7
    $\rightarrow$ class later [0.837]
Rule 06: (48/8, lift 3.2)
  Unit process time $\Leftarrow$ long
  Remain process time $\Leftarrow$ mean

Table 7
Detailed classification of attributes with Chi2 algorithm.

Job	Operation	Machine	Process time	Remaining process time	Operation'	Process time'	Remaining process time'	Machine loading
1	1	1	10	55	First	Mean	Long	Long
1	2	2	20	35	Middle	Long	Mean	Short
1	3	3	5	30	Middle	Short	Mean	Long
1	4	4	30	0	Last	Long	Short	Short
2	1	1	15	30	First	Mean	Mean	Long
2	2	2	8	22	Middle	Short	Mean	Short
2	3	3	12	10	Middle	Mean	Short	Long
2	4	4	10	0	Last	Mean	Short	Short
3	1	1	20	21	First	Long	Mean	Long
3	2	2	7	14	Middle	Short	Short	Short
3	3	3	9	5	Middle	Mean	Short	Long
3	4	4	5	0	Last	Short	Short	Short
4	1	1	13	34	First	Mean	Mean	Long
4	2	2	7	27	Middle	Short	Mean	Short
4	3	3	17	10	Middle	Mean	Short	Long
4	4	4	10	0	Last	Mean	Short	Short
5	1	1	8	26	First	Short	Mean	Long
5	2	2	3	23	Middle	Short	Mean	Short
5	3	3	16	7	Middle	Mean	Short	Long
5	4	4	7	0	Last	Short	Short	Short
6	1	1	6	38	First	Short	Long	Long
6	2	2	9	29	Middle	Mean	Mean	Short
6	3	3	22	7	Middle	Long	Short	Long
6	4	4	7	0	Last	Short	Short	Short
7	1	1	7	32	First	Short	Mean	Long
7	2	2	5	27	Middle	Short	Mean	Short
7	3	3	15	12	Middle	Mean	Short	Long
7	4	4	12	0	Last	Mean	Short	Short
8	1	1	19	23	First	Long	Mean	Long
8	2	2	13	10	Middle	Mean	Short	Short
8	3	3	6	4	Middle	Short	Short	Long
8	4	4	4	0	Last	Short	Short	Short

```
    Machine load >short
      Job>5
      Job⇐6
      →Class middle [0.820]
  Rule 07: (45/6, lift 3.5)
    Unit process time⇐long
    Remain process time⇐short
    Machine load>short
      Job>1
      Job⇐2
      →Class last [0.851]
```

These rules inducted were having error percentage 7.75 as shown in above. This is due to misclassification of the 29 attributes in the training cases. For the decision tree, total 10 boosting trails were performed without the global pruning. The result of the rule is that if a job having unit process time is long, remain process time is long, and the machine load is short then classifies that job as first.

5.2. Application to other cases

The applicability of these rules was tested using five 8×4 flow shop test cases generated at random. The process time for these cases is shown in Table 7. These cases were all scheduled using the ant colony algorithm. Table 8 shows that in 3 out of 5 cases, the learned rules were able to produce approximately optimal completion time nearer to the optimal.

Table 8
Completion time of the schedules generated by ant colony and discovered rules.

Case no.	Ant colony		Rule based	
	Completion time	Optimal sequence	Completion time	Sequence
1	138	5,6,1,2,4,3,7,8	144	5,4,1,6,3,7,2,8
2	155	2,3,8,6,5,1,4,7	166	2,3,8,5,1,4,6,7
3	183	4,2,3,1,8,7,5,6	189	4,2,3,7,5,8,1,6
4	188	7,4,2,5,1,8,6,3	194	8,7,4,2,5,1,6,3
5	140	5,4,3,8,6,1,2,7	140	5,4,1,2,7,3,8,6

6. Conclusions

In this paper, we propose an approximate method to resolve a multi-product batch flow shop schedule. We propose an ant colony algorithm that is fast and easy to implement to generating a learning population. Further we propose another method of knowledge extraction that will generate decision rules finding the affection order of operation on all machines. We studied several benchmark problems of Taillard [2] and we got very encouraging results. The results have shown that data mining can be used to learn from flow shop schedules produced by ant colony algorithms. Data mining requires an understanding of the problem domain, knowledge of mining algorithms, and an insight into which attributes might be significant. In this project, when compared to problems with the same structure (8×4 flow shop) and different operation times and sequences, the rules were able to consistently perform the heuristic rules. However, the learned rules were unable to match the performance of the ant colony algorithm on these problems. We obtained an approach solution with a distance of 7.04% percent from the optimal. This study envisages the application of DM tool for extraction of knowledge patterns of scheduling in a job shop or flow shop. The results of this study indicate that production managers should develop a rule that depends upon the characteristics of their own production system by using a data mining tool (rule induction or decision tree). By analyzing the historical production, data the scheduling knowledge can be extracted and then be expressed in IF-THEN rules. This form of representing Knowledge provides clear indications as to which factors are most influential in predicting the scheduling and how various levels of critical factors affect the scheduling. At the same time, the scheduling knowledge can significantly improve.

References

[1] Koonce DA, Tsai SC. Using data mining to find patterns in genetic algorithm solutions to a job shop schedule. Computers & Industrial Engineering 2000;38:361–74.
[2] Taillard ED. Benchmarks for basic scheduling problems. European Journal of Operation Research 1993;64:278–85.
[3] Dorigo M, Maniezzo V, Colorni A. The ant system: optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man & Cybernetics B 1996;26(2):29–41.
[4] Rock Hans. The three-machine no-wait flow shop is NP-complete. Journal of the Association for Computing Machinery 1984;31(2):336–45.
[5] Tra Niem-Trung L, Comparison of Scheduling Algorithms for a Multi-Product Batch-Chemical Plant with a Generalized Serial Network. MS Thesis, The Virginia Polytechnic Institute and State University, Blacksburg, Virginia, 2000.
[6] Wang Xiao-Rong, Wu Tie-Jun, An ant colony optimization approach for no-wait flow line batch scheduling with limited batch size. In: Proceedings of the 42nd IEEE conference on decision and control Maui, Hawaii USA, December 2003.
[7] Gupta, Srusevich VA, Zwaneveld C. Two-stage no wait scheduling models with setup and removal times. Computers and Operations Research 1997;24(11): 1025–31.
[8] Gupta, Tunc EA. Scheduling a two-stage hybrid flowshop with separable setup and removal times. European Journal of Operational Research 1994;77(3):415–28.
[9] Dileepan P, Sen T. Job lateness in a two-machine flowshop with setup times separated. Computers and Operations Research 1991;18(6):549–56.
[10] Cheng, Wang. A note on two-machine flexible manufacturing cell scheduling. IEEE Transactions on Robotics and Automation 1999;15(1):187–90.
[11] Yamada Takeshi. Solving the Csum permutation flowshop scheduling problem by genetic local search. IEEE International Conference on Evolutionary Computation 1998:230–4.
[12] Jayaraman VK, Kulkarni BD. Ant colony framework for optimal design and scheduling of batch plants. Computers and Chemical Engineering 2000;24:1901–12.
[13] Fayyad UM, Data mining and knowledge discovery in data bases. In: Proceedings of ninth international conference on scientific and statistical database management 1997. p. 2–11.
[14] Liu Huan, Setiono Rudy, Chi2: feature selection and discretization of numeric attributes. In: Proceedings of IEEE seventh international conference on tools with Artificial Intelligence.
[15] Boulle M. Khiops: a statistical discretization method of continuous attributes. Machine Learning 2004;55:53–69 [Kluwer Academic Publishers].
[16] Liu Huan, Hussein Farhad, Tan Chew Lim, Das Manoranjan. Discretization: an enabling technique. Data Mining and Knowledge Discovery 2002;6:393–423 [Kluwer Academic Publishers].
[17] ⟨http://www.rulequest.com/see5⟩.