Int J Adv Manuf Technol (2004) 24: 376–380
DOI 10.1007/s00170-003-1665-3

ORIGINAL ARTICLE

A. Noorul Haq · D. Ravindran · V. Aruna · S. Nithiya

A hybridisation of metaheuristics for flow shop scheduling

Received: 30 August 2002 / Accepted: 28 January 2003 / Published online: 5 August 2004

Abstract The present paper deals with the formation of an optimal sequence of flow shop scheduling (FSS) for efficient operation. The primary concern of FSS is to obtain the optimal sequence, which minimises the idle time, tardiness, makespan, etc. Among these, the criteria of minimising the makespan plays a vital part. Thus, in this paper, the sequencing of the FSS for minimising the makespan is addressed. An effective hybrid has been formed with the metaheuristics, namely an ant system and a genetic algorithm (GA). A number of illustrative examples with different combinations of machines and jobs have been solved using the proposed hybrid method.

Keywords Flow shop scheduling · Metaheuristics · Makespan · Ant system · Genetic algorithm · Hybridisation

1 Introduction

A flow shop is characterised by a unidirectional flow of work with a variety of jobs being processed sequentially in a one-pass manner. In many manufacturing and assembly facilities a number of operations needs to be done on every job. A "job" is thus a collection of operations to be performed on an item or unit with applicable technological constrains. This implies that all jobs have to follow the same route, even if the jobs are identical. The machines are set up in a series and such a processing environment is referred to as a flowshop.

Shop schedules are generally evaluated by aggregate quantities that involve information about all jobs, resulting in a one-dimensional performance measure usually expressed as a function of the set of job completion or processing times. A performance measure frequently used to compare the quality of two schedules is a makespan, the total time required to process all the jobs. Other common measures include the mean flow time and the job tardiness. Although the different performance measures are often correlated, one can construct an example for which a schedule may be good according to one measure, but which may perform poorly on others. In this paper, it has been considered desirable to investigate the optimal sequence, which minimises the makespan for the flow shop scheduling (FSS). A heuristic algorithm namely, the ant system (AS) is employed to minimise the objective function namely the makespan. Further, its output is given to an evolutionary technique, namely a genetic algorithm (GA) to optimise the main criteria.

A. N. Haq (✉) · D. Ravindran · V. Aruna · S. Nithiya
Department of Production Engineering,
Regional Engineering College,
Tiruchirappalli 620 015, India
E-mail: anhaq@nitt.edu

2 The literature survey

Janiak in his work [1] addresses the problem where some of the job operations' processing times are a convex decreasing function of the amounts of resources allocated with the objective of minimising the makespan.

Leslie A.Hall work [2] involves a polynomial approximation scheme for a two-machine flow shop scheduling problem with the additional constraint that each job has a release date, when it first becomes available for processing.

Celia A. Glass, Chris N. Potts and Vitaly A. Strusevich in their work [3] deal with batching of operations of jobs—a key issue in machine scheduling. The objective is to minimise makespan and processing time of a batch on a machine.

Okamoto Shusuke, Watanabe Chie and Iizuka Hajime presented a parallel algorithm for solving *n*-job, *m*-machine flow shop problems [4] based on the parallelisation of the branch and bound method along

with all search methods to minimise the processing time.

Mohamed Haouari and Thouraya Daouas proposed [5] an exact branch and bound method based on a recursive enumeration of potential inputs and outputs of machines for three-machine assembly type flow shop problem.

Victor Portugal and J. L. Scott in their work [6] address the problem of FSS with the objective of minimising the makespan and the maximum tardiness. Thomas Stutzle and Hoos applied the MAX-MIN ant system [7], one of the most efficient ant colony optimisation algorithm to the traveling salesman problem.

Riad Aggoune's work [8] deals with the scheduling of flow shop with the availability constraints (FSPAC) where two pre-emptive FSPAC with arbitrary number of machines and arbitrary number of unavailability periods on each of them are considered. A GA and tabu search are used to attain the objective.

An evolutionary search approach, a GA for job shop scheduling problems is given by Kobayashi, Ono and Yamamura [9]. K.J. Shaw and P.J. Fleming's paper [10] reports that applying a GA to scheduling problems is a powerful technique which has the potential to be used as a multi-objective optimisation tool in production manufacturing applications.

S. Chen and S. Smith [11] considered the precedence constraints for the search space to minimise tardiness and earliness cost. Johann Hurink and Sigrid Knust [12] deal with flow shop problems involving transportation times where all transportations are done by a single robot.

From the literature survey undergone, the most important criteria in flow shop scheduling is found to be the minimisation of makespan. It can also be seen that not much concentration has been given on the hybridization of heuristics. Therefore, in this paper, an attempt has been made to hybridise the AS and the GA for the flow shop problem with the minimum makespan as objective.

3 Problem statement

The objective of the problem is to minimise the makespan. The problem is defined as follows:

a) Given the processing time (time matrix) of 'n' jobs on 'm' machines, first the best sequence of the jobs is determined according to the probability function which comprises of
   - Intensity of trial
   - Evaporation of trial
   - Visibility

   using the heuristic AS algorithm.

b) This optimum job sequence is given as input to an evolutionary search technique, namely; a GA due to which the more optimal sequence is generated.

4 The AS

An AS is a new general purpose heuristic, which can be used to solve different combinatorial problems. Given a set of machines, jobs and processing time of job 'i' on machine 'j' (t_{ij}), flow shop scheduling can be stated as the problem of finding the minimal processing time such that all the jobs are processed in all the machines.

Here the number of ants are taken as equal to the number of the jobs at time 't'.

The algorithm [13] is stated as follows:

1 Initialise

Set: $t = 0$ (t is the time counter)
Set: $NC = 0$ (NC is the cycles counter)
For every successive jobs (i,j) set an initial value $\tau_{ij}(t) = c$ for trial intensity and $\Delta\iota_{ij} = 0$
Place the m ants on the n nodes.

2 Set $s := 1$ (s is the tabulist index)

For $k := 1$ to m do
Place the starting job of the kth ant in $tabu_k(s)$

3 Repeat until the tabulist is full (this step will be repeated $(n–1)$ times)

Set $s := s + 1$
For $k := 1$ to m do
Choose the job 'j' to be processed, with probability $p^k{}_{ij}(t)$

$$p_{ij}^k(t) = \left[\tau_{ij}(t)\right]\left[\eta_{ij}\right] / \sum \left[\tau_{ik}(t)\right]\left[\eta_{ik}\right] \text{ if } j \in allowed_k$$

$k \in allowed_k$
where η_{ij}—visibility

4 For k: = 1 to m do

Compute the processing time L_k of the tour described by the kth ant
Update the shortest processing time found.
For any successive jobs (i,j)
For k: = 1 to m do

$$\Delta l_{ij}^k = \begin{cases} Q/L_k \text{ if the kth ant uses successive jobs } (i,j) \text{ in its tour} \\ 0, \text{ otherwise} \end{cases}$$

where $\Delta\iota^{k}_{ij}$—quantity per unit length of trial substance.

$$\Delta\iota_{ij} = \Delta\iota_{ij} + \Delta\iota^{k}_{ij}$$

5 For every successive jobs (i,j) compute $\iota_{ij}\ (t+n)$ according to the equation

$$\iota_{ij}(t+n) = \rho\iota_{ij}(t) + \Delta\iota_{ij}$$

Set $t := t+n$
Set $NC := NC+1$
For every successive job (i,j) set $\Delta\iota_{ij}=0$

6 If $(NC < NC_{max})$ and (not stagnation behaviour)

Then
Empty all the tabulists
Go to step 2
Else
Print shortest sequence
Stop

5 GAs

GAs mimic the principles of natural genetics and natural selection to constitute search and optimisation procedures. GAs differing from conventional search techniques start with an initial set of random solutions called population. Each individual in the population is called a chromosome (a string of symbols), representing a solution to the problem at hand. The chromosomes evolve through successive iterations, called generations. During each generation, the chromosomes are evaluated using some measures of fitness. To create the next generation, new chromosomes called offspring, are formed by

(a) Merging two chromosomes from current generation using a crossover operator or
(b) Modifying a chromosome using a mutation operator

Formation of new generation involves (a) selecting some of the parents and offspring according to the fitness values and (b) rejecting others so as to keep the population size constant. After several generations, the algorithm converges to the best chromosome.

The steps involved in GAs [14] are as follows:

Step 1: choose a coding to represent problem parameters, a selection operator, a crossover operator, and a mutation operator. Choose the population size, n, crossover probability P_c and mutation probability P_m. Initialise a random population of strings of size L. Choose a maximum allowable generation number t_{max}. Set $t=0$.
Step 2: evaluate each string in the representation.
Step 3: if $t > t_{max}$ or any other termination criteria is satisfied, terminate.
Step 4: perform reproduction on the population.
Step 5: perform crossover on random pairs of strings.
Step 6: perform mutation on every string.

Table 1 The processing time for jobs in machines

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
J_1	9	17	4	5	11	8	14	6	7	12
J_2	21	6	7	18	1	12	5	9	14	8
J_3	25	15	14	24	22	12	20	22	23	13
J_4	16	25	26	10	13	29	12	2	6	12
J_5	8	3	6	12	13	4	6	3	9	12
J_6	4	14	10	6	7	12	12	7	2	10
J_7	18	7	5	9	12	4	6	6	10	9
J_8	5	13	11	7	2	12	9	10	4	10
J_9	17	5	8	9	13	6	4	14	9	8
J_{10}	3	12	15	6	10	11	9	18	7	10

Table 2 The sequence of jobs after the first iteration in the ant system

Seq. no.	Order of jobs									
1	J_1	J_5	J_8	J_6	J_7	J_9	J_{10}	J_2	J_4	J_3
2	J_2	J_5	J_8	J_6	J_7	J_9	J_1	J_{10}	J_4	J_3
3	J_3	J_5	J_8	J_6	J_7	J_9	J_1	J_{10}	J_2	J_4
4	J_4	J_5	J_8	J_6	J_7	J_9	J_1	J_{10}	J_2	J_3
5	J_5	J_8	J_6	J_7	J_9	J_1	J_{10}	J_2	J_4	J_3
6	J_6	J_5	J_8	J_7	J_9	J_1	J_{10}	J_2	J_4	J_3
7	J_7	J_5	J_8	J_6	J_9	J_1	J_{10}	J_2	J_4	J_3
8	J_8	J_5	J_6	J_7	J_9	J_1	J_{10}	J_2	J_4	J_3
9	J_9	J_5	J_8	J_6	J_7	J_1	J_{10}	J_2	J_4	J_3
10	J_{10}	J_5	J_8	J_6	J_7	J_9	J_1	J_2	J_4	J_3

Table 3 The random input to the genetic algorithm

Seq. no.	Order of jobs									
1	J_1	J_5	J_9	J_8	J_{10}	J_6	J_7	J_2	J_3	J_4
2	J_2	J_4	J_5	J_6	J_7	J_8	J_9	J_{10}	J_3	J_1
3	J_3	J_1	J_4	J_5	J_8	J_{10}	J_7	J_6	J_2	J_9
4	J_4	J_8	J_5	J_1	J_9	J_6	J_2	J_3	J_7	J_{10}
5	J_5	J_6	J_4	J_7	J_8	J_9	J_2	J_{10}	J_1	J_3
6	J_6	J_7	J_8	J_9	J_{10}	J_1	J_2	J_3	J_4	J_5
7	J_7	J_8	J_9	J_4	J_5	J_6	J_1	J_2	J_3	J_{10}
8	J_8	J_9	J_4	J_5	J_7	J_1	J_2	J_3	J_6	J_{10}
9	J_9	J_1	J_4	J_5	J_2	J_3	J_6	J_7	J_{10}	J_8
10	J_{10}	J_1	J_2	J_3	J_4	J_5	J_8	J_7	J_9	J_6

Table 4 A comparison of results

Serial No.	No. of jobs	No. of machines	Percentage of times best makespan yielded		
			Ant system	Genetic algorithm	Hybrid (AS and GA)
1	5	5	10	90	100
2		10	10	100	90
3		15	10	100	100
4		20	0	100	100
5		25	20	100	100
6		30	50	100	100
1	10	5	10	50	70
2		10	0	40	70
3		15	0	70	40
4		20	0	20	80
5		25	20	70	30
6		30	0	40	60
1	15	5	30	60	90
2		10	10	50	70
3		15	0	60	40
4		20	0	40	70
5		25	0	10	90
6		30	0	60	40
1	20	5	10	50	50
2		10	30	20	80
3		15	20	20	100
4		20	0	40	60
5		25	0	50	50
6		30	0	20	80
1	25	5	0	50	60
2		10	0	30	70
3		15	0	40	60
4		20	0	40	60
5		25	0	50	50
6		30	0	40	60
1	30	5	0	40	70
2		10	0	20	80
3		15	10	30	70
4		20	0	40	60
5		25	0	20	80
6		30	0	20	80

Step 7: evaluate strings in the new population. Set $t = t + 1$ and go to step 3.

6 An illustrative example of the AS

To illustrate the flow shop scheduling, let there be 10 jobs to be processed on 10 machines. The time matrix (i.e., the processing time of 10 jobs on 10 machines) is given in Table 1. In the AS, the sequences equivalent to the number of jobs are generated. The AS can therefore process the 10 jobs on 10 machines in any of the 10 possible combinations as shown in Table 2 (first iteration). After several iterations, the optimal sequence is

$J_3\ J_4\ J_2\ J_{10}\ J_1\ J_9\ J_7\ J_6\ J_8\ J_5$

The makespan for the above sequence is 281.

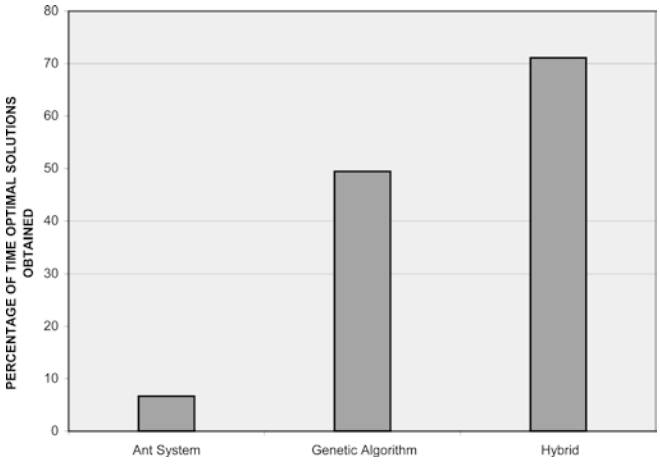

Fig. 1 The best minimum makespan

7 An illustrative example of a GA

In a GA, the input is given as random as shown in Table 3 and the parents for the crossover and mutation are selected based on the respective probabilities.

This procedure is repeated for several iterations and the optimal sequence with the minimum makespan is found to be

$J_8\ J_5\ J_6\ J_{10}\ J_3\ J_1\ J_4\ J_2\ J_7\ J_9$ with makespan 261.

8 An integration of the AS and a GA

On giving the output of the AS as an input to the evolutionary search technique-GA, the optimal sequence is obtained.

The optimal sequence is $J_{10}\ J_8\ J_6\ J_5\ J_3\ J_9\ J_4\ J_7\ J_2\ J_1$ with makespan 259.

The hybrid technique is evaluated for a large number of flow shop problems of varying sizes i.e., 360 problems in total with jobs varying from 5 to 30 and machines varying from 5 to 30 in steps of 5.

It has been observed from Table 4 that the hybridisation of metaheuristics has given better results than the pure metaheuristics. A hybrid has given best minimum makespan for 71.11% of time whereas the AS for 6.67% of the time and genetic algorithm for 49.44% of time as shown in Fig 1.

9 Conclusions

This paper has focused on minimising the makespan for the flow shop scheduling problem. These problems are modelled first; subsequently solution schemes are chosen and then analysed in detail. In this context, the AS and the evolutionary search technique called a GA are approached separately towards the objective and then compared with the hybrid of both these techniques. The heuristic approach presented in this paper is not limited to the specific scenario and can be extended to the general '*n*' jobs and the '*m*' machines case.

The hybrid approach has been shown to be more productive than the pure heuristic methods to combinatorial problems. The approach can also be extended to the objectives such as minimising the tardiness, meeting due dates, maximising the machine utilisation, minimising time lags, etc. Similar hybrids can be formed from various heuristics for meeting the objectives.

References

1. Janiak A (1997) Flow shop scheduling with resources—exact and approximation algorithms. In: Proceedings of the International Symposium on Mathematical Programming, Lausanne, Switzerland, 24–29 August 1997
2. Hall LA (1994) A polynomial approximation scheme for a constrained flow shop scheduling problem. Math Oper Res 19:68–85
3. Glass CA, Potts CN, Strusevich VA (1997) Shop scheduling with batching to minimise the makespan. In: Proceedings of the International Symposium on Mathematical Programming, Lausanne, Switzerland, 24–29 August 1997
4. Shusuke O, Chie W and Hajime I (2001) A new parallel algorithm for *n* job *m* machine flow shop scheduling problem and its implementation on nCUBE2. IPSJ SIGNotes Contents Software Engineering, Abs. No. 087–006
5. Haouari M. Daouas T (1999) Optimal scheduling of the 3 machine assembly type flow shop. RAIRO Ops Res 33(4):439–445
6. Portugal V, Scott JL (2001) The asymptotic convergence of some FSS heuristics. Asia Pacific J Op Res 18(2):
7. Stutzle T, Hoos H (1999) MAX-MIN Ant System and Local Search for the Traveling Salesman Problem. In: Proceedings of the IEEE Conference on Evolution Computation (ICEC '97), Indianapolis, IN, 13–16 April 1999
8. Aggoune R (2001) Minimizing the makespan for flow shop scheduling problem with availability constraints. In: Proceedings of ORP–2001, Paris, France, 26–29 September 2001
9. Kobayashi S, Ono I, Yamamura M (1995) An efficient GA for job shop scheduling problems. In: Proceedings of ICGA '95, Pittsburgh, PA, 15–19 July 1995
10. Shaw KJ, Fleming PJ (1996) An initial study of practical multiobjective production scheduling using genetic algorithms. In: Proceedings of the International Conference on Control '96, University of Exeter, UK, 2–5 September 1996
11. Chen S, Smith S (1999) Improving GAs by search space reduction with application to flow shop scheduling. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '99), Orlando, FL, 14–17 July 1999
12. Hurink J, Knust S (2001) Makespan minimization for flow shop problems with transportation times and a single robot. Disc Appl Math 112:199–216
13. Dorigo M, Colorni A (1996) The ant system: optimization by a colony of co-operating agents. IEEE Trans Sys Man Cybern 26(1):1–13
14. Gen M, Cheng R (1996) Genetic algorithms and engineering design. Ashikaga Institute of Technology, Wiley, New York