SIMULTANEOUS SCHEDULING OF MACHINES AND AGVs USING CROW SEARCH ALGORITHM: A NEW NATURE-INSPIRED META-HEURISTIC

[1]Sivarami Reddy N, [2]Dr. Ramamurthy DV and [3]Dr. Prahlada Rao K

[1]Research scholar, JNTUA, Ananthpuram, AP, [2]Principal, GIET, Rajahmundhry
[3]Professor, Mech. Engg. JNTUACEA, Ananthapuram
[1]E-mail: siva.narapureddy@gmail.com

Abstract: *This paper addresses the problem of simultaneous scheduling of machines and two identical automated guided vehicles (AGVs) in a flexible manufacturing system (FMS). It is a NP–hard problem which is very complex. For solving this problem, a new nature inspired meta-heuristic Crow Search Algorithm (CSA) is proposed. The problem consists of two interrelated problems, scheduling of machines and scheduling of AGVs. A simultaneous scheduling of these, in order to minimize the makespan will result in an FMS being able to complete all the jobs assigned to it at earliest time possible, thus saving resources. Improvement in performance of FMS can be expected by efficient utilization of its resources, by proper integration and synchronization of their scheduling. The proposed heuristic is tested on problems generated by various researchers and the results are compared with the results of existing methods. The results show that the proposed heuristic outperforms the existing methods.*

Keywords: *Flexible Manufacturing Systems, Crow Search Algorithm, Simultaneous Scheduling of Machines and AGVs, Minimization of Makespan*

1. INTRODUCTION

FMS is an integrated manufacturing system which incorporates many modern facilities such as Computer Numerically Controlled (CNC) machines, Automated Guided Vehicles (AGVs), Automated storage/ Retrieval Systems (AS/RS), Central Tool Magazine (CTM), Robots and Automated inspection using machine vision system under the control of a central computer [3,1]. Various subsystems flexibilities are integrated together in creating an overall flexibility in FMS. One of the modern techniques in industrial automation is FMS, and many researchers have been attracted towards FMS over the last three decades. FMS has many advantages such as greater productivity, minimum work-in-process inventory, high machine utilization, production with minimum supervision, increased product variety and high quality to satisfy customer needs. The use of fixtures, pallets, tool transporter and CTM practically eliminated the job setting time [4]. Broadly FMS is classified into four different categories; Single Flexible machines (SFM), Flexible Manufacturing Cells (FMCs), Multi machine FMS (MMFMS) and multi cell FMS (MCFMS) [2]. FMS aims at combining the advantages of higher efficiency in high volume mass production and higher flexibility in low volume job shop production.

In FMS, in order to achieve the higher efficiency and flexibility various scheduling decisions such as allocation of machines to jobs, allocation of AGVs and selection of tools are made. Proper scheduling plays a critical role in FMS in improving productivity.

2. LITERATURE REVIEW

In scheduling problems, for 'p' jobs and 'q' machines, $(p!)^q$ different number of sequences are to be examined with respect to any performance measure, to suggest a best sequence. This implies that the search region is increased exponentially for problem of larger size that makes the scheduling problem as NP-hard problem. In FMS various jobs are to be allocated to machines to

optimize the performance of FMS. This is similar to job shop scheduling. The main difference between them is that the job shop considers only jobs and machines, where as FMS considers resources such as AGVs, CTM, AS/RS, Robots, Pallets and Fixtures in addition to Jobs and machines. Hence scheduling problems connected with FMS are also NP-hard. Many researchers have addressed the machine and vehicle scheduling as independent problems. However the importance of simultaneous scheduling of jobs and automated guided vehicles (AGVs) has been emphasized by only few researchers. Raman et al [5] addressed the problem as an integer programming problem and procedure for solution based on the concepts of project scheduling under resource constraints. It was assumed that after transferring the load, the vehicle always returns to the load/ unload station, which reduces the AGV flexibility and influences the schedule length. Ulusoy and Bigle [6] attempted to make AGV scheduling an integral part of scheduling activity in an FMS. The problem was decomposed into two sub problems i.e. machine scheduling problem and vehicle scheduling problem. At each iteration , a new schedule for machines, generated by heuristic procedure was examined for its feasibility to the vehicle scheduling sub problem. The combined machine and AGVs scheduling problem was formulated as a non-linear mixed integer programming (MIP) model. BILGE and ULUSOY [7] proposed an iterative method based on the decomposition of the master problem into two sub-problems i.e., machine scheduling problem and vehicle scheduling problem. They developed a heuristic, named 'sliding time window (STW)', to solve the simultaneous off-line scheduling of machines and material handling in FMS. Ulusoy et al [8] proposed a genetic algorithm for this problem. Suitable coding scheme was provided, in which chromosome represents both the operation number and AGV assignment. Special genetic operators were developed for this purpose. The authors implemented their GA program with this coding and tested it on the 82 test problems that were solved earlier by the STW heuristic. Abdelmaguid et al [9] has presented a new hybrid genetic algorithm for the simultaneous scheduling problem for the makespan minimization objective. The hybrid GA is composed of GA and a heuristic. The GA is used to address the first part of the problem that is theoretically similar to the job shop scheduling problem and the vehicle assignment is handled by a heuristic called vehicle assignment algorithm (VAA). Murayama and Kawata [10] also addressed simultaneous scheduling of machines and AGVs. However it is assumed that AGVs can carry multiple loads instead of single load at a time. The genetic algorithm was applied to the problem. JERALD et al [11] proposed an adaptive GA (AGA) and ants colony optimization (ACO) for a 16-machine and 43-part problem. Their objective function is a combined objective of minimizing penalty cost and minimizing machine idle time. They also examined the speed of the AGV and found that AGA is superior to the ACO algorithm. Jerald et al [12] proposed the two approaches such as genetic algorithm and adaptive genetic algorithm used for scheduling both parts and AGVs simultaneously in an FMS environment. MURAYAMA and KAWATA [13] proposed a simulated annealing method for the simultaneous scheduling problems of machines and multiple-load AGVs to obtain relatively good solutions for a short time. The proposed method is based on a local search method for job shop scheduling problems. They provided a new representation of solutions and neighborhood operation in order to consider the transportation by multiple-load automated guided vehicles. Reddy and Rao [14] addressed the simultaneous scheduling problem as a multi objective problem in scheduling with conflicting objectives and solved by non-dominated sorting evolutionary algorithms. DEROUSSI et al [15] also addressed the problem of simultaneous scheduling of machines and vehicles in FMS. They proposed a new solution representation based on vehicles rather than machines, whereby each solution can thus be evaluated using a discrete event approach. An efficient neighbouring system is then implemented into three different meta-heuristics, namely iterated local search, simulated annealing and their hybridisation. Their results were compared with previous studies and show the effectiveness of the presented approach. Philippe Lacomma et al [16] attempted to model simultaneous scheduling of machines and identical automated guided vehicles using a frame work based on disjunctive graph and used memetic algorithm for scheduling machines and AGVs with the objective of minimum makespan.

3. FMS ENVIRONMENT

The FMS considered consists of 4 machines, a CTM consisting of tools, Automatic tool changer (ATC), Two identical AGVs and tool transporter (TT). On one end there is loading and unloading station and on other end there is a CTM. Buffer storage at each machine centre is provided to store the jobs before and after processing. There is an automated storage and retrieval system (AS/RS) for storage of

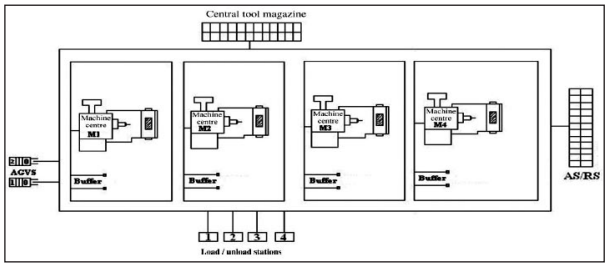


Fig 1. FMS Environment

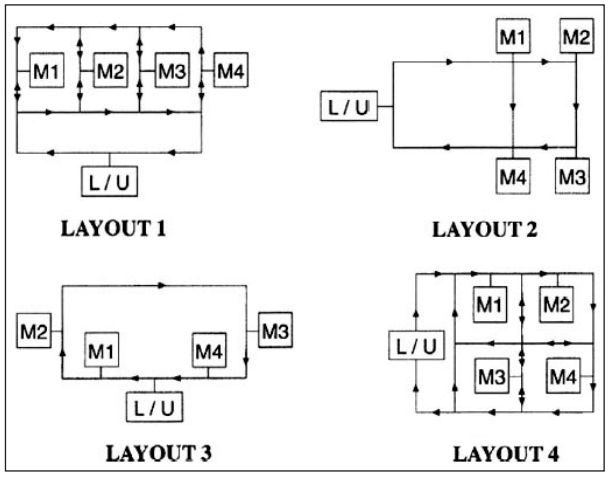


Fig 2. Layout Configurations used for Examples

raw materials and retrieval. The system is shown in figure 1 with the elements.

4. PROBLEM AND ASSUMPTIONS

Simultaneous scheduling of the machines and the material handling system in an FMS can be defined as follows: For the FMS described above determine the starting and completion times of operations for each job and the trips between workstations together with the vehicle assignment according to the objective of minimizing the make span.

It is assumed that all the design and set-up issues for the FMS as suggested by STECKE [19] have already been resolved. Four layout configurations as shown in figure 2 and ten job sets reported in Bilge et al [7] are used. The number of automated guided vehicles (AGVs) in the system is two. The types and number of machines are known. There is a sufficient input/output buffer space at each machine. Machine loading has been done i.e., allocation of tools to machines and the assignment of operations to machines. Operations are not pre- emptive. Each job is available at the beginning of the scheduling period. Ready times of all jobs are known. The routing of each job type is available before making scheduling decisions. Limitations on the jobs simultaneously allowed in the shop are ignored. The load/unload (L/U) station serves as a distribution centre for parts not yet processed and as a collection centre for parts finished. All vehicles start from the L/U station initially. There is a sufficient input/output buffer space at the L/U station. Trips follow the shortest path between two points and occur either between two machines or between a machine and the L/U station. Pre-emption of the trips is not allowed. The trips are called loaded or deadheading (empty) trips depending on whether or not a part is carried during that trip, respectively. The duration of deadheading trips is sequence-dependent and is not known until the vehicle route is specified. Processing, set-up, loading, unloading and travel times are available and deterministic. Vehicles move along predetermined shortest paths, with the assumption of no delay due to the congestion. As a result of this assumption, it would follow that the guide paths on segments can be uni-directional or bi-directional. However, on busy segments, two uni-directional paths should be used instead of a bi-directional guide path so that traffic congestion does not reach a critical level leading to the violation of this assumption. Furthermore, such issues as traffic control, machine failure or downtime, scraps, rework and vehicle dispatches for battery changes are ignored here and left as issues to be considered during real time control.

The following constraints are to be satisfied by the AGV travel when scheduling these FMSs:

A. For each operation j, there is a corresponding loaded trip whose destination is the machine where operation j is to be performed and its origin is either the machine where the operation preceding j is assigned or the L/U station.

B. Operation j of job I can start only after the trip to load has been completed.

C. An AGV trip cannot start before the maximum of the completion time of the previous operation of a job and the deadheading trip of the AGV to the job is obtained. The AGV travel times and the machine allocation and operation times for the jobs are given in Appendix A.

5. SIMULTANEOUS SCHEDULING METHODOLOGY

5.1 Algorithm / Procedural Steps In Simultaneous Scheduling Methodology

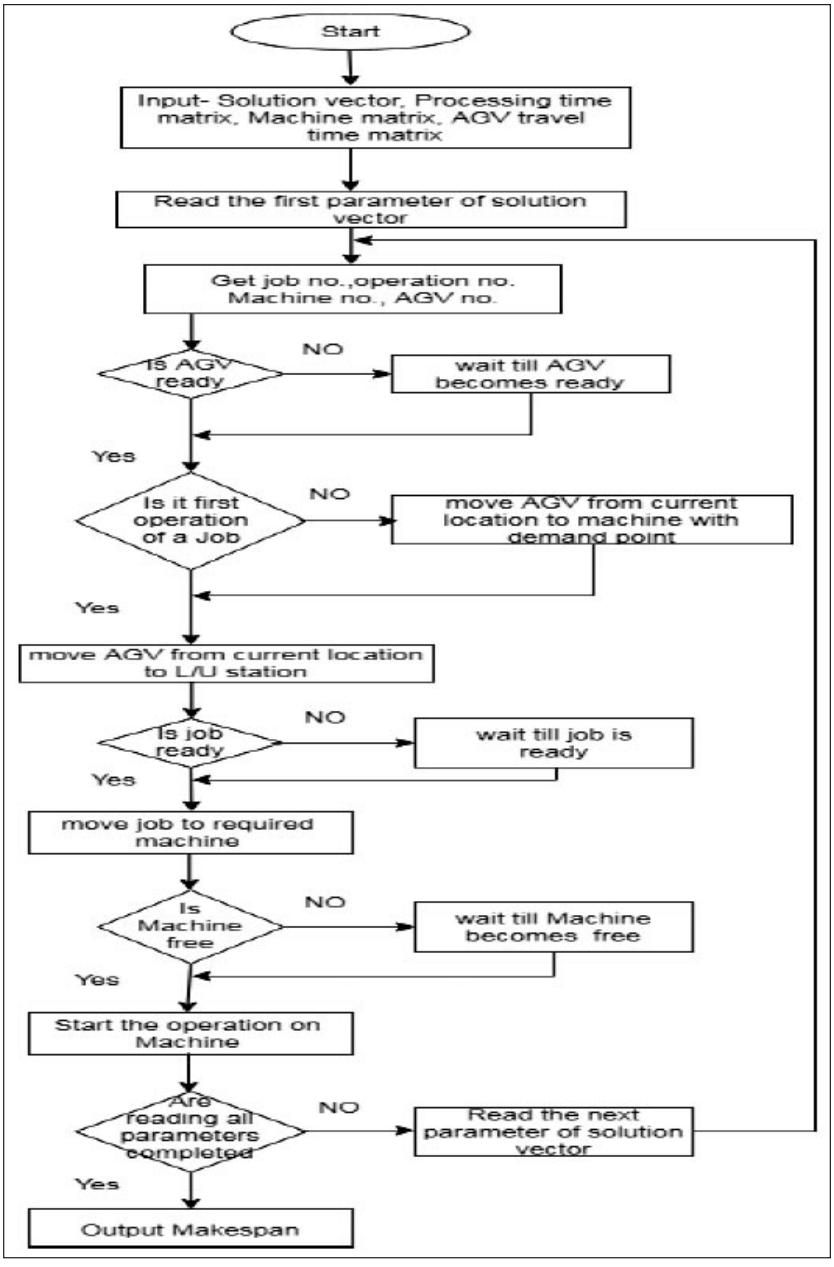


Fig 3. Simultaneous Scheduling Methodology Flow Chart

Step 1: Enter the input data: Job set details, solution vector, AGV travelling time matrix.

Step 2: Read parameter of solution vector one after another.

Step 3: Get job no, operation no, machine no, AGV no.

Step 4: Whenever AGV is ready move the AGV to the job, AGV waits till the job is ready then the AGV moves the job from its current location to the machine on which the job next operation is to be performed.

Step 5: Check whether machine is ready or not. If machine is ready load the job, else the job waits in the buffer till machine becomes ready.

Step 7: Start the operation on the machine.

Step 8: Check whether all parameters of solution vector are completed. If not, repeat from step2 onwards.

Step 9: If all the parameters are completed output the makespan.

5.2 Limits Function And Bounds Function

Limits function is used to make sure that the operations in the vector so generated using random numbers follows precedence requirement constraints of the operations. If the precedence

is not followed, the limits function correct the vector so that the operations of the vector follows precedence requirement constraints. Bounds function is used to make sure that the AGVs in the vector so generated using random numbers are within bounds. If the AGVs are not within the bounds which will be corrected by bounds function so that the AGVs of the vector are within the bounds.

The flow chat for simultaneous scheduling methodology is shown in Figure 3.

6. CROW SEARCH ALGORITHM [18]

In a crow flock, there is a behavior which has many similarities with an optimization process. According to this behavior, crows hide their excess food in certain positions (hiding places) of the environment and retrieve the stored food when it is needed. Crows are greedy birds since they follow each other to obtain better food sources. Finding food source hidden by a crow is not an easy work since if a crow finds another one is following it, the crow tries to fool that crow by going to another position of the environment. From optimization point of view, the crows are searchers, the environment is search space, each position of the environment is corresponding to a feasible solution, the quality of food source is objective (fitness) function and the best food source of the environment is the global solution of the problem. Based on these similarities, CSA attempts to simulate the intelligent behavior of the crows to find the solution of optimization problems.

Crows (crow family or corvids) are considered the most intelligent birds. They have demonstrated self-awareness in mirror tests and have tool-making ability. Crows can remember faces and warn each other when an unfriendly one approaches.

Moreover, they can use tools, communicate in sophisticated ways and recall their food's hiding place up to several months later. Crows have been known to watch other birds, observe where the other birds hide their food, and steal it once the owner leaves. If a crow has committed thievery, it will take extra precautions such as moving hiding places to avoid being a future victim. In fact, they use their own experience of having been a thief to predict the behavior of a pilferer, and can determine the safest course to protect their caches from being pilfered. Based on the above-mentioned intelligent behaviors, a population-based meta-heuristic algorithm, CSA, is developed.

The principles of CSA are listed as follows:

- Crows live in the form of flock.
- Crows memorize the position of their hiding places.
- Crows follow each other to do thievery.
- Crows protect their caches from being pilfered by a probability.

It is assumed that there is a *d*-dimensional environment including a number of crows. The number of crows (flock size) is *N* and the position of crow *i* at time (iteration) iter in the search space is specified by a vector $X^{i,iter} = [x_1^{i,iter}, x_2^{i,iter}, \ldots\ldots x_d^{i,iter}]$ and $iter_{max}$ is the maximum number of iterations. Each crow has a memory in which the position of its hiding place is memorized. At iteration iter, the position of hiding place of crow *i* is shown by $m^{i,iter}$. This is the best position that crow i has obtained so far. Indeed, in memory of each crow the position of its best experience has been memorized. Crows move in the environment and search for better food sources (hiding places). Assume that at iteration iter, crow j wants to visit its hiding place, $m^{j;iter}$. At this iteration, crow i decides to follow Crow *j* to approach to the hiding place of crow j. In this situation two states may happen:

State 1: Crow *j* does not know that crow *i* is following it. As a result, crow *i* will approach to the hiding place of crow *j*. In this case, the new position of crow i is obtained as follows $X^{i,iter+1} = X^{i,iter} + r_i \times fl^{i,iter} \times (m^{j,iter} - X^{i,iter})$.

where r_i is a random number with uniform distribution between 0 and 1 and $fl^{i;iter}$ denotes the flight length of crow i at iteration *iter*.

Figure 4 shows the schematic of this state and the effect of *fl* on the search capability. Small values of *fl* leads to local search (at the vicinity of $x^{i,iter}$) and large values results in global search (far from $x^{i,iter}$). As Fig. 4(a) shows, if the value of *fl* is selected less

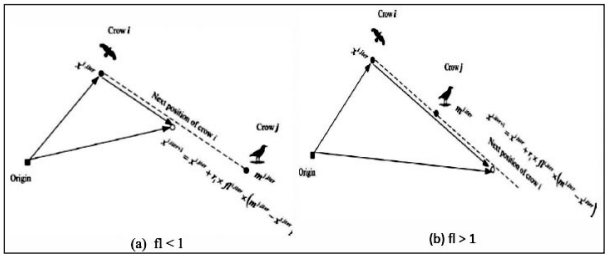


Fig 4. Flow Chart of State 1 in CSA.
Crow i Can go to Every Position on the Dash Line

Crow Search Algorithm

Randomly initialize the position of a flock of N crows in the search space
Evaluate the position of the crows
Initialize the memory of each crow
while *iter* < *itermax*
  for *i* = 1 : *N* (*all N crows of the flock*)
    Randomly choose one of the crows to follow (for example j)
Define an awareness probability
      if $r_j = AP^{j,iter}$,
$X^{i,iter+1} = X^{i,iter} + r_i \times fl^{i,iter}, \times (m^{i,iter}, - X^{i,iter})$
      else
        $X^{j,iter+1}$ = *a random position of search space*
      end if
  end for
Check the feasibility of new positions
Evaluate the new position of the crows
Update the memory of crows
  end while

Fig 5. Pseudo Code of the Proposed CSA

than 1, the next position of crow *i* is on the dash line between $x^{i,iter}$ and $m^{j,iter}$. As Fig. 4(b) indicates, if the value of *fl* is selected more than 1, the next position of crow *i* is on the dash line which may exceed $m^{j,iter}$.

State 2: Crow j knows that crow i is following it. As a result, in order to protect its cache from being pilfered, crow j will fool crow i by going to another position of the search space.

Totally, states 1 and 2 can be expressed as follows:

$$X^{I,ITER+1} = \begin{cases} X^{i,iter} + r_i \times fl^{i,iter} \times \left(m^{j,iter} - X^{i,iter}\right) r_i \geq AP^{i,iter} \\ Otherwise \\ random\ position \end{cases}$$

Where r_i is a random number with uniform distribution between 0 and 1 and $AP^{i,iter}$ denotes the awareness probability of crow *j* at iteration iter.

In CSA, intensification and diversification are mainly controlled by the parameter of awareness probability (AP). By decrease of the awareness probability value, CSA tends to conduct the search on a local region where a current good solution is found in this region. As a result, using small values of AP, increases intensification. On the other hand, by increase of the awareness probability value, the probability of searching the vicinity of current good solutions decreases and CSA tends to explore the search space on a global scale (randomization). As a result, use of large values of AP increases diversification.

Pseudo code of CSA is shown in Figure 5.

The step-wise procedure for the implementation of CSA is given in this section.

Step 1: Initialize problem and adjustable parameters
The optimization problem, decision variables and constraints are defined. Then, the adjustable parameters of CSA (flock size (*N*), maximum number of iterations ($iter_{max}$), flight length (*fl*) and awareness probability (*AP*)) are valued.

Step 2: Initialize position and memory of crows
N crows are randomly positioned in a d-dimensional search space as the members of the flock. Each crow denotes a feasible solution of the problem and d is the number of decision variables.

$$crows = \begin{bmatrix} X_1^1 & X_2^1 & .....X_d^1 \\ X_1^2 & X_2^2 & .....X_d^2 \\ . & . & ..... \\ . & . & ..... \\ X_1^N & X_2^N & .....X_d^N \end{bmatrix}$$

The memory of each crow is initialized. Since at the initial iteration, the crows have no experiences, it is assumed that they have hidden their foods at their initial positions

$$Memory = \begin{bmatrix} m_1^1 & m_2^1 & .....m_d^1 \\ m_1^2 & m_2^2 & .....m_d^2 \\ . & . & ..... \\ . & . & ..... \\ m_1^N & m_2^N & .....m_d^N \end{bmatrix}$$

Step 3: Evaluate fitness (objective) function
For each crow, the quality of its position is computed by inserting the decision variable values into the objective function.

Step 4: Generate new position
Crows generate new position in the search space as follows. Suppose crow i wants to generate a new position. For this aim, this crow randomly selects one of the flock crows (for example crow *j*) and follows it to discover the position of the foods hidden by this crow (m^j). The new position of crow *i* is obtained by Eq. (2). This process is repeated for all the crows.

Step 5: Check the feasibility of new positions
The feasibility of the new position of each crow is checked. If the new position of a crow is feasible, the crow updates its position. Otherwise, the crow stays in the current position and does not move to the generated new position.

Step 6: Evaluate fitness function of new positions
The fitness function value for the new position of each crow is computed.

Step 7: Update memory
The crows update their memory as follows:

$$m^{i,iter+1} = \begin{cases} X^{i,iter+1} & f\left(X^{i,iter+1}\right) is\ better\ than\ f\left(m^{i,iter}\right) \\ otherwise & \\ m^{i,iter} & \end{cases}$$

where f(·) denotes the objective function value.

It is seen that if the fitness function value of the new position of a crow is better than the fitness function value of the memorized position, the crow updates its memory by the new position.

Step 8: Check termination criterion
Steps 4–7 are repeated until itermax is reached. When the termination criterion is met, the best position of the memory in terms of the objective function value is reported as the solution of the optimization problem.

6. RESULTS AND DISCUSSIONS

The proposed algorithm is tested on 10 job sets with four different layouts (LY1, LY2, LY3 and LY4) for different traveling time/processing time (t/p) ratios reported in Bilge et al. [7]. Three cases are considered here for makespan calculation with increasing processing times with four different layouts (LY1, LY2, LY3 and LY4). In case I original processing times are used, in case II processing times are taken as double the original processing times and in case III processing times are taken as triple the original processing times. For LY1, LY2 and LY3 Case I and II are considered, for LY4 all the three cases are considered. For case II and case III AGV travelling times are halved. The three cases are grouped into two sets, one with relatively high t/p ratio >0.25(case I) and the other with relatively low t/p <0.25(case II and case III).

Results of proposed algorithm for case I of LY1, LY2, LY3 and LY4 in Table No. 1 , case II of LY1, LY2, LY3 and LY4, case III of LY4 in Table No.2 are presented. From these Tables it is observed that the proposed algorithm is superior in most of the cases or at least equal to the other methods. Table 1 consist of make span of problems whose ti/pi ratios greater than 0.25 while Table 2 consist of make span of problems whose ti/pi ratios are less than 0.25. A code is used to designate the problems which are given in the first column. The digits that follow EX indicates the job set in the layout. In table 2, another digits is appended to the code. Here having zero or one as the last digit implies that the process times are doubled or tripled respectively, where as travel times are halved in both cases.

The results of the proposed CSA are better over the STW on 25 problems, the UGA on 11 problems, the AGA on 7 problems, the RGA on 4 problems, the PDE1 on 5 problems and the PDE2 on 5 problems, same on 9 problems in STW, 20 problems in UGA, 20 problems in AGA, 21 problems in RGA, 20 problems in PDE1 and 17 problems in PDE2 and poor on 6 problems in STW, 9 problems in UGA, 13 problem in AGA , 15 problem in RGA, 15 problems in PDE1 and 18 problems in PDE2 for the case t/p ratio >0.25.

For t/p ratio <0.25 the proposed CSA performs better on 17 problems in STW, 4 problems in UGA, 4 problems in AGA, 4 problems in RGA, 4 problems in PDE1 and 4 problems in PDE2, same on 21 problems in STW, 35 problems in UGA, 34 problems in AGA,

Table 1: Comparison of CSA Results with other Methods for t_i/p_i>0.25

Job set	CSA	STW	UGA	AGA	RGA	PDE1	PDE2
EX 1.1	96	96	96	96	96	96	96
EX 1.2	82	82	82	82	82	82	82
EX 1.3	84	84	84	84	84	84	84
EX 1.4	103	108	103	103	103	103	103
EX 2.1	105	105	104	102	100	100	100
EX 2.2	76	80	76	76	76	76	76
EX 2.3	86	86	86	86	86	86	86
EX 2.4	108	116	113	108	108	108	106
EX 3.1	99	105	105	99	99	99	99
EX 3.2	85	88	85	85	85	85	85
EX 3.3	86	86	86	86	86	86	86
EX 3.4	111	116	113	111	111	111	110
EX 4.1	116	118	116	112	112	112	112
EX 4.2	88	93	88	88	87	85	86
EX 4.3	89	95	91	89	89	89	89
EX 4.4	89	95	91	89	89	89	89
EX 5.1	87	89	87	87	87	87	87
EX 5.2	69	69	69	69	69	69	69
EX 5.3	74	76	75	74	74	74	74
EX 5.4	96	99	97	96	96	96	95
EX 6.1	119	120	121	118	118	115	118
EX 6.2	98	98	98	98	98	98	98
EX 6.3	103	104	104	104	103	103	103
EX 6.4	124	120	123	120	120	120	120
EX 7.1	120	119	118	115	111	112	114
EX 7.2	85	90	85	79	79	79	79
EX 7.3	94	91	88	86	83	83	84
EX 7.4	137	136	128	127	126	126	126
EX 8.1	151	161	152	161	161	161	161
EX 8.2	141	151	142	151	151	153	151
EX 8.3	143	153	143	153	153	152	153
EX 8.4	154	163	163	163	163	163	163
EX 9.1	118	120	117	118	116	114	114
EX 9.2	102	104	102	104	102	104	104
EX 9.3	105	110	105	106	105	103	103
EX 9.4	123	125	123	122	122	123	123
EX 10.1	150	153	150	147	147	147	147
EX 10.2	145	139	137	136	135	135	135
EX 10.3	149	143	143	141	139	139	139
EX 10.4	165	171	164	159	158	158	158

Table 2: Comparison of CSA Results with Other Methods for $t_i/p_i<0.25$

Job set	CSA	STW	UGA	AGA	RGA	PDE1	PDE2
EX 1.10	126	126	126	126	126	126	126
EX 1.20	123	123	123	123	123	123	123
EX 1.30	122	122	122	122	122	122	122
EX 1.40	124	124	124	124	124	124	124
EX 2.10	148	148	148	148	148	148	148
EX 2.20	143	143	143	143	143	143	143
EX 2.30	146	146	146	146	146	146	146
EX 2.41	217	217	217	217	217	217	217
EX 3.10	150	148	150	150	150	150	150
EX 3.20	145	148	145	145	145	145	145
EX 3.30	146	149	146	146	146	146	146
EX 3.41	221	221	221	221	221	221	221
EX 4.10	119	121	119	119	119	119	119
EX 4.20	114	116	114	114	114	114	114
EX 4.30	114	116	114	114	114	114	114
EX 4.41	173	179	172	172	172	171	171
EX 5.10	102	102	102	102	102	102	102
EX 5.20	100	100	100	100	100	100	100
EX 5.30	99	99	99	99	99	99	99
EX 5.41	148	154	148	148	148	148	148
EX 6.10	186	186	186	186	186	186	186
EX 6.20	181	181	181	181	181	181	181
EX 6.30	182	184	182	182	182	182	182
EX 6.40	184	185	184	184	184	184	184
EX 7.10	137	137	137	137	137	137	137
EX 7.20	136	136	136	136	136	136	136
EX 7.30	137	137	137	137	137	137	137
EX 7.41	203	203	203	203	203	203	203
EX 8,10	272	292	271	292	292	292	292
EX 8.20	267	287	268	287	287	287	287
EX 8.30	268	288	270	288	288	288	288
EX 8.40	273	293	273	293	293	293	293
EX 9,10	176	176	176	176	176	176	176
EX 9.20	173	174	173	173	173	173	173
EX 9.30	174	176	174	174	174	174	174
EX 9.40	175	177	175	175	175	175	175
EX10.10	238	238	236	238	238	238	238
EX10.20	238	236	238	236	236	236	236
EX10.30	240	237	241	237	237	237	237
EX10.40	243	240	244	240	240	240	240

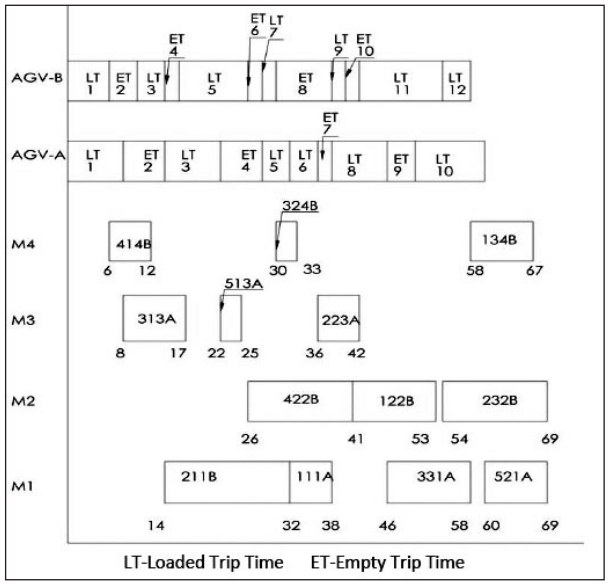


Fig 6. Gantt Chart for Jobset 5 Layout 2

34 problems in RGA, 34 problems in PDE1 and 34 problems in PDE2 and poor on 4 problem in STW, 3 problems in UGA, 4 problems in AGA, 4 problems in RGA , 4 problem in PDE1 and 4 problem in PDE2.

At the outset, out of the 82 problems the proposed CSA method performs better than STW on 42 problems, UGA on15 problems, AGA on 11 problems, RGA on 8 problems, PDE1 on 9 problems and PDE2 on 9 problems.

7. GANTT CHART

The Gantt chart for the sequence generated for job set 5, Layout 2 by CSA is shown in figure 6. The operations that are assigned to each machine as well as the start and finish times of each operation are shown in the Gantt chart. AGVs Loaded trip times(LT), Empty Trip times(ET) are also shown in Gantt chart. The Gantt chart shows the correctness of the solution provided by the proposed CSA method.

Each operation is denoted as three digit number followed by an alphabet. For example in the operation - 211B

2 - represents Job number,

1 – represents operation number,

1 – represents machine that is used for performing operation and

B – represents AGV that is used for moving job.

For AGV-A

LT1 for 313A , ET2 for 511A, LT3 for 513A, ET4 for 111A, LT5 for 111A,

LT6 for 223A, ET7 for 331A, LT8 for 331A, ET9 for 521A, LT10 for 521A

For AGV-B

LT1 for 414B, ET2 for 211B, LT3 for 211B, ET4 for 422B, LT5 for 422B,

ET6 for 324B, LT7 for 324B, ET8 for 122B, LT9 for 122B, ET10 for 232B

LT11 for 232B and LT12 for 134B

8. CONCLUSIONS

Scheduling of jobs and AGVs is carried out for minimizing the makespan objective by Crow search algorithm (CSA). The proposed algorithm is tested on 10 job sets with four different layouts and it is noticed that proposed algorithm outperforms the existing methods in minimizing makespan. The work can be extended by considering down time and AGVs dispatch time for battery change.

BIBLIOGRAPHY

1. Baker, K.R Introduction to Sequencing and Scheduling. New York, Wiley, 1974.
2. Lee,D; ANDF.DICESARE. Integrated Scheduling of FMSs Employing Automated Guided Vehicles, IEEE Transactions on Industrial Electronics, 41(6), 602–610, 1994.
3. AGNETIS, A.A.ALFIERI, P BRANDIMARTE, and P. Prinsecchi: Joint Job/Tool Scheduling in a Flexible Manufacturing Cell with No On-Board Tool Magazine, 'Computer Integrated Manufacturing System', 10 (1), 61–68 1997.
4. Jerald, J; Asokan, P: Simultaneous Scheduling of Parts and Automated Guided Vehicles in an FMS Environment using Adaptive Genetic Algorithm, 'International Journal of Advanced Manufacturing Technology', 29 (5), 584–589, 2006.
5. Raman, N; Talbot, FB, Rachamadgu, RV: Simultaneous scheduling of machines and material handling devices in automated manufacturing [C]// STECKE K E, SURI R. Proceedings of the Second ORSA/TIMS Conference on Flexible Manufacturing Systems. University of Michigan, Ann Arbor, MI,USA, 1986: 455−466.
6. Ulusoy, G; Bilge, U: Simultaneous scheduling of machines and automated guided vehicles [J]. International Journal of Production Research, 1993, 31(12): 2857−2873.
7. Bilge U, Ulusoy G: A time window approach to

simultaneous scheduling of machines and material handling system in FMS [J]. Operations Research, 1995, 43: 1058−1070.

8. Ulusoy, G; Sivrikaya-Serifoglu F, Bilge U: A genetic algorithm approach to the simultaneous scheduling of machines and automated guided vehicles [J], 'Computers & Industrial Engineering', 1997, 24 (4): 335−351.
9. Abdelmaguid, TF; Nassef, ON; Kamal, BA; Hassan, MF: A hybrid GA/heuristic approach to the simultaneous scheduling of machines and automated guided vehicles [J], 'International Journal of Production Research', 2004, 42: 267−281.
10. Murayama, N; Kawata, S: A genetic algorithm approach to simultaneous scheduling of processing machines and multiple-load automated guided vehicles [J], 'Transactions of the Japan Society of Mechanical Engineers C', 2005, 71 (712): 3638−3643. (in Japanese)
11. Jerald, J; Asokan, P; Saravanan, R; Rani A D C: Simultaneous scheduling of parts and AGVs in an FMS using non-traditional optimization algorithms [J], 'International Journal of Applied Management and Technology', 2005, 3(1): 305−315.
12. Jerald, J; Asokan P; Saravanan R; Rani A D C: Simultaneous scheduling of parts and automated guided vehicles in an FMS environment using adaptive genetic algorithms [J], 'International Journal of Advanced Manufacturing Technology', 2006, 59: 584−589
13. Murayama, N; Kawata, S: Simulated annealing method for simultaneous scheduling of machines and multiple-load AGVs [C]// IJCC Workshop on Digital Engineering, Pyeongchang-gun, Gangwon-do, South Korea, 2006: 55−62. (in Japanese)
14. Reddy, B S P; Rao, C S P: A hybrid multi-objective GA for simultaneous scheduling of machines and AGVs in FMS [J], 'International Journal of Advanced Manufacturing Technology', 2006, 31 (5/6): 602−613.
15. Deroussi, L; Gourgand, M; Tchernev, N: A simple met heuristic approach to the simultaneous scheduling of machines and automated guided vehicles [J]. International Journal of Production Research, 2008, 46(8): 2143−2164.
16. Philippe Lacomme, Mohand Larabi, Nikolay Tchernev: Job-shop based framework for simultaneous scheduling of machines and automated guided vehicles, 'Int. J. Production Economics', 143, 2013, 24–34
17. Stecke, KE: Design, planning, scheduling, and control problems of flexible manufacturing systems [J], 'Annals of Operations Research', 1985, 3(1):3−12.
18. Alireza Askarzadeh, A novel metaheuristic method for solving constrained engineering optimization problems: Crow search algorithm,Computers and Structures 169, 2016, 1– 12 ◘

APENDIX A

A. Travel time matrix for the example problem

	L/U	M1	M2	M3	M4
L/U	0	4	6	8	6
M1	6	0	2	4	2
M2	8	12	0	2	4
M3	6	10	1	0	2
M4	4	8	10	12	0

B. Data for the job sets used:

Job Set 1		**Job Set 2**		**Job Set 3**	
Job1	M1- (8);	Job1	M1-(10);	Job1	M1-(16);
	M2- (16);		M4-(18)		M3-(15)
	M4-(12)				
Job2	M1- (20);	Job2	M2-(10);	Job2	M2- (18);
	M3-(10);		M4- (18)		M4- (15)
	M2- (18)				
Job3	M3-(12);	Job3	M1- (10);	Job3	M1- (10);
	M4-(8);		M3- (20)		M2- (10)
	M1- (15)				
Job4	M4- (14);	Job4	M2- (10); M3-(15);	Job4	M3- (15);
	M2-(18)		M4-(12)		M4- (10)
Job5	M3-(10);	Job5	M1- (10); M2-	Job5	M1-(8); M2- (10); M3-(15);
	M1-(15)		(15);		M4-
			M4- (12)		(17)
		Job6	M1-(10);	Job6	M2-(10); M3- (15); M4-
Job Set 4			M2-(15);		(8); M1-
			M3- (12)		(15)
Job1	M4- (11);				
	M1-(10);	**Job Set 5**		**Job Set 6**	
	M2- (7)				
Job2	M3- (12);	Job1	M1-(6);	Job1	M1- (9);
	M2-(10);		M2-(12);		M2-(11);
	M4-(8)		M4- (9)		M4-(7)
Job3	M2- (7);	Job2	M1- (18);	Job2	M1-(19);
	M3-(10);		M3-(6);		M2-(20);
	M1- (9);		M2-(15)		M4-(13)
	M3- (8)				
Job4	M2-(7);	Job3	M3- (9);	Job3	M2- (14);
	M4- (8);		M4-(3);		M3-(20);
	M1- (12);		M1- (12)		M4- (9)
	M2- (6)				

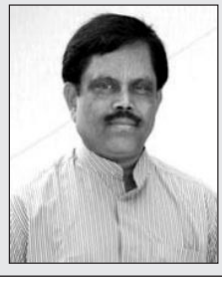
Job5	M1- (9);	Job4	M4-(6);	Job4	M2-(14);
	M2-(7);		M2-(15)		M3- (20);
	M4- (8);				M4-(9)
	M2- (10);				
	M3- (8)				
		Job5	M3-(3);	Job5	M1-(11);
Job Set 7			M1- (9)		M3- (16);
					M4-(8)

Job1	M1-(6);			Job6	M1-(10);
	M4- (6)	**Job Set 8**			M3-(12);
					M4-(10)
Job2	M2-(11);	Job1	M2-(12);		
	M4-(9)		M3- (21);	**Job Set 9**	
			M4- (11)		
Job3	M2-(9);	Job2	M2-(12);	Job1	M3- (9);
	M4-(7)		M3-(21);		M1-(12);
			M4-(11)		M 2 - (9) ; M4-(6)
Job4	M3- (16);	Job3	M2-(12);	Job2	M3-(16);
	M4-(7)		M3-(21);		M2- (11);
			M4-(11)		M4-(9)
Job5	M1-(9);	Job4	M2-(12);	Job3	M1-(21);

	M3-(18)		M3-(21);		M2-(18);
			M4-(11)		M4-(7)
Job6	M2-(13);	Job5	M1-(10);	Job4	M2- (20);
	M3-(19);		M2-(14);		M3- (10);
	M4-(6)		M3-(18);		M4-(11)
			M4-(9)		
Job7	M1-(10);	Job6	M1-(10);	Job5	M3-(14);
	M2-(9);		M2-(14);		M1- (16);
	M3-(13)		M3-(18);		M2-(13);
			M4-(9)		M4-(9)
Job8	M1-(11);				
	M2-(9);				
	M4-(8)				
Job Set 10					
Job1	M1-(11);	Job3	M3-(8);	Job5	M1-(9);
	M3-(19);		M2-(10);		M3-(16);
	M2-(16);		M1-(14);		M4-(18)
	M4-(13)		M4-(9)		
Job2	M2-(21);	Job4	M 2 - (1 3) ; M3-(20);	Job6	M2-(19);
	M3-(16);		M4-(10)		M1-(21);
	M4-(14)				M3-(11);
					M4-(15)

N Sivarami Reddy Graduated in Mechanical Engineering from Andhra University in 1989. He completed M.Tech (Advanced Manufacturing Engineering) at NITK, Surathkal in 1992. He is pursuing research in JNTUA, Ananthapuramu. He has 24 years of experience in teaching and administration in engineering colleges. Currently he is a Professor, Mechanical Engineering Department, AITS, Rajampet. He has 10 papers to his credit. His research areas include Scheduling of FMS and Softcomputing Techniques. (E-mail: siva.narapureddy@gmail.com)

Dr. D V Ramamurthy is working currently as Principal, Godavari Institute of Engineering and Technology, Rajahmundry. He graduated from College of Engineering, Andhra University in 1987 and obtained his Post Graduation from IIT, Delhi in 1992. He obtained his Ph.D from Oklahama State University, USA. He has 28 years of teaching experience. He has 20 Publications to his credit. His research areas include Optimization, Design manufacturing and Control systems.



Dr. K Prahlada Rao is presently working as Professor, Department of Mechanical Engineering, JNTUACEA, Ananthapuramu. He has 31 years teaching experience. He Published 80 papers in reputed International Journals and presented 50 papers in International Conferences. His areas of interests include Artificial Intelligence, Expert systems applications in design, Production and industrial Engineering Domains.