We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

4,800
Open access books available

122,000
International authors and editors

135M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

CLARIVATE ANALYTICS BOOK CITATION INDEX INDEXED

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

8

Parallel Robot Scheduling with Genetic Algorithms

Tarık Cakar[1], Harun Resit Yazgan[1] and Rasit Koker[2]
[1]*Sakarya University Industrial Engineering Department*
[2]*Sakarya University Computer Engineering Department*
Sakarya Turkey

1. Introduction

There are some main goals in parallel robot scheduling. Those are total completion time, maximum earliness, and maximum tardiness. According to the theoretical viewpoint, parallel robot scheduling is a generalization of the single robot scheduling and a special study of the flow shop. From the practical viewpoint, solution techniques are useful in the real-world problems. Parallel robot scheduling has to deal with balancing the load in practice. Scheduling parallel robot may be considered as a double-step. First, which jobs are allocated to which Robot. Second, allocated jobs sequence. Also, preemption plays a more important role in parallel robot scheduling. Robots may be identical or not. Jobs have a precedence constraint. For all problem structures may be applied different solution techniques for instance algorithms, search algorithms or artificial intelligence techniques. In this chapter we interest in different solution techniques for parallel robot scheduling.

In this chapter, first, a genetic algorithm is used to schedule jobs that have precedence constraints minimizing the total earliness and tardiness cost and maximum flow time on n-number of job and m-number of identical parallel robots. The second one is without precedence constraint. There are many algorithms and heuristics related to the scheduling problem of parallel machines and robots. In this study, a genetic algorithm has been used to find the job schedule, which minimizes maximum flow time. We know that this problem is in the class of NP-hard combinatorial problem.

(Kanjo & Ase, 2003) studied about scheduling in a multi robot welding system. (Sun & Zhu, 2002) applied a genetic algorithm for scheduling dual resources with robots. (Zacharia & Asparagatos, 2005) proposed a method on GAs for optimal robot task scheduling. In this study, the job with n-number of precedence constraints is assigned minimizing mean tardiness on m-number of parallel robot using genetic algorithms.

(Koulamas,1997) developed a heuristic noted hybrid simulated annealing (HAS) based on simulated annealing. (Chen et al.,1997) has developed highes priority job first (HPJF) method, which is based on extension of the WI method extended with various priority rules such as minimum processing time first (priority = 1/processing time), maximum processing time first (priority=processing time), minimum deadline first (priority=1/due date) and maximum deadline first (priority = Due date). (Alidaee & Rosa, 1997) proposed a heuristic which is based on extending the modified due date (MDD) method belonging (Baker &

Source: Parallel Manipulators, New Developments, Book edited by: Jee-Hwan Ryu, ISBN 978-3-902613-20-2, pp. 498, April 2008, I-Tech Education and Publishing, Vienna, Austria

Bertrand, 1982). Their method is quite effective for parallel machine problem according to their reports. (Azizoglu & Kirca, 1998) proposed a branch and bound (BAB) approach to solve the same problem mentioned in this paper. Another example can be given by considering identical due dates and processing times, (Elmaghraby & Park, 1974), developed an algorithm based on a branch and bound to minimize a function of penalties belonging to tardiness. (Barnes & Brennan,1977) evaluated and improved their method again.

In addition to these previous studies, there are a few more studies, which deal with parallel machine scheduling problem. But these studies are interested in alternatives. A few examples are given in the following for the minimization of the total weighted tardiness: (Emmons & Pinedo, 1990), (Arkin & Roundy, 1991); for uniform or unspecified parallel machines scheduling, the example studies are: (Emmons, 1987) or (Guinet, 1995). (Karp, 1972) has shown that even the total tardiness minimization in two identical machine scheduling problem was NP-hard. A branch and bound algorithm to minimize maximum lateness considering due dates, family setup times and release dates have been presented by (Shutten & Leussink, 1996). A genetic algorithm was used to find a scheduling policy for identical parallel machine with setup times in (Tamimi & Rajan, 1997). (Armento Yamashita , 2000) applied tabu search into parallel machine scheduling. A scheduling problem for unrelated parallel machine with sequence dependent setup times was studied by (Kim et al. , 2002) using simulated annealing. SA was used to determine a scheduling policy to minimize total tardiness. (Min & Cheng, 1995) proposed an algorithm for identical parallel machine problem. Their algorithm is based on using GA and SA to minimize makespan. According to their studies, it is seen that GA proposed is efficient and fit for larger scale identical machine scheduling problem to minimize the makespan.

(Kashara and Narita, 1985) developed a heuristic algorithm and optimization algorithm for parallel processing of robot arm control computation on a multiprocessor system. (Chen et al., 1988) developed a state-space search algorithm coupled with a heuristic for robot inverse dynamics computation on a multiprocessor system. An assignment rule noted traffic priority index (TPI) was built in 1991 by (Ho & Chang, 1991). In this method, SPT and EDD rules are combined using by using a new measurement named as traffic congestion ratio (TCR). Then, for the cases with one or identical machine they built heuristics. Their heuristics consist of building a first solution by scheduling jobs in increasing order of their priority index. Then they improved this solution using permutation technique of WI method, which was developed previously by (Wilkerson & Irwin, 1971).

2. Definition of the problems

In this study, the job with n-number of precedence constraints is scheduled minimizing total earliness and tardiness cost and maximum flow time on m-number of parallel robots. There are process time and due date for each job. There is not any ready time that belongs to jobs. A robot can do just one job at the same time. The processing is non-preemptive. The target function, which will be minimized, is given below in Eq. (1).

$$Total\_earlines\_tardines\_\cos t = w_e \sum_{j=1}^{n} e_i + w_T \sum_{i=1}^{n} T_i \qquad (1)$$

Here, $T_j = \max\{0, C_j - d_j\}$ is the tardiness of job j. $e_j = \max\{0, d_j - C_j\}$ is the earliness of job j. C_j being the completion time and d_j being due date for job j. R(i,j), represents processing or unprocessing of j job on i robot. w_e is unit earliness cost, w_T is unit tardiness cost. If j job is being processed on i robot, R(i,j)=1, otherwise (if not being processed) R(i,j)=0. Fmax is maxsimum flow time. Pj is processing time.

$$\text{Fmax} = \max\ (F_i = \sum_{i=1}^{m}\sum_{j=1}^{n} R(i,j)\,p_j\ ) \tag{2}$$

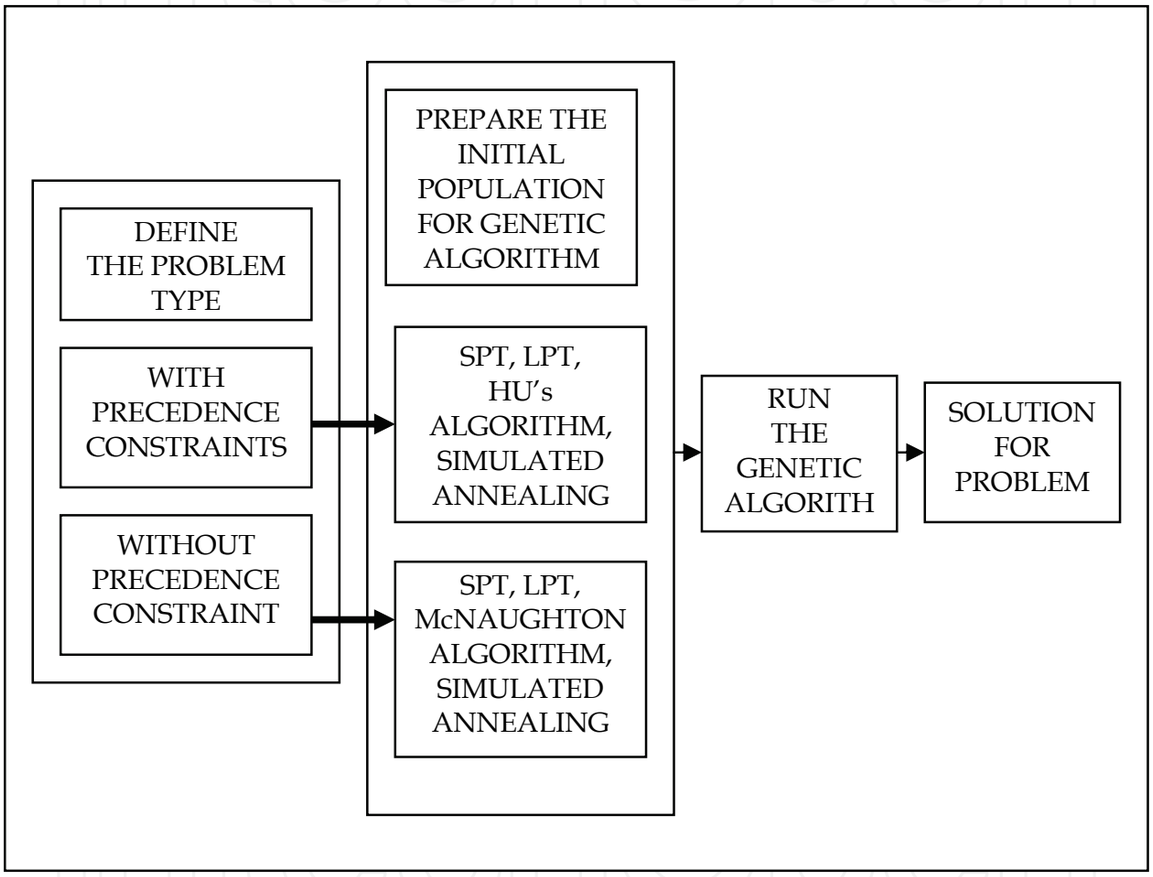

Figure 1. Proposed solution system for the parallel machine scheduling problem.

3. Genetic algorithm

The advantages of the genetic algorithms have been mentioned in the previous section. In this section, the modeling and the application of the GA are explained. From the view point of the working principle, genetic algorithms firstly needs the coding of the problem with the condition that it should be fitting with the GA. After coding process, GA operators are applied on chromosomes. It is not guaranteed that the obtained new offsprings are good solutions by the working of crossover and mutation operators. Feasible solutions are evaluated, and others are left out of evaluation. The feasible ones of the obtained offsprings

are taken and new populations are formed by reproduction process using these offsprings. Crossover, mutation and reproduction processes go on until an optimal solution is found. The modeling of the defined problem using genetic algorithm has been presented below with its details.

3.1 Coding for problem statement

The scheduling of the jobs on each robot forms the chromosomes. Here, the chromosomes give the number of robots too. The gene code are c_1, c_2, c_3,.., c_j,… , c_n, where $c_j \in [1,m]$. c_j is positive integer number. Here, each parallel robot represents a chromosome; and gene in chromosome, represents ordered jobs on a robot. The assigned of jobs on robots when forming initial population is done randomly, and while this ordering is done, precedence constraints are taken under care. For instance, let us suppose that there are 8 jobs and 2 robots, and their precedence constraints are given in Figure 3. Sample list representation of the schedule of the jobs on M1 and M2 robots has been given in figure 3. The sample schedule gives also a sample gene code.

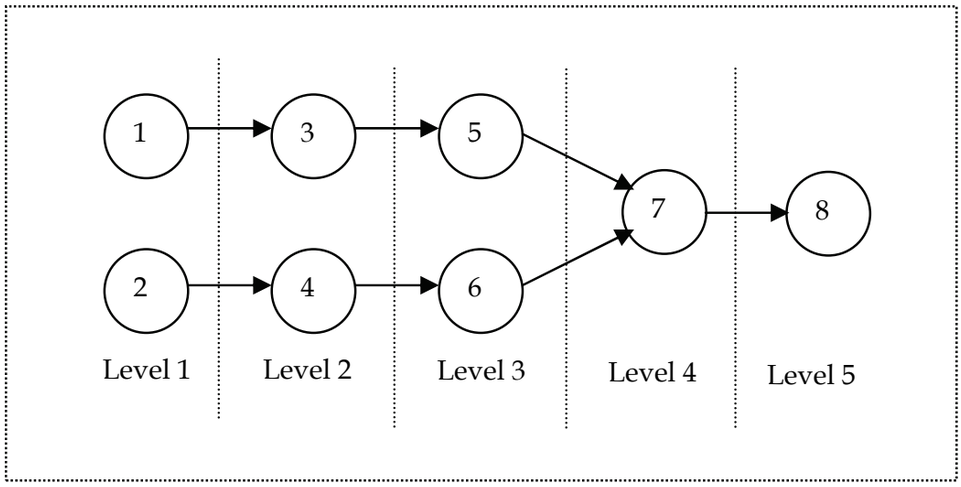

Figure 2. The jobs with precedence constraints

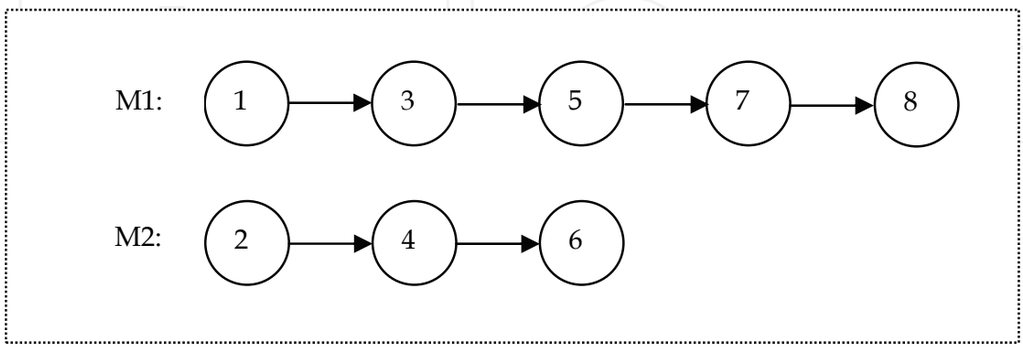

Figure 3. List representation of the schedule

Here, the scheduling of the jobs on robots also shows chromosomes code. M job can be scheduled on N robots in different combinations. But, because of the fact that some of the obtained schedules will be precedence constraints in problem definition, they will not be

possible solution. For example, the solution given in figure 4 is not a feasible solution for the precedence constraints in Figure 4. Because the precedence constraints have not been taken under care.

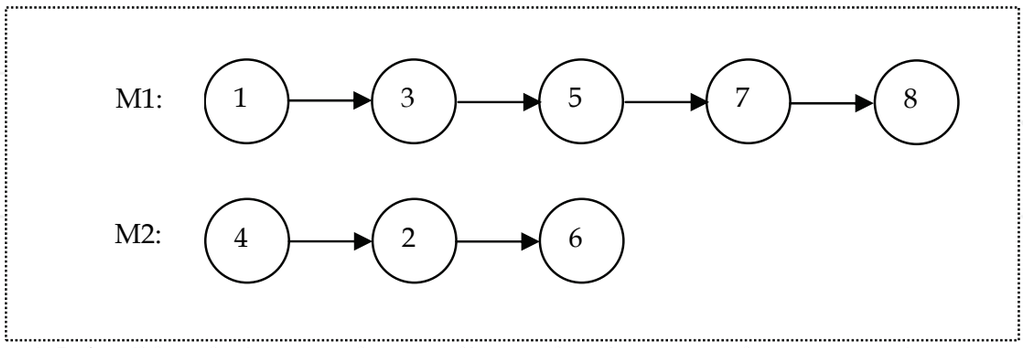

Figure 4. Infeasible solution sample according to the given precedence constraints

3.2 Preparing initial population

Initial population is not produced randomly, fully. In initial population, the solutions for problem with precedence constraints, which are obtained from SPT and EDD heuristics, Simulated Annealing, Hu's algorithm (Baker, 1974) exist and the other. In initial population, the solutions for problem without precedence constraints, which are obtained from SPT and EDD heuristics, Simulated Annealing, McNaughton's algorithm (Baker, 1974) exists. The chromosomes out of these are generated randomly. The jobs are randomly let (determined or given) on robots. However, because of the precedence constraints, in other words, there are some situations like that some jobs may be done before others; some of obtained solutions will not be feasible. These solutions, which are not feasible, will be thrown and the new solutions will be tried to be obtained, randomly.

3.3 Applying crossover operator for the problem

The crossover process is crossing obliquely from cut points of randomly determined two chromosomes. At the end of this operation, two new chromosomes are obtained. In this problem when chromosomes are crossed with, cross is taken care to the chromosomes in the same robots. For instance, number 1 robot in the first chromosomes and number 1 robot in the second chromosomes are crossed. Then, the second robot in the first chromosomes and the second robot in the second chromosomes are crossed. Let us explain this with an example;

By taking care of the given precedence constraints given in Figure 2, let us crossover the given two chromosomes in figure5.

	M1 1 3 5 7 8		M1 2 4 6 7 8
CHROMOSOME #1	,	CHROMOSOME #2	
	M2 2 4 6		M2 1 3 5

Figure 5. Two different chromosomes for crossover process

As it is seen above, the jobs in the first chromosomes on the first robot have been scheduled as 1-3-5-7-8 and in the second robot they have been scheduled as 2-4-6. The schedule in the

second chromosomes on the first robot is as 2-4-6-7-8, and on the second robot as 1-3-5. When crossover process is applied to these chromosomes, the first robot in the first chromosome and the first robot in the second chromosome and the second robot in the first chromosome and the second robot in the second chromosome gene will be crossed from randomly determined points. The result of the crossover operation has been given in figure 6.

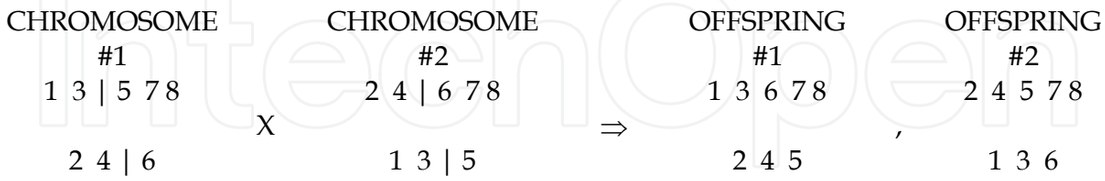

Figure 6. Crossover process and obtained offsprings

Here, the sign "|" refers to randomly selected crossover point. On the other hand, the sign "X" represents the crossover operation. At the end of crossover operation, two new chromosomes are obtained. The selected crossover point is the same on the parts representing M1 and M2 parallel robots of chromosomes in the example given in figure 5, and it is after than second gene. But, for instance, the point after than second gene for M1 part may be crossover point, likewise the point after than the first gene may be crossover point for M2 part. Here, there is the possibility of obtaining unfeasible solutions when there are precedence constraints between jobs.

3.4 Applying mutation operator for the problem

In the mutation operation, a gene is randomly selected from inside of the chromosomes in the population according to the given mutation rate. This gene will represent a job. This job will be swapped with any other job, which has the same precedence constraint on another robot or on the same robot with it. If there is more than one job, which is on the same level with it, one of them will be selected randomly. At the end of the mutation operation, a new chromosome will be obtained. For example, let us apply mutation operation to the chromosome given in figure 7;

SELECTED CHROMOSOME AND GEN	MUTATION	OFFSPRING
1 3 **5** 7 8	The job, which is on the same level with number 5 job, will be replaced with number 6 job so two jobs will be swapped.	1 3 6 7 8
2 4 6		2 4 **5**

Figure 7. Mutation process and obtained offspring

3.5 Reproduction

A copy of each gene is made by the reproduction operator in the population and it is added to the list of candidate genes. Fundamentally, this warrants that each chromosome in the current population remains a candidate to be selected for the next population. In this problem, the aim is to find the solution that minimizes the given fitness function. As it is known the fitness function is a tardiness value function. Here, the obtained chromosomes

are scheduled from low tardiness value to high tardiness value in every population. GA may have better chances to survive chromosomes with quite higher fitness. The living good chromosomes stay in the population. This process will be kept going until an optimal solution is found in each population.

4. Simulated annealing

In this study, two operators have been used in the application of SA. The first operator is that a randomly selected job has been swapped with another job, which is on the same level, and then, a new offspring has been obtained. The second operator is that a randomly selected job has been again swapped with another job and then, a new solution alternative has been obtained. If these obtained solution alternatives are valid, they are taken into consideration. Used first operator does the same operation with the mutation operation in GA. The working mechanism of these used operators has been revealed in figure 8 and 9.
SA begins with an initial solution (A), and initial temperature (B), and an iteration number (C). The duty of temperature (T) is controlling the possibility of the acceptance of a disturbing solution, and an iteration number (C) is used in the decision of the number of repetitions until a solution has a stable state under the temperature. The T may have the following implicit meaning of flexibility index. At high temperature situation, namely, early in the search, there is some flexibility to move to a worse solution situations, on the other hand, at lower temperature, in other words later in the search, less of this flexibility exists. A new neighborhood solution (N) is generated based on these B, C through a heuristic perturbation on the existing solutions. If the change of an objective function is improved, the neighborhood solution (N) becomes a good solution. Even though it is not improved, the neighborhood solution will be a new solution with a convenient probability which is based on $e^{-\Delta/T}$. This situation leaves the possibility of finding a global optimal solution out of a local optimum. The algorithm will be stopped when there is no change after C iterations. Otherwise, the algorithm will be continuing with a new temperature value (T).

4.1. Simulated annealing algorithm

```
Begin;
        INITIALIZE (A,B,C);
Repeat
        For I=1 to C do
            N= PERTURB (A);  {generate new neighborhood solution}
            D= C(N)-C(A)
            If((C(N)<=C(A) or (exp(-D/T)>RANDOM(0,1))
                Then A=N; {Accept the movement)
            Endif
      Endfor;
        UPDATE (T, C);
Until (Stop-Criterion)
End
```

In order to apply SA to practical problems, there are several factors to be decided initially. Firstly, the definition of a procedure to generate neighborhood solutions from a current solution is necessary. To generate these solutions efficiently, some parameters should be decided appropriately. Some examples to these parameters can be given as an initial

temperature, the number of repetitions, conditions for completion and the ratio of temperature change. The combination of these parameters should be adjusted according to the problem to obtain a good solution.

SA has some weak points such as long running time and difficulty in selecting cooling parameter when the problem size becomes larger. A geometric ratio was used in SA as $T_{k+1} = \alpha T_k$, where T_k and T_{k+1} are the temperature values for k and k+1 steps, respectively. Geometric ratio is used more commonly in practice. In this study, the initial temperature was taken 10000 and 0.95 was used for cooling ratio (α).

OLD SOLUTION	SA OPERATOR-1	NEW SOLUTION
1 3 **5** 7 8	Only number 6 work is on the same level with number 5 work that is selected randomly; so two works will be exchanged.	1 3 6 7 8
2 4 6		2 4 **5**

Figure 8. The first new solution generation operator used in SA

OLD SOLUTION	SA OPERATOR-2	NEW SOLUTION
1 3 5 7 8	Randomly selected number 1 work will be swapped with again randomly selected number 4 work	4 3 6 7 8
2 4 6		2 **1** 5

Figure 9. The second new solution generation operator used in SA

5. Comparison of GA and SA

GA and SA are not much different algorithms; theoretically, both of them are quite relative algorithms. However, their formulations are done using very different terminology. In a problem solution with SA, the costs, neighbors and moves of the solutions are talked (discussed), however, in a problem solution with GA, one discusses about chromosomes, their crossover, fitness and mutation. Another difference; a chromosome is considered as a genotype, which only indicates a solution. This is a traditional feature of GA and there is not any reason about that why a resembling approach could not be used in SA in the same way.

Fundamentally, for the situation of that the population size is only one, SA can be considered as GA. Because there is only chromosome, and there is not any crossover, but only mutation. Indeed, this the most important difference between GA and SA. SA generates a new solution by modifying only one solution with a local move; however, GA generates solutions by using the different solutions in a combination. It is not exactly known that if this actually makes the algorithm better or worse, however, it is clear that it depends on the problem and the representation. The principles of these two algorithms are based on the same basic supposition that convenient solutions are mode probably found "near" already known convenient solutions than by randomly selecting from the whole solution space. If this were not the case with a particular problem or representation, they

would not perform better than random sampling. The difference in the action of the GA is treating combinations of two existing solutions as being "near", supposing that such combinations (children) significantly share the properties of their parents, so that a child of two suitable solutions is more probably good solution than a random one. It should us significantly emphasized that this is just valid for a particular problem or representation; otherwise GA will not have an advantage over SA.

6. Example problem-I

Seven jobs and two parallel machines problem is given as an example below. The process and due dates belongs the works in table 1 and additionally, the precedence constraints in figure 9 were given. The solution, which minimizes maximum flow time, was obtained by considering these data. The problem was solved by using three different methods, which are SPT heuristic, SA and GA. The data and the results were given below.

Job i	Processing time	Due date
1	3	9
2	2	8
3	4	3
4	6	7
5	7	4
6	5	5
7	8	6

Table 1. Processing time and Due date of every job

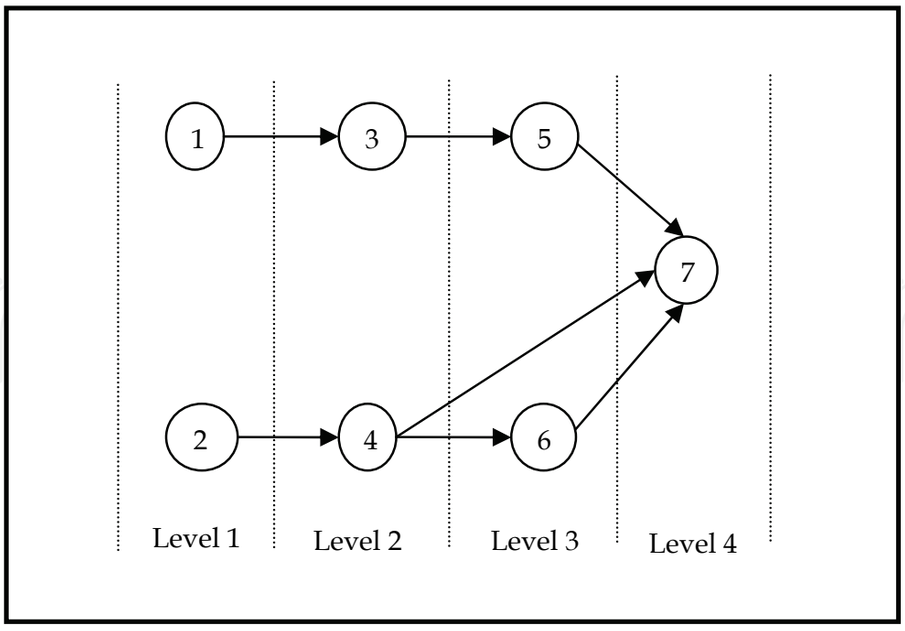

Figure 10. Precedence constraints of every job for example problem

The result of the implementation of GA and SA to the problem stated above has been given in Table 2. Furthermore, in figure 10, the view of the obtained solution from GA on Gannt

Chart has been given. The complementing (finishing) time of each job has been shown on Gannt chart. For example, the finishing time of the number 5 job and number 7 job are 14 and 22, respectively. 7x2 refers to 7 jobs and 2 machines.

Heuristic	Schedule	Maximum Flow Time
SPT	M1: 2-4-6 M2: 1-3-5-7	22
EDD	M1: 2-1-3-5-7 M2: 4-6	24
GA	M1: 1-3-5 M2: 2-4-6-7	22
SA	M1: 1-3-5 M2: 2-4-6-7	22

Table 2. The result of calculation for 7 X 2 problem size

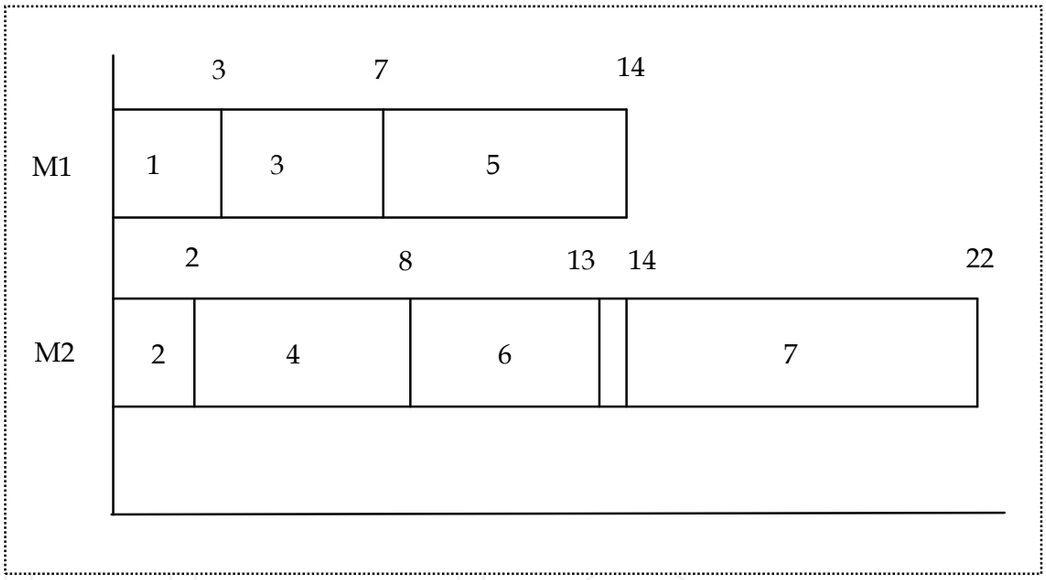

Figure 11. A schedule for two machines displayed as Gannt Chart

7. Example problem-II

As another example, a parallel machine problem with 12 jobs and 2 parallel machines was taken under consideration below. The process times, delivery times and precedence constraints of jobs were given. The solution, which minimizes the total earliness and tardiness cost, was obtained by considering these data. The problem was solved by using SPT, EDD, SA and GA. The data and the results were given below. In Table 3, the jobs with process and due dates belonging to them were given. The precedence constraints of the jobs were given in Figure 11. In Table 4, the solutions obtained from GA, SA, SPT and EDD were given. Tardiness cost and earliness cost have been taken as 1 and 0.5, respectively.

Job i	Processing time	Due date

1	2	1
2	4	3
3	5	2
4	3	8
5	8	7
6	7	4
7	10	12
8	12	14
9	9	11
10	3	8
11	5	9
12	9	15

Table 3. Processing time and Due date of every job for example problem-II

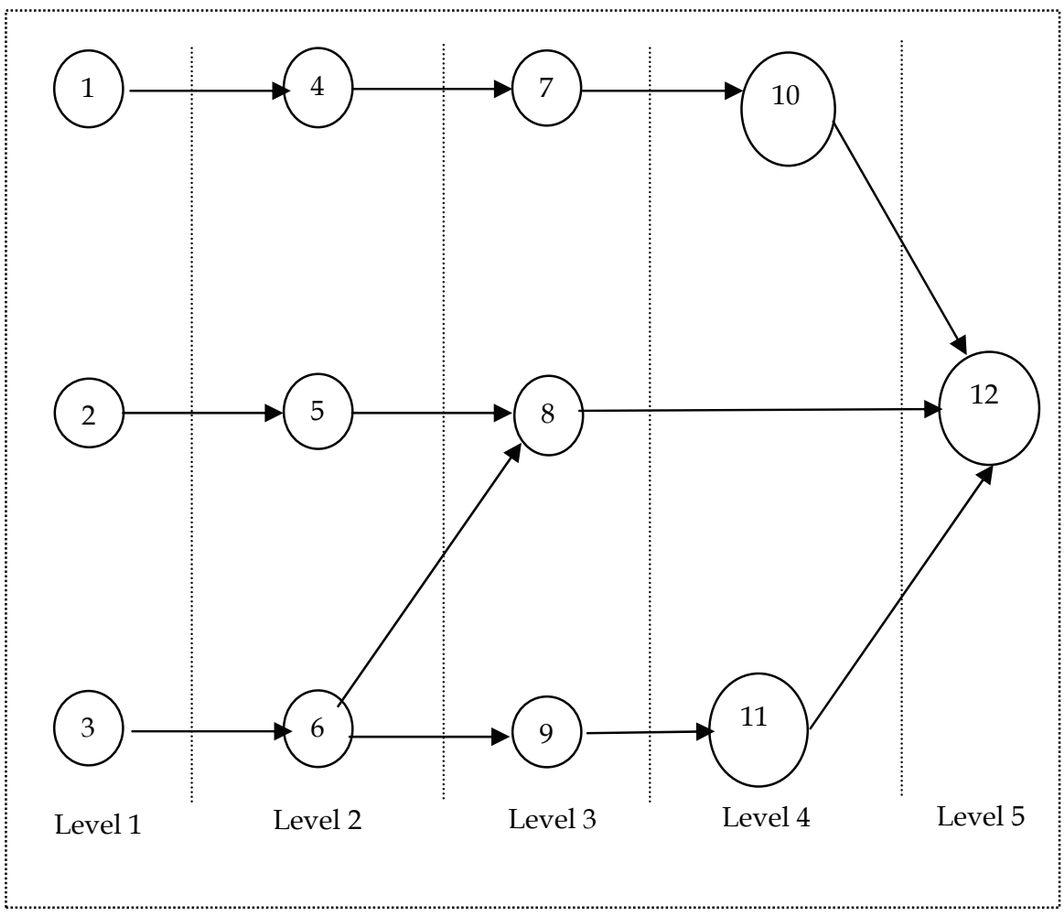

Figure 12. Precedence constraints of every job for example problem-II

Heuristic	Schedule	Total Earliness and Tardiness Cost
SPT	M1: 1-4-5-7-10-8 M2: 2-3-6-9-11-12	148,5
EDD	M1: 1-2-5-9-11-8-12 M2: 3-6-4-7-10	153
GA	M1: 1-4-7-10-9-11 M2: 3-2-6-5-8-12	144,5
SA	M1: 1-4-5-7-10-8-12 M2: 3-2-6-9-11	169,5

Table 4. The result of calculation for 12 X 2 problem size

8. Computational experimentation for scheduling with precedence constraints

The number of jobs used in the problems in this study were given in Table 5. In this table, i denotes the jobs and p_i is an integer processing time and w_i is an integer weight, which were generated from two uniform distributions. The function of [1, 10] and [1, 100] are to create low or high variations, respectively. TF, which is the relative range of due dates, RDD and Average tardiness factor, were selected from the set [0.1, 0.3, 0.5, 0.7, 0.9]. Here, d_i is an integer due date from the uniform distribution [P (1-TF-RDD/2), [P(1-TF+RDD/2)] and it was generated for each job i. In these expressions, P denotes total processing time. As summarized in Table 5, 1700 examples set were considered, totally. The problems were considered in 17 different sizes and for each size 100 different samples were examined. The parameters of the GA were given below. These parameters are firstly tried with different

Population size : 20, Crossover rate :%100,
Max generation : 100, Mutation rate :0.05.

Factors	**Settings**
Number of jobs	[10],[20],[30],[40],[50],[60],[70],[80],[90],[100] [120],[150],[170],[200],[220],[250],[300]
Processing time variability	[1-10] [1-100]
Weight variability	[1-10] [1-100]
Relative range of due dates	0.1, 0.3, 0.5, 0.7, 0.9
Average tardiness factor	0.1, 0.3, 0.5, 0.7, 0.9

Table 5. Experimental design

values and according the results of these experimental studies these parameters were determined as the best ones. In different studies, these parameters are determined like the ones obtained in this study. The obtained optimal solutions for different population sizes were given below in Figure 12 for the problem defined with 100x8 sizes. In Figure 13, the cost values for initial population, generation 50 and generation 100 were presented. These figures give clearly information about the selected parameters of GA. As seen in Figure 12,

to obtain optimal solution, the different population size values were applied. When the population size is selected as 20, the obtained optimal solution is found better than ones examined with other population sizes.

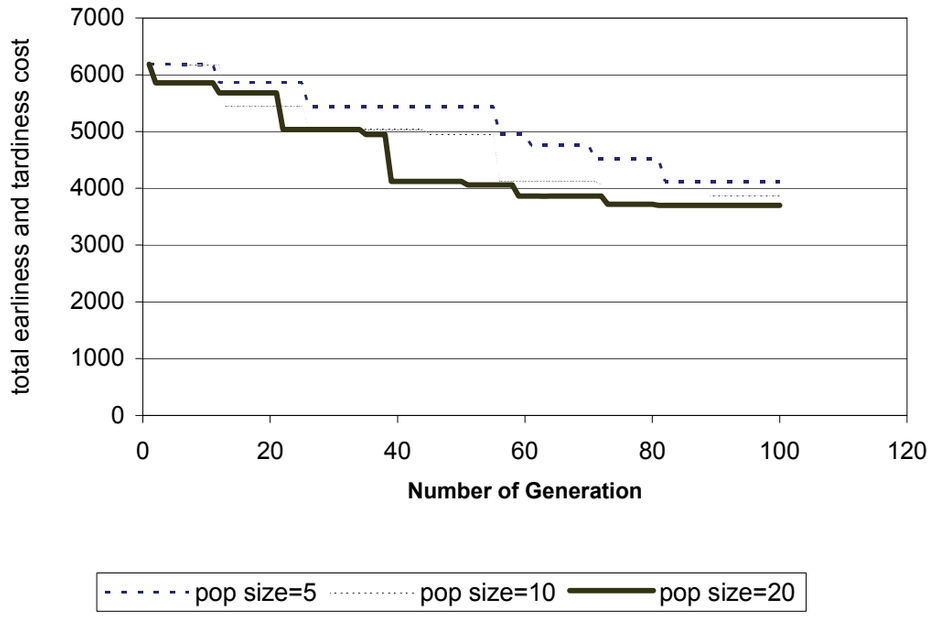

Figure 13. The obtained near optimal solutions according to the different population sizes

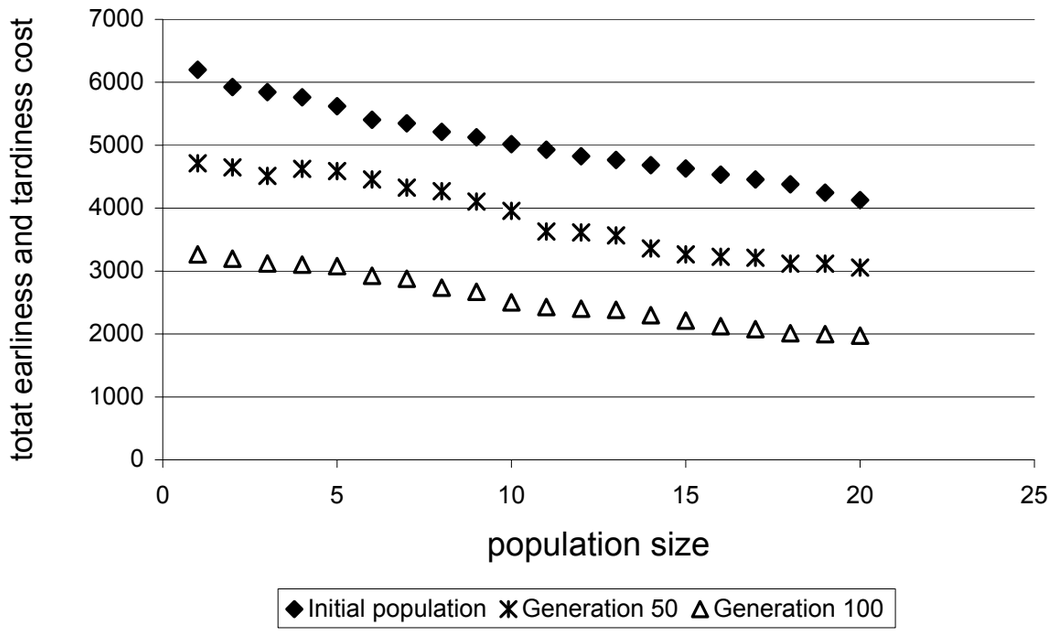

Figure 14. The obtained cost values for initial population, generation 50-100

The results have shown that GA has given better results than SA in large-size problems. SA has some weak points such as long running time and difficulty in selecting cooling parameter when the problem size becomes larger. A geometric ratio was used in SA as $T_{k+1} = \alpha T_k$, where T_k, and T_{k+1} are the temperature values for k and k+1 steps, respectively. Geometric ratio is used more commonly in practice. In this study, the initial temperature was taken 10000 and 0.95 was used for cooling ratio (α). In Table 6 and Table 8, the obtained solutions for different problem sizes were given

Problem size	Number of example	Average value of GA for total earliness and tardiness cost	Average Value of SA for total earliness and tardiness cost	CPU time for GA for an example (s)	CPU time for SA for an example (s)	t statistics
60 X 7	100	756.4	921.2	28.07	34.15	11.30
70 X 7	100	890.0	1056.7	32.71	44.19	12.47
80 X 8	100	1018.1	1263.6	39.03	48.22	15.21
90 X 8	100	1293.0	1512.8	43.35	54.17	16.23
100 X 8	100	1650.8	2004.2	62.28	73.05	17.46
120 X 8	100	1926.2	2137.9	78.05	91.33	19.33
150 X 8	100	2184.4	2410.5	92.17	102.09	21.96
170 X 8	100	2432.7	2985.0	100.02	114.43	22.07
200 X 8	100	3257.3	3863.3	118.34	136.57	24.97
220 X 8	100	3469.2	4112.4	127.28	151.48	25.35
250 X 8	100	3966.4	4698.9	139.11	178.12	29.46
300 X 8	100	5469.6	7282.7	152.22	196.47	31.45

Table 6. The results of the problems in different sizes for total earliness and tardiness cost

In Table 7 and Table 9, the 100 samples given for each problem size were evaluated and how many of the obtained results by using GA are better or equal to SA.. For each problem size, 100 different samples were used. GA and SA were applied to these samples. The average value of the obtained optimal solutions was revealed in the table. According to the average value, it is clearly seen that GA has given the better result. From the viewpoint of evaluating CPU time, the obtained result with GA is again better. All algorithms were coded in C++ and implemented on a Pentium IV 2.4 GHz computer.

Problem size	Number of examples	Number of examples for that GA is better than SA	Number of examples for that GA is equal to SA
60 X 8	100	91	9
70 X 7	100	95	5
80 X 8	100	98	2
90 X 8	100	100	0
100 X 8	100	100	0
120 X 8	100	100	0
150 X 8	100	100	0
170 X 8	100	100	0
200 X 8	100	100	0
220 X 8	100	100	0
250 X 8	100	100	0
300 X 8	100	100	0

Table 7. Comparison of the results of the examples according to the optimal values for total earliness and tardiness cost

Problem size	Number of example	Average value of GA for maximum flow time	Average Value of SA for maximum flow time	CPU time for GA for an example (s)	CPU time for SA for an example (s)	t statistics
60 X 7	100	72	78	18.03	22.21	13.45
70 X 7	100	85	93	26.07	30.18	15.68
80 X 8	100	96	108	32.09	39.43	17.13
90 X 8	100	119	134	39.01	51.19	19.86
100 X 8	100	132	142	45.15	63.05	18.94
120 X 8	100	148	161	54.45	71.45	19.73
150 X 8	100	176	183	62.22	76.39	22.12
170 X 8	100	189	202	70.56	83.55	24.28
200 X 8	100	217	230	81.30	90.57	24.88
220 X 8	100	239	255	92.12	103.49	27.35
250 X 8	100	264	292	102.37	114.42	29.49
300 X 8	100	286	305	129.21	142.47	33.57

Table 8. The results of the problems in different sizes for maximum flow time

Problem size	Number of examples	Number of examples for that GA is better than SA	Number of examples for that GA is equal to SA
60 X 8	100	94	6
70 X 7	100	95	5
80 X 8	100	100	0
90 X 8	100	100	0
100 X 8	100	100	0
120 X 8	100	100	0
150 X 8	100	100	0
170 X 8	100	100	0
200 X 8	100	100	0
220 X 8	100	100	0
250 X 8	100	100	0
300 X 8	100	100	0

Table 9. Comparison of the results of the examples according to the optimal values for maximum flow time

9. Conclusions

The genetic algorithms (GA) have the great advantage and success in the solution of NP problems. There are various important applications on this way. In this study, the job with n-number of precedence constraints is assigned minimizing total earliness and tardiness and maximum flow time on m-number of parallel machine. Genetic algorithms and simulated annealing methods were used to find the solutions, which minimizes the total earliness and tardiness costs. In GA, the solution alternatives, which were obtained by using genetic operators, were investigated to understand that if they are feasible or not and the feasible ones according to precedence constraints were considered. The way, trying to make infeasible solutions feasible, was not selected. Likewise, obtained infeasible solutions were not evaluated. Again any study about making these infeasible solutions feasible was not done. According to the results obtained by using GA and SA methods, it was evidently observed that GA algorithm is more successful. Especially for larger problem sizes, it is seen that GA gives results better than SA.

10. References

Kasahara H., Narita S., Parallel processing of robot arm control computation on a multi microprocessor system, *IEEE J. of Robotics Automation* vol. RA-1, no.2, pp. 104-113, June 1985.

Kanjo, -Y., Ase, -H., Robust scheduling in a multi robot welding system, Transaction of the Institute of systems, *Control and Information Engineers*. 16(8), pp.369-376, 2003.

Jun, S.Z., Ying, Z.J., A genetic algorithm based approach to intelligent optimization for scheduling dual resources with robots, *Robot*, 24(4), pp.342-357, 2002.

Zacharia, P.T., Asparagathos, N.A., Optimal robot task scheduling based on genetic algorithms, *Robotics and Computer Integrated Manufacturing*, 21(1), pp. 67-79, 2005.

E.M. Arkin, R.O. Roundy, Weighted-Tardiness scheduling on parallel machines with proportional weights, *Operational Research*, 39, pp. 64-81. 1991.

H. Emmons, Scheduling to a common due date on parallel uniform processors, *Naval Research Logistics*, 24, pp. 803-810, 1987.

A. Guinet, Scheduling independent jobs on uniform parallel machines to minimize tardiness criteria, *Journal of Intelligent Manufacturing*, 6, pp. 95-103, 1995.

R.M. Karp, Reductibility among combinatorial problems: complexity of computer computations, New york, Plenum press, pp. 85-103, 1972.

J.M.J. Schutten R.A.M. Leussink, Parallel Machine scheduling with release dates, and familiy setup times, International Journal of Production Economy, 46-47, pp. 119-125,1996.

S.A. Tamimi, V.N. Rajan, Reduction of Total weighted tardiness on uniform machines with sequence dependent setups, *Industrial Engineering Research – Conference Proceedings,* pp. 181-185, 1997.

V.A. Armentano, D.S. Yamashita, Tabu Search for scheduling on identical parallel machines to minimize mean tardiness, *Journal of Intelligent Manufacturing*, 11, pp. 453-460, 2000.

Kim K.H., Kim D.W., Unreleated parallel machine scheduling with setup times using simulated annealing, *Robotics and computer Integrated Manufacturing*, Volume 18, Issues 3-4, Pages 223-231,2002.

Min L., Cheng W., A genetic algorithm for minimizing the makespan in case of scheduling identical parallel machines, *Artificial Intelligence in Enginering*, 13, pp. 399-403, 1995

Chen C.L., Lee C.S.G., & Hou, E.S.H., Efficient scheduling Algorithm algorithms for robot inverse dynamics computation on a multiprocessor system, *IEEE Transaction on Systems, Man, and Cybernetics,* Vol..18, pp. 729-743, Dec. 1988.

J.C.Ho, Y.-L.Chang, Heuristics for minimizing mean tardiness for m parallel machines, *Naval Research Logistics*, 38, pp. 367-381, 1991.

L.J. Wilkerson, J.D. Irwin, An improved algorithm for scheduling independent jobs, *AIIE Transactions*, 3, pp. 239-245, 1971.

C. Koulamas, Decomposition and hybris Simulated annealing heuristics for the parallel machine total tardiness problem, *Naval Research Logistics*, 44, pp. 109-125, 1997.

K. Chen, J.S. Wong & J.C. Ho, A heuristic algorithm to minimize tardiness for parallel machines *Proceedings of ISMM\International Conference*. ISSM-ACTA Press, Anaheim CA, USA pp. 118-121,1997.

B. Alidaee, D. Rosa, Scheduling parallel Machines to weighted and un-weight tardiness, *Computers Operational Research*, 24 (8), pp. 775-788, 1997.

K.R.Baker, J.W. Bertrand, A Dynamic priority rule for sequencing against due-date, *Journal of Operational Management*, 3, pp. 37-42, 1982.

M. Azizoğlu, O.Kirca, Tardiness minimization on parallel machines, *International Journal of Production Economics*, 55, pp. 163-168, 1998.

S.E. Elmaghraby, S.H. Park, Scheduling jobs on a number of identical machines, *AIIE Transactions*, 6, pp. 1-13, 1974.

J.W. Barnes, J.J. Brennan, An Improved algorithm for independent jobs to reduce mean finishing time , *AIIE Transactions*, 17, pp. 382-387, 1977.

H. Emmons, M. Pinedo, Scheduling stochastic jobs, with due dates on parallel machines, *Europan Journal of Operational Research*, 47, pp. 49-55, 1990.

T.Cakar , R.Koker & H.I.Demir, Parallel Robot Scheduling to Minimize Mean Tardiness with Precedence Constraints Using a Genetic Algorithm, *Advance in Engineering Software,* Vol 39, No 1, pp. 47-54, January 2008.

K.R.Baker, *Introduction to Sequencing and Scheduling*, John Wiley & Sons, New York,1974

Parallel Manipulators, New Developments

Edited by Jee-Hwan Ryu

ISBN 978-3-902613-20-2

Hard cover, 498 pages

Publisher I-Tech Education and Publishing

Published online 01, April, 2008

Published in print edition April, 2008

Parallel manipulators are characterized as having closed-loop kinematic chains. Compared to serial manipulators, which have open-ended structure, parallel manipulators have many advantages in terms of accuracy, rigidity and ability to manipulate heavy loads. Therefore, they have been getting many attentions in astronomy to flight simulators and especially in machine-tool industries.The aim of this book is to provide an overview of the state-of-art, to present new ideas, original results and practical experiences in parallel manipulators. This book mainly introduces advanced kinematic and dynamic analysis methods and cutting edge control technologies for parallel manipulators. Even though this book only contains several samples of research activities on parallel manipulators, I believe this book can give an idea to the reader about what has been done in the field recently, and what kind of open problems are in this area.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen