Scheduling with Setup Time Matrix for Sequence Dependent Family

Senthilvel Nataraj[1], Umamaheshwari Sankareswaran[2], Hemamalini Thulasiram[3], and Senthiil Varadharajan[4]
[1]Department of Computer Science, Anna University, India
[2]Department of Electronics and Communication Engineering, Coimbatore Institute of Technology, Affiliated to Anna University, India
[3]Department of Mathematics, Government Arts and Science College, Bharathiar University, India
[4]Production Engineering Department, Saint Peter's University, India

Abstract: *We consider a problem of scheduling n jobs in k families on a single machine subjected to family set-up time to minimize the overall penalty. This paper proposes three heuristic approaches based on neighbourhood structures using setup time matrix. These three approaches minimize the maximum penalty which in turn minimizes the total penalty. Inserting the Idle Time initially (ITF approach) or between the families perform efficiently on large instances. The computational results prove the efficiency of the algorithm.*

Keywords: *Scheduling, sequence dependent scheduling, heuristic algorithm, idle time insertion.*

Received January 1, 2016; accepted June 26, 2016

1. Introduction

In many practical situations, grouping of jobs is necessary due to the technological feature and the processing technique of the jobs. No setup time is required between the production of two products of the same group. In some situations, the setup time is considered to be negligible or part of the processing time. However, in many environments the setup-time is significant. Industries like parts manufacturing, processing industries, etc., need setup time to change from one model to another, one size to another, and so on and so forth. The consideration of setup time tends to group the jobs into families. Hence, we consider a single machine scheduling problem with family set-up time. A set of n jobs is partitioned into k families that have to be scheduled on a single machine. A set-up time is incurred by the machine between the completion of i^{th} family and the starting of j^{th} family. Here, a job does not need a setup time if the next job is of the same family. Thus, to minimize the total penalty, the jobs are scheduled in their respective families according to their processing time and penalties. Further, the families are scheduled according to their setup times. Using Setup Time Matrix (STM), an algorithm is derived to schedule the jobs which minimizes the total completion time and in turn minimizes the net penalty. The reason behind framing the STM matrix is that, the matrix gives a clear structure in the sequencing of j^{th} family next to the i^{th} family and hence helps to compare all the setup times after the i^{th} family given in $(i+1)^{th}$ row. Further, the algorithm deletes $(i+1)^{th}$ row when the i^{th} family is scheduled, leading to the reduction in matrix size.

2. Problem Formulation

A set of n jobs to be processed on a single machine are partitioned into k families, where all the jobs are available concurrently. The jobs are grouped based on the characteristic of the machine and their processing requirements. The number of jobs in each family need not be equal.

Let n_r be the number of jobs in the r^{th} family such that $\sum_{i=1}^{k} n_i = n$, and $P_r = \sum_{i=1}^{n_r} p_{ri}$ is the total processing time of r^{th} family. Here, p_{ri} is the Processing time of i^{th} job in r^{th} family. At a time, the machine can handle at most one job and preemption of job in processing is not permitted. The machine requires a setup time before starting the first job from each family and between two consecutive jobs from two different families. No setup time is required if the consecutive jobs are from the same family.

The Setup Time in matrix structure is given by

$$STM = \left[a_{ij} \right] \quad for\, i, j = 1, 2, 3, \ldots k$$
$$where\, a_{ij} = \begin{cases} s_{(i-1)j} & for\ all\, i, j\, except\, i = j \\ \infty & for\, i = j \end{cases} \tag{1}$$

Processing of i^{th} family after the i^{th} family is meaningless.

ie.,

$$STM = \begin{bmatrix} s_{01} & s_{02} & s_{03} & \cdots & s_{0k} \\ \infty & s_{12} & s_{13} & \cdots & s_{1k} \\ s_{21} & \infty & s_{23} & \cdots & s_{2k} \\ s_{31} & s_{32} & \infty & \cdots & s_{3k} \\ \vdots & \vdots & \vdots & & \vdots \\ s_{k1} & s_{k2} & s_{k3} & \cdots & \infty \end{bmatrix} \qquad (2)$$

Where s_{ij} is the set up time after the ith family and before the jth family.

In addition, each job is associated with earliness and tardiness cost. The job ^{J}ri of r^{th} family has a due date d_{ri} and incurs a tardiness penalty B_{ri}=0 and earliness penalty a_{ri} for each time unit if its completion time C_{ri} is less than d_{ri}. If its completion time C_{ri} is greater than d_{ri}, the job incurs no earliness penalty (a_{ri}=0), but incurs tardiness penalty B_{ri} for each time unit. The completion time C_{ri} includes the time taken to process (i-1) jobs summed with the setup time of the machine before commencing r^{th} family. C_{ri} is computed using the equation

$$C_{ri} = \sum_{j=1}^{r-1} s^{(j)} + \sum_{k=1}^{i} p^{(k)} \qquad (3)$$

Where, $s^{(r)}$ is the setup time of r^{th} family after completing $(r\text{-}1)^{th}$ family in the optimal sequence and $p^{(k)}$ is the processing time of k^{th} job in the optimal sequence.

The objective of the proposed algorithm is to minimize the maximum penalty

$$z_{\max} = \max\{z_i\}$$
$$where\ z_i = \alpha_{ri}\left|C_{ri} - d_{ri}\right| + \beta_{ri}\left|C_{ri} - d_{ri}\right|\ and\ \ i = 1,2,\ldots n \qquad (4)$$

3. Literature Review

Gave a comprehensive review on scheduling problems with setup times and classified the problem according to shop environments such as single machine [2], parallel machines, flow-shops and job shops. Taner *et al.* [16] proposed two approaches, a two-steps neighborhood search procedure and an implicit enumeration scheme. Out of these, the two-steps neighborhood search procedure performs efficiently in large instances. Schaller [11] solved a single machine scheduling problem with family setup time in minimizing the total tardiness by branch-and-bound procedure with and without group technology assumption. Erel and Ghosh [6] described a production schedule to minimize the total order lead time. They suggested that the problem is solvable in polynomial time for special cases. Allahverdi *et al.* [3] presented an extensive review of more than 300 papers on scheduling models with setup times. Schaller and Gupta [12] proposed two algorithms and explore how the procedure affects the total earliness and tardiness by implementing lean production methods. Their investigation showed that scheduling jobs without group technology assumption reduces the total earliness and tardiness. Uzsoy and Velásquez [17] introduced three different formulations of production scheduling problem with sequence dependent and time dependent setup times on a single machine. Sels and Vanhoucke [13] developed a hybrid genetic algorithm by combining different local search neighborhood structures. In their paper, each job is characterized by its processing time, release time, and due date with the objective of minimizing the maximum lateness. Zhou and Liu [18] developed two heuristics: Time forward and Time backward algorithms with the incorporation of the new property. Their experimental computation showed a significant improvement in the execution time. Sabouni and Logendran [10] solved the problem of minimizing the make-span on a single machine with carryover sequence-dependent setup times by branch and bound algorithm and a lower-bounding structure. Croce *et al.* [5] proposed a matheuristic algorithm for one-machine total completion time sequencing problem subject to release times. Srikanth *et al.* [14] used Ant Colony Optimization (ACO) for the scheduling problem. They arrived at a feasible schedule for a task set on heterogeneous processors ensuring fair load balancing across the processors within a reasonable amount of time. Munir *et al.* [9] proposed two novel approaches for the task scheduling problem. Ababneh *et al.* [1] experimentally evaluated the Priority Genetic Algorithms (PGA). Ijaz *et al.* [8] proposed an approach for the efficient mapping of the Directed Acyclic Graph (DAG)-based applications, an approach that takes into account the lower and upper bounds for the start time of the tasks. Atoum and Al-Rababaa [4] presented a solution to the problem of multiple warehouses scheduling using the steady state genetic algorithm. Steccoa *et al.* [15] developed a number of heuristics to minimize maximum lateness with family-dependent-setup times.

4. STM Heuristic Approach with Setup Time Matrix

The heuristic includes the local neighborhood search in each row of the STM to minimize the completion time in scheduling *k* families. In each family, each job has specific due date, processing time, earliness penalty and lateness penalty. If the jobs are processed according to their due dates, the setup times repeat several times and increase the overall completion time. With the setup times of *k* families, the Setup Time Matrix (STM) is constructed. This STM consists of *k+1* rows and *k* columns. The first row of the matrix gives the initial setup times required for the machine and the remaining rows give the setup times required between the jobs from two different families for the machine. The n_r Jobs of each family are scheduled using the Ratio Scheduling Algorithm (RSA) proposed by [7], according to its ratio between processing time and penalty within their family. The proposed algorithm examines the setup time of each element from the first row and selects the highest among the STM elements. If one of the elements a_{ij}, (*where* $i \neq 1$) is the highest, then the setup time s_{0j} is chosen, else if a_{ij}, (*where* $i=1$) is the highest, then the least s_{oj} is chosen. That is, *j*th family will be the first in the sequence. Since *j*th family is being processed first, setup time to switch over once again to the *j*th family is no longer necessary, so the *j*th column and the1st row is removed, thus reducing the STM to $k \times (k\text{-}1)$ matrix. In the reduced matrix, a search is performed once again for the highest setup time. If a_{mn}, (*where* $m \neq (j+1)$) is the highest, then the setup time s_{jn} is chosen, else if a_{mn}, (*where* $m = (j+1)$) is the highest then the least s_{jn} is chosen. This procedure is repeated until all the families are selected. At each stage, the completion time C_{ri} and penalty z_i are calculated. Once the initial sequence is derived, $z^1_{\max} = \max\{z_i\}$ is calculated. The second sequence is derived by interchanging the last two families, and $z^2_{\max}$ is calculated. If $z^2_{\max} \geq z^1_{\max}$, then the initial sequence is the optimal sequence. Otherwise, if $z^2_{\max} < z^1_{\max}$, the algorithm is repeated for the last three families. That is, considering the second sequence as the best sequence, the algorithm progresses to the next best sequence with the reduced 3×2 ST Matrix. Here, instead of choosing the highest, the next highest among the STM elements is selected. In this case, the algorithm interchanges only the last three families to get the best sequence. $z^3_{\max}$ is then calculated and compared with $z^2_{\max}$. If $z^3_{\max} < z^2_{\max}$, then the algorithm repeats for the last four families and the next highest among the STM elements is selected. Once we arrive at $z^{i+1}_{\max} \geq z^i_{\max}$, the *i*th sequence is the optimal sequence. Even though the approach is iterative, the computational time is less. Since the setup times are arranged in the form of matrix, after the *i*th family is selected, its corresponding row and column are deleted and the size of the matrix is reduced. Also, while approaching the optimal sequence, the second iteration examines the reduced 3x2 ST matrix and the third iteration examines the 4x3 ST matrix and so on.

5. Idle Time Insertion Between Families (ITF Approach)

STM Heuristic approach finds an optimal schedule for the given families. To further improve the optimized sequence, ITF approach inserts idle time between families. The ITF approach compares the completion time $C^{(r)}$ of the *r*th family with the mean due date $m^{(r)}$ of that family. If the completion time is less than the mean due date, then an idle time of $I^{(r)} = [m^{(r)} - C^{(r)}]$ is inserted before processing *r*th family, which certainly reduces the earliness penalty of the jobs in *r*th family. ITF algorithm differs from STM heuristic approach only in the insertion of idle time between the families. Also, the completion time of *i*th job of *r*th family is calculated as

$$C_{ri} = I^{(r-1)} + \sum_{j=1}^{r-1} s^{(j)} + \sum_{k=1}^{i} p^{(k)} \qquad (5)$$

6. Idle Time Insertion Between the Jobs (ITJ Approach)

We now describe ITJ approach, which is similar to ITF approach; however it differs in the location for the insertion of the idle time. In ITJ approach, the idle time is inserted between the jobs within a family. The algorithm compares the completion time of each job with its due date. If the completion time of a job is less than its due date, then an idle time of $I_{ri} = [d_{ri} - C_{ri}]$ is inserted before starting the *i*th job of *r*th family. Even though this algorithm reduces earliness penalty, the overall completion time is increased. However, the algorithm works particularly good when the due date is very large. Here the completion time is calculated as

$$C_{ri} = \sum_{j=1}^{r-1} s^{(j)} + \sum_{k=1}^{i} p^{(k)} + \sum_{u=1}^{r-1}\sum_{v=1}^{i-1} I_{uv} \qquad (6)$$

7. Computational Experiments

This section gives the results gathered from STM heuristic approach, ITF approach, and ITJ approach. These algorithms were tested on 144 problem sets of various sizes in terms of families, ranging from 2 to 6, and the jobs within each family ranging from 30 to 70. The earliness and lateness penalties are generated randomly in a uniform distribution between 1 and 10. The setup time and the processing time p_{ri} are generated randomly using a uniform distribution over the integers 1 to 30. The due date d_{ri} is generated following the discrete uniform distribution,

$$d_{ri} = Uniform[0, \lambda \sum_{r=1}^{k} \sum_{i=1}^{n_r} p_{ri}] \qquad (7)$$

Here, λ is the parameter used to specify due date by the user.

For computational purpose, λ starts with 0.2, and is incremented by a value of 0.2 and reaches upto a maximum value of 3.2. For higher r values the variance among the due dates is higher. The objective of this test is to analyze the efficiency of the proposed three approaches and to examine the importance of idle time between the families and between the jobs.

Table 1. $z_{\max}$ values of SPT, ITF and ITJ Approaches for 30 jobs.

λ	30 jobs in 2 families			30 jobs in 4 families			30 jobs in 6 families		
	SPT $z_{\max}$	ITF $z_{\max}$	ITJ $z_{\max}$	SPT $z_{\max}$	ITF $z_{\max}$	ITJ $z_{\max}$	SPT $z_{\max}$	ITF $z_{\max}$	ITJ $z_{\max}$
0.2	171	162	214	217	203	229	242	227	269
0.4	189	172	226	221	198	235	256	243	271
0.6	195	183	238	234	197	246	264	251	283
0.8	209	192	246	245	228	257	269	257	294
1.0	214	202	245	258	231	267	278	265	303
1.2	229	213	237	264	247	281	281	298	317
1.4	235	224	221	278	250	294	299	284	326
1.6	241	231	257	289	253	305	308	298	336
1.8	257	195	274	291	266	318	317	309	348
2.0	261	191	263	301	281	329	327	314	357
2.2	278	252	282	307	277	337	336	325	361
2.4	283	272	298	317	311	341	341	334	372
2.6	294	282	311	329	292	356	352	356	385
2.8	304	298	308	337	318	368	362	345	391
3.0	316	302	312	341	326	379	378	365	407
3.2	327	312	345	356	331	376	387	375	419

Table 2. $z_{\max}$ values of SPT, ITF and ITJ Approaches for 50 jobs.

λ	50 jobs in 2 families			50 jobs in 4 families			50 jobs in 6 families		
	SPT $z_{\max}$	ITF $z_{\max}$	ITJ $z_{\max}$	SPT $z_{\max}$	ITF $z_{\max}$	ITJ $z_{\max}$	SPT $z_{\max}$	ITF $z_{\max}$	ITJ $z_{\max}$
0.2	432	380	453	449	412	478	476	424	496
0.4	435	375	461	441	409	482	482	412	508
0.6	446	361	469	432	398	494	492	406	518
0.8	452	348	477	427	391	501	523	392	527
1.0	471	334	483	416	373	523	534	387	539
1.2	479	321	495	409	365	537	541	372	546
1.4	486	316	499	402	352	548	552	364	558
1.6	495	309	508	398	342	569	563	351	562
1.8	518	298	515	386	339	573	578	349	579
2.0	527	286	523	369	321	582	581	335	584
2.2	541	271	534	356	316	597	589	327	592
2.4	552	265	542	347	309	609	599	315	603
2.6	561	254	558	332	298	615	608	309	617
2.8	569	243	561	321	286	637	614	298	629
3.0	576	245	571	314	274	648	625	277	638
3.2	587	256	598	328	267	657	639	312	643

Table 3. $z_{\max}$ values of SPT, ITF and ITJ Approaches for 70 jobs.

λ	70 jobs in 2 families			70 jobs in 4 families			70 jobs in 6 families		
	SPT $z_{\max}$	ITF $z_{\max}$	ITJ $z_{\max}$	SPT $z_{\max}$	ITF $z_{\max}$	ITJ $z_{\max}$	SPT $z_{\max}$	ITF $z_{\max}$	ITJ $z_{\max}$
0.2	778	762	789	833	792	897	965	854	978
0.4	789	751	794	849	783	904	978	843	984
0.6	795	649	808	852	776	913	981	831	993
0.8	816	737	816	867	764	928	992	821	1012
1.0	825	728	835	872	752	937	1014	815	1023
1.2	831	715	842	889	741	947	1023	802	1038
1.4	849	704	856	894	739	952	1032	795	1049
1.6	857	692	869	906	728	958	1046	783	1051
1.8	861	684	872	919	716	961	1052	772	1066
2.0	873	671	884	923	703	972	1062	761	1075
2.2	881	668	891	937	692	992	1078	752	1082
2.4	896	657	908	948	684	1007	1081	741	1093
2.6	903	641	916	953	674	1021	1094	738	1109
2.8	917	639	927	962	662	1030	1116	725	1116
3.0	926	642	932	978	651	1042	1127	714	1129
3.2	934	647	945	986	643	1123	1145	734	1137

The performance of each algorithm is analyzed individually by examining $z_{\max}$ values for λ between 0.2 and 3.2, which is the factor that determines the due date. For sequencing the jobs within the family, Ratio Scheduling Algorithm isused, which has the running time of *O(2log n)*, where *n* is the total number of jobs. In Tables 1, 2, and 3 the $z_{\max}$ values are tabulated for SPT, ITF and ITJ algorithms. The values show that the ITF approach performs better than SPT and ITJ approaches. This shows that the insertion of idle time between the families influences the relative performance of the algorithm. Moreover, the insertion of idle time within the families increases the maximum penalty.

Table 4. Mean $z_{\max}$ values.

Mean $Z_{\max}$		SPT	ITF	ITJ
30 jobs	2 families	250	230	267
	4 families	287	263	307
	6 families	312	303	340
50 Jobs	2 families	508	304	515
	4 families	383	341	566
	6 families	562	352	571
70 jobs	2 families	858	687	868
	4 families	911	719	974
	6 families	1049	780	1058

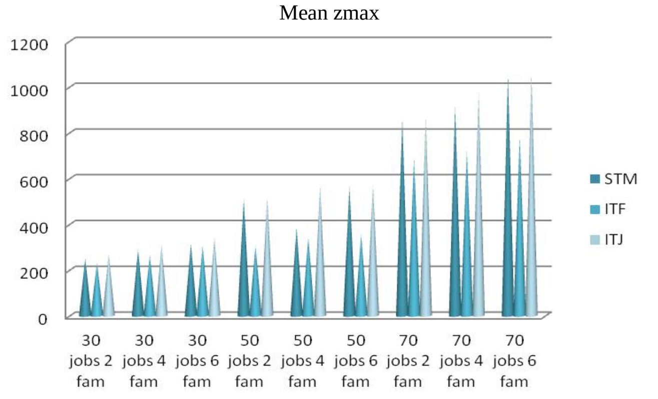

Figure 1. Comparison of Mean $z_{\max}$ values.

Table 5. Average number of early jobs, on-time jobs and late jobs.

		Average Number of early jobs			Average Number of on-time jobs			Average Number of late jobs		
		SPT	ITF	ITJ	SPT	ITF	ITJ	SPT	ITF	ITJ
30 jobs	2 families	23	6	4	0	13	10	7	11	16
	4 families	21	5	4	2	16	13	7	9	13
	6 families	24	3	3	1	15	13	5	12	14
50 Jobs	2 families	39	7	6	0	19	15	11	24	29
	4 families	35	7	5	1	26	21	14	17	24
	6 families	41	5	7	3	31	26	6	14	17
70 jobs	2 families	58	15	12	2	19	14	10	36	44
	4 families	56	11	17	5	25	24	9	34	29
	6 families	52	7	9	7	37	29	11	26	32

Table 4 gives the Mean $z_{\max}$ values of the three proposed approaches; Figure 1 clearly shows that the ITF approach yields better solution in reducing the maximum penalty, almost in all the instances. In table 5, the average number of early jobs, the average number of on-time jobs, and the average number of late jobs are tabulated using the formula, average number of on-time jobs=(Total number of on-time jobs)/16, average number of jobs with earliness penalty = (Total number of jobs with earliness penalty)/16, and average number of jobs with lateness penalty = (Total number of jobs with lateness penalty)/16. While examining the number of on-time jobs, ITF performs the best among the three proposed approach. STM approach gives a sequence with more number of earliness penalty jobs, whereas ITJ approach gives a sequence with more number of tardiness jobs. Also in Table 4, mean z_{max} value of STM and ITJ are large when compared with values from ITF approach. This shows that ITF approach performs better in all aspects. Also, computational times of all the proposed approaches are minimal. Therefore, all the three approaches are time efficient. For higher values of the parameter λ, z_{max} values for ITF approach are drastically less, as against lesser values of λ. The previous four sentences imply that the due date influences lateness penalty in ITJ approach and earliness penalty in STM approach. To demonstrate this, test of significance for difference of means of two large samples are used. The λ values are partitioned in to two sample sets, i.e., with $\lambda = 0.2 to 1.6$ having 72 sequencing problems of various sizes as sample set 1, and λ=1.8 to 3.2 having another 72 problems as sample set 2. Let $\overline{z}_1 = 124$ and $\sigma_1^2 = 6145.39$ be the mean and variance of sample set 1 of size 72, and $\overline{z}_2 = 98$ and $\sigma_2^2 = 5930.006$ be the mean and variance of sample set 2 of size 72. The calculated value of the test statistics is

$$t = \frac{\overline{z}_1 - \overline{z}_2}{\sqrt{\frac{\sigma_1^2}{n} + \frac{\sigma_2^2}{n}}} = 2.2393 \quad (8)$$

Whereas the value of t at 5% level of significance is 1.96. Since the calculated t value is greater, the null hypothesis that there is no significant difference in z_{max} values in two samples is thus rejected. This shows that there is significant difference in the values of z_{max} for larger values of λ. Therefore, the due date parameter λ influences the z_{max} values. Hence, when the due dates of the jobs are very large when compared against their processing time, ITJ approach gives the optimal solution; however, if the due dates are merely large, ITF approach gives the optimal solution. If the due dates are in range (more or less equal) with their processing time, STM approach works better than other two approaches. Thus ITF performs the best, both in terms of z_{max} and in number of on-time jobs. STM approach is the next best approach, when the scheduler does not want to insert the idle time.

7. Conclusions

In this paper, three approaches are proposed. STM approach is the original method which uses ST matrix for its computation. ITF approach is the extension of STM approach which includes insertion of idle time before processing the first family or insertion of idle time between the families. ITJ, on the other hand, is the extension of STM approach wherein we insert the idle time between the jobs in a family. Upon extensive computation and comparison with other approaches, ITF approach justifies that the idle time insertion is highly effective, particularly when the due date is large. It gives an optimal schedule with maximum number of on-time jobs. In industries like Food processing, the goods will be delivered immediately after preparation. Also season based industries like textiles, cosmetics, fireworks, etc., know their demand and due date tentatively. Instead of wasting the cost in inventory, Idle time may be inserted and the completion time may coincide with the due date in some cases. The proposed algorithms can be extended to different job environments.

References

[1] Ababneh M., Hassan S., and Bani-Ahmad S., “On Static Scheduling of Tasks in Real Time Multiprocessor Systems: An Improved GA-Based Approach,” *The International Arab Journal of Information Technology*, vol. 11, no. 6, pp. 560-572, 2014.

[2] Allahverdi A., Gupta J., and Aldowaisan T., “A Review of Scheduling Research Iinvolving Setup Considerations,” *Omega*, vol. 27, no. 2, pp. 219-239, 1999.

[3] Allahverdi A., Ng C., Cheng T., and Kovalyov M., “A Survey of Scheduling Problems with Setup Times or Costs,” *European Journal of Operational Research*, vol. 187, no. 3, pp. 985-1032, 2008.

[4] Atoum J. and Al-Rababaa M., “Multiple Warehouses Scheduling Using Steady State Genetic Algorithms,” *The International Arab Journal of Information Technology*, vol. 7, no. 3, pp. 310-316, 2010.

[5] Croce F., Salassa F., and T’Kindt V., “A Hybrid Heuristic Approach for Single Machine Scheduling with Release Times,” *Computers and Operations Research*, vol. 45, pp. 7-11, 2014.

[6] Erel E. and Ghosh J., “Customer Order Scheduling on a Single Machine with Family Setup Times: Complexity and Algorithms,” *Applied Mathematics and Computation*, vol. 185,

pp. 11-18, 2007.

[7] Hemamalini T., Senthilvel N., and Somasundaram S., "Scheduling Algorithm to Optimize Jobs in Shop Floor," *Journal of Mathematics and Statistics*, vol. 6, no. 4, pp. 416-420, 2010.

[8] Ijaz S., Munir E., Anwar W., and Nasir W., "Efficient Scheduling Strategy for Task Graphs in Heterogeneous Computing Environment," *The International Arab Journal of Information Technology*, vol. 10, no. 5, pp. 486-492, 2013.

[9] Munir E., Ijaz S., Anjum S., Khan A., Anwar W., and Nisar W., "Novel Approaches for Scheduling Task Graphs in Heterogeneous Distributed Computing Environment," *The International Arab Journal of Information Technology*, vol. 12, no. 3, pp. 270-277, 2015.

[10] Sabouni M. and Logendran R., "A Single Machine Carryover Sequence-Dependent Group Scheduling in PCB Manufacturing," *Computers and Operations Research*, vol. 40, no. 1, pp. 236-247, 2013.

[11] Schaller J., "Scheduling on a Single Machine with Family Setups to Minimize Total Tardiness," *International Journal of Production Economics*, vol. 105, no. 2, pp. 329-364, 2007.

[12] Schaller J. and Gupta J., "Single Machine Scheduling with Family Setups to Minimize Total Earliness and Tardiness," *European Journal of Operational Research*, vol. 187, no. 3, pp. 1050-1068, 2008.

[13] Sels V. and Vanhoucke M., "A Hybrid Genetic Algorithm for the Single Machine Maximum Lateness Problem with Release Times and Family Setups," *Computers and Operations Research*, vol. 39, no. 10, pp. 2346-2358, 2012.

[14] Srikanth U., Maheswari U., Palaniswami S., and Siromoney A., "Task Scheduling Using Probabilistic Ant Colony Heuristics," *The International Arab Journal of Information Technology*, vol. 13, no. 4, pp. 375-379, 2016.

[15] Steccoa G., Francois J., and Moretti E., "A Branch-and-Cut Algorithm for a Production Scheduling Problem with Sequence-Dependent and Time-Dependent Setup Times," *Computers and Operations Research*, vol. 35, no. 8, pp. 2635-2655, 2008.

[16] Taner M., Hodgson T., King R., and Schultz S., "Satisfying Due-Dates in the Presence of Sequence Dependent Family Setups with a Special Comedown Structure," *Computers and Industrial Engineering*, vol. 52, no. 1, pp. 57-70, 2007.

[17] Uzsoy R. and Velásquez J., "Heuristics for Minimizing Maximum Lateness on a Single Machine with Family-Dependent Set-up Times," *Computers and Operations Research*, vol. 35, no. 6, pp. 2018-2033, 2008.

[18] Zhou S. and Liu Z., "A Theoretical Development for the Total Tardiness Problem and its Application in Branch and Bound Algorithms," *Computers and Operations Research*, vol. 40, no. 1, pp. 248-252, 2013.

Senthilvel Nataraj received his post graduate degree (M.S By Research) in computer sciences and engineering from Anna University, Chennai, Tamilnadu, India in 2008. He completed his B.E degree in computer science and engineering from Regional Engineering College, Tiruchirappalli, Tamilnadu, India in 1997. Currently, he is an Assistant Professor (Senior Grade) in the department of Computer Science & Engineering at Coimbatore Institute of Technology, Coimbatore, Tamilnadu, India. His research interests include Scheduling Algorithms, Optimization, Remote Sensing & Image Processing.

Umamaheshwari Sankareswaran received her doctoral degree in Electrical and Electronics Engineering with specialization in Biometrics, from Bharathiar University, Coimbatore, Tamilnadu, India. She got her Master Degree in Applied Electronics from Coimbatore Institute of Technology in 1991 and Bachelor's Degree in Electronics and Communication Engineering from Government College of Technology in 1985, Tamilnadu, India. She is presently working as Professor in the Department of Electronics and Communication Engineering, Coimbatore Institute of Technology, Coimbatore, Tamilnadu, India. She holds 29 years of teaching experience. Her research interests are VLSI Design, Optimization, Image and Signal Processing, Embedded Systems and Wireless Communication.

Hemamalini Thulasiram received her PhD. degree in Mathematics from Anna University, Chennai, Tamilnadu, India in 2011. She completed her Master of Science degree in Mathematics from Thiyagaraja College of Engineering, Madurai, Tamilnadu, India in 1999. Currently, she is an Assistant Professor in the department of mathematics at Government Arts and Science College, Coimbatore, Tamilnadu, India. Her research interests include Scheduling, Graph Theory, Optimization.

Senthiil Varadharajan received his doctoral degree in Mechanical Engineering with specialization in manufacturing simulation from Bharathiar University, Coimbatore, Tamilnadu, India. He got his Master Degree in Production Engineering from PSG College of Technology in 1985 and Bachelor's Degree in Mechanical Engineering from PSG College of Technology, Coimbatore, Tamilnadu, India in 1983. Presently, he is working as Professor in the department of Production Engineering at St. Peter's University, Chennai, Tamilnadu, India. His research interests include Optimization, Image Grabbing and Analysis, Virtual Simulation.