THE ABURAS HEURISTIC: A MODIFIED HEURISTIC FOR SCHEDULING UNRELATED PARALLEL MACHINE PROBLEMS

H.M. Aburas

Department of Industrial Engineering
Kin Abdulaziz University, Saudi Arabia
haburas@kau.edu.sa

ABSTRACT

Scheduling problems form an important class of decision-making problems where two types of decision have to be taken: allocation and sequencing. Scheduling is primarily considered with resource allocation; sequencing, however, is concerned with the order of jobs to be performed on the allocated resource (Sipper and Bulfin [8]). This paper proposes the use of a new heuristic called ABURAS, which is designed to minimize the makespan of an unrelated parallel machine scheduling problem. To evaluate the performance of the suggested heuristic, a real scheduling problem involving roof truss manufacturing at a major housing construction company is investigated. The ABURAS heuristic shows superiority in terms of reduced makespan over originally developed heuristics.

OPSOMMING

Skeduleringsvraagtukke vorm 'n belangrike versameling van besluitvormings-probleme wat gekenmerk word deur toedeling en volgorde. Skedules hou verband met die toewysing van hulpbronne, terwyl sekwensie verband hou met volgorde. ’n Nuwe sogenaamde ABURAS-skeduleringsheuristiek word voorgehou vir praktyktoepassing. Die toepassingsprosedure word beskryf, en die heuristiek word toegepas by 'n onderneming wat dakkappe vervaardig. Die praktyktoepassing lewer bemoedigende resultate.

1. INTRODUCTION

Scheduling is important both in modern manufacturing and in service industries, where it can have a major impact on the productivity of any process. Common objectives in scheduling problems are to minimize the makespan (duration) of production/service, or to maximize total profit for a given set of customer demands. Many scheduling problems are NP-hard (Non-deterministic Polynomial-time hard), and finding efficient ways to solve larger scheduling problems is a promising area of research (Ignizo & Cavalier [3]).

Scheduling is defined as "the process of organizing, choosing and timing resource usage to carry out all the activities necessary to produce the desired output at the desired time, while satisfying a large number of time and relationship constraints amongst the activities and the resource" (Morton & Pentico [4]). Therefore, it is obvious that scheduling is concerned with allocating resources to particular tasks. A resource may be a processor, a medical doctor, or a machine, and a task may be a computer program, a patient, or a job. Originally, scheduling and sequencing algorithms were applied to manufacturing environments; hence, in this paper, resources are called 'machines', and tasks are called 'jobs'. As an important class of problems in the field of Operations Research, machine scheduling problems are divided into the following sub-classes: single machines, identical, uniform, or unrelated parallel machines, flow shop, job shop, and open shop. In addition, hybrid systems or environments might be formed by combining two or more of the previously mentioned subclasses (Rabadi [5]).

The parallel machine scheduling environment can be divided into three sub-environments, depending on whether the machines are identical, uniform, or unrelated. When the machines are identical, the processing time of a certain job is the same on all machines. When the machines are uniform, the processing time varies in a simple fashion according to the assigned machine speed factor. However, when the machines are unrelated, the processing time of any specific job varies from one machine to another in a completely random fashion.

2. THE PROPOSED ABURAS HEURISTIC FOR UNRELATED PARALLEL MACHINE SCHEDULING PROBLEMS

This section describes the approach adopted to find the optimal completion time for the unrelated parallel machine scheduling problem before applying it to a real-world problem. The proposed heuristic is then compared with the latest developed heuristic in the field of unrelated parallel machine scheduling problems (Salem [6]).

The rationale is to consider all job assignment alternatives – i.e., assigning n jobs to m machines and selecting the optimal option. The proposed heuristic employs the following three steps or phases: initial assignment, job exchange heuristic machines, and resequencing. Before navigating through the steps of the ABURAS heuristic, a list of all notations used and their definitions is presented. For simplicity and ease of use, the same notations originally suggested by Salem [6] are used.

2.1 Notations and their definitions

m	number of machines
Mj	the set of machines that can process job j
mj	number of machines that belong to M_j
h, i, j	indices of corresponding jobs
k, L	indices of corresponding machines
n	number of jobs
Nk	the set of jobs that can be processed on machine k
$N_{(k)}$	the set of jobs assigned to machine k
nk	number of jobs that belong to N_k
P_j	processing time of job j
P_{jk}	processing time of job j on machine k
S_{jk}	setup time to process job j on machine k
$Cpartial_{(k)}$	the partial makespan to date on machine k
$C^*_{partial}$	the maximum partial makespan
C_{max}	makespan that is equivalent to the completion time of the last job scheduled

2.2 Description of the proposed heuristic

As mentioned earlier, the heuristic employs three major steps or phases. A brief description of each phase is illustrated next (using pseudo code).

2.2.1 Job Initial Assignment (JIA)

For each available job, $j = 1,. . . , n$

For all eligible machines, M_j
Compute the processing time plus the setup time, C_{jk}
If there is one eligible machine ($M_j = 1$)
Assign the job to it before updating its associated up-to-date makespan,
$C_{partial\ (k)}$
Else (i.e., more than one machine is eligible, $M_j > 1$)
Assign the job to the machine with the minimum up-to-date resulting makespan, if it exists
Establish the new up-to-date makespan of the assigned machine only.
End
End

2.2.2 Job Exchanging Among Machines (JAM)

In this phase of the heuristic, both a modified One-to-Zero exchange technique and a modified One-to-One exchange technique are applied. The first exchange technique starts with the initial job schedule or assignment obtained from the heuristic's first phase. However, the later exchange technique starts with the final schedule resulting from the One-to-Zero exchange technique. Within each applied exchange technique, resulting schedules are evaluated, and if there is no improvement, it stops. Otherwise,

it takes the resulting improved schedule (with minimum makespan) and starts again until no further improvement is possible.

(A) Modified One-to-Zero Exchange Technique

This technique can be described as follows:

1. Select any machine k for which $C_{partial\ (k)} = C^*_{partial}$
2. Search for job j, where $j \in N_{(k)}$ and $j \in N_L$, such that $C_{partial(L)} + P_{jl} + S_{jL}$ is $< C_{partial(k)}$.
3. If no such job j is found, then $C_{partial}$ is the final solution, otherwise; job j is assigned to machine L.
4. Update $N_{(L)} = N_{(L)} \cup \{j\}$, $C_{partial(L)} = C_{partial\ (L)} + P_{jL} + S_{jL}$ and $N_{(k)} = N_{(k)} - \{j\}$.

Here, the modification is attributed to the fact that, whenever a job exchange between machines is suggested, the selected job will be tested in all possible positions within the sequence – i.e., as the first job in the new sequence, the last, and in between each pair of jobs. The entire procedure is repeated until no further reduction in the maximum completion time is possible. In other words, the concept of a permutation search heuristic is applied.

(B) Modified One-to-One Exchange Technique

1. Select any machine k such that $C_{partial\ (k)} = C^*_{partial}$
2. Search for jobs h and i where $h \in N_{(k)}$ and $i \in N_{(L)}$ where $h \in N_L$ and $i \in N_k$ $(k \neq L)$, for which $C_{partial\ (k)} - P_{hk} - S_{hk} + P_{ik} + Si_k < C^*_{partial}$ and $C_{partial\ (L)} - P_{ik} - S_{ik} + P_{hk} + S_{hk} < C^*_{partial}$.
3. If jobs j and i and machine L cannot be found, then $C_{*partial}$ is the final solution. Otherwise, jobs h and i are interchanged (h is rescheduled on machine L, and i is rescheduled on machine k). Similarly, both will be inserted in the best place between the existing jobs, at the beginning or at the end of the sequence, to get the best $C_{partial\ (k)}$ and $C_{partial\ (L)}$.
4. Update $N_{(k)}$, $N_{(L)}$, $C_{partial\ (k)}$ and $C_{partial\ (L)}$.

The entire procedure is repeated until no further reduction in the maximum completion time is possible – i.e., employing the permutation search heuristic (Rabadi [5]).

2.2.3 Resequencings (RSQ)

This step is suggested after assigning all jobs to available machines. In this phase, each machine sequence is considered separately; hence, no job exchange among machines is performed. For each resultant job sequence, jobs are removed from the pre-finally-sequenced jobs, one at a time. Each time a job is removed, its successor is recorded. The removed job will then be reassigned to all possible places within the other jobs. Once the sequence is evaluated, the successor-recorded job is also evaluated in all possible locations – and so on, until no further improvement is observed.

Assume for illustration purposes that jobs 2-5-3-4 are assigned to machine k in the order given. This phase entails that jobs are removed one at a time, and for each removed job, its successor job is recorded. Therefore job 2 is removed and 5 will be recorded as a successor. Now, job 2 is to be placed in all possible locations within the sequence. Suppose that it is found that the best location (i.e., gives minimum makespan) is after job 5; the new sequence, therefore, is 5-2-3-4. Next, the previously recorded job, job 5, is removed and tried in all possible locations within the newly resulting sequence. According to this sequence job 2 is a successor, so it will be considered once job 5 is properly sequenced, and the same procedure is repeated until no further improvement or non-distinct sequences are observed.

3. ILLUSTRATION

3.1 Problem background

The illustrative case study was originally tested for recent developed heuristics for unrelated parallel machine scheduling problems (Salem [6]). It was initially motivated by MiTek Corporation, an industry leader in developing software for the design of trusses for residential roof construction and the manufacture of truss plates for securing truss joints. (The complete description of this scheduling problem can be found in Salem [6] and Salem and Armacost [7].) According to this scheduling problem, each truss is characterized by some number of joints and components, and the design accurately specifies the joint location and the cut angles on all of the components. Each truss requires a setup time and a processing time. A setup time is used for placing the components at the appropriate respective joint location, while processing involves placing the components on the stands and setting the truss plates with a power crimper. Furthermore, truss manufacturing operations may be restricted to one machine type or another depending on size, capacity, and other special considerations. Previously, manufacturers sequenced and scheduled individual trusses for manufacture using a heuristic that depends heavily on the scheduling expertise and experience of a shop manager/foreman. The corporation under study in this illustration uses two machines, MARK V and MARK VIII. The second machine is similar to the first one, but less efficient in terms of both setup times and processing times. The sequencing problem is further complicated by which machines may be used to process which trusses. In other words, this application includes machine eligibility restrictions – i.e., in an adopted application, it was assumed that the first three trusses (R1, R1A, R1B) could be only manufactured on machine MARK V, and the next three trusses (R1C, RG1, R2A) could be manufactured only on machine MARK VIII, while the remaining four trusses (GE1, GE2, R11, R1AA) could be manufactured on either machine MARK V or MARK VIII (see Table 1).

Truss	Total Processing Time on MARK V (min)	Total Processing Time on MARK VIII (min)
R1	75.6	81
R1A	23.52	25.2
R1B	21.84	23.4
R1C	21.84	23.4
RG1	23.52	25.2
R2A	28.56	30.6
GE1	40.32	43.2
GE2	31.92	34.2
R11	25.2	27
R1AA	13.44	14.4

Table 1: Processing time per machine

Similarly, the setup time vary based on the selected machine. The various setup times for both machines are shown in Tables 2 and 3.

Truss	R1	R1A	R1B	R1C	RG1	R2A	GE1	GE2	R11	R1AA
R1	30	27	24	30	24	81	75	105	33	99
R1A	15	39	24	27	16	81	78	105	24	99
R1B	24	27	36	24	21	81	75	105	33	99
R1C	21	30	21	36	27	69	75	105	33	81
RG1	21	24	24	30	33	78	78	105	30	90
R2A	21	33	21	21	21	74	60	48	18	21
GE1	18	24	18	33	24	66	84	99	24	75
GE2	24	33	24	24	18	42	75	108	21	63
R11	24	24	27	33	24	69	78	93	36	78
R1AA	21	24	30	24	18	6	75	63	33	102

Table 2: Setup times on machine MARK V (in minutes)

Truss	R1	R1A	R1B	R1C	RG1	R2A	GE1	GE2	R11	R1AA
R1	36	32.4	28.8	36	28.8	97.2	90	126	39.6	118.8
R1A	18	46.8	28.8	32.4	21.6	97.2	93.6	126	28.8	118.8
R1B	28.8	32.4	43.2	28.8	25.2	97.2	90	126	39.6	118.8
R1C	25.2	36	25.2	43.2	32.4	82.8	90	126	39.6	97.2
RG1	25.2	28.8	28.8	36	39.6	93.6	93.6	126	36	108
R2A	25.2	39.6	25.2	25.2	25.2	100.8	72	57.6	21.6	25.2
GE1	21.6	28.6	21.6	39.6	28.2	79.2	100.8	118.8	28.8	90
GE2	28.8	39.6	28.8	28.8	21.6	50.4	90	129.6	25.2	75.6
R11	28.8	28.8	32.4	39.6	28.8	82.8	93.6	111.6	43.2	93.6
R1AA	25.2	28.8	36	28.8	21.6	7.2	90	75.6	39.6	122.4

Table 3: Setup times on machine MARK V (in minutes)

In Tables 2 and 3 the reader should notice that diagonal elements represent the setup time of the first sequenced truss type to be manufactured. In addition, shaded cells in the Tables represent trusses that are not eligible to be processed on the specific machine.

3.2 Computational results

To evaluate the proposed heuristic and compare it to the latest developed heuristic in this field of research, a computer program was developed to solve the scheduling problem described in 3.1. The developed software used Visual Basic 6 (Brown [2]) owing to its simplicity and popularity among programmers, especially in academia. Microsoft Excel was used as an interface for the same reasons. Table 4 shows the sequences obtained from the ABURAS and other previously developed heuristics, work completion time per machine, and the overall makespan.

Heuristic	**Description**	**Machine type**	
		MARK V	**MARK VIII**
Aburas	**Sequence of Trusses**	**Trusses: R11, R1A, R1, GE1, and R1B**	**Trusses: R1C, R1AA, R2A, GE2, and RG1**
	Work Completion Time	354.48 minutes	354.60 minutes
	Makespan	354.60 minutes	
Other	**Sequence of Trusses**	**Trusses: R11, R1B, R1A, R1, and GE1**	**Trusses: RG1, R1C, R1AA, R2A, and GE2**
	Work Completion Time	366.48 minutes	365.40 minutes
	Makespan	366.48 minutes	

Table 4: Performance comparison

It is obvious that the proposed heuristic results in a makespan reduction of about 5%, compared with previously developed heuristics. Furthermore, the proposed heuristic maintains an even work load distribution between the two machines.

4. CONCLUSION

The primary objective of this paper is to propose a new heuristic for the unrelated parallel machine class of scheduling problems. A previous heuristic for four different unrelated parallel machine problems was developed based on two important parameters: the job selection parameter and the makespan estimation parameter – α and β respectively (Salem [6] and Salem & Armacost [7]). However, both parameters are subjective to a large extent, and may be confusing in practice.

The proposed ABURAS heuristic provides a simplified solution: it completely excludes the two subjective parameters mentioned above, as well as many other unnecessary parameters. Finally, the proposed heuristic is a combination of several different heuristics.

To evaluate its performance, the proposed ABURAS heuristic was applied to a real world problem, reported and solved by Salem's suggested heuristics (Salem [6]). It has been found that the ABURAS heuristic results in a reduced makespan of about 5%, as well as better machine utilization. In addition, it reduces the number of parameters. Furthermore, a computer program is developed with Microsoft Visual Basic 6, using Microsoft Excel as an interface (Figures 1 through 3).

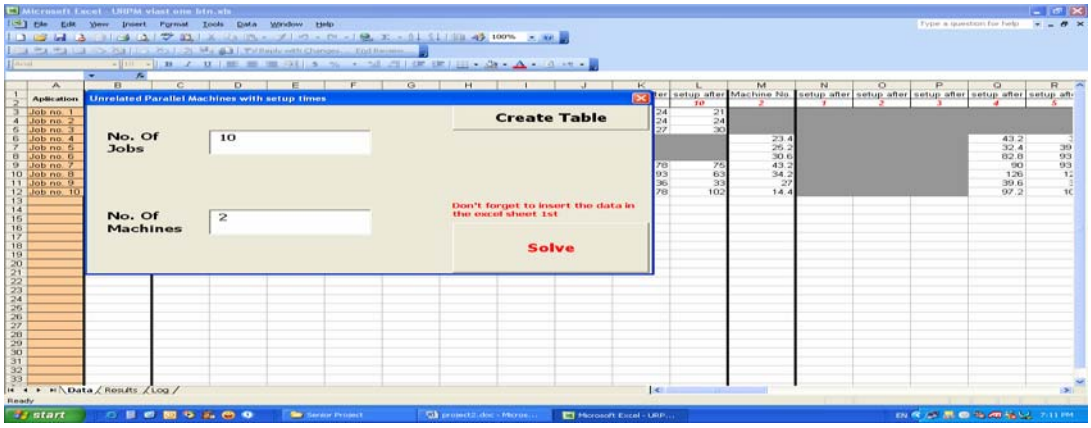

Figure 1: Defining number of jobs and machines (snapshot)

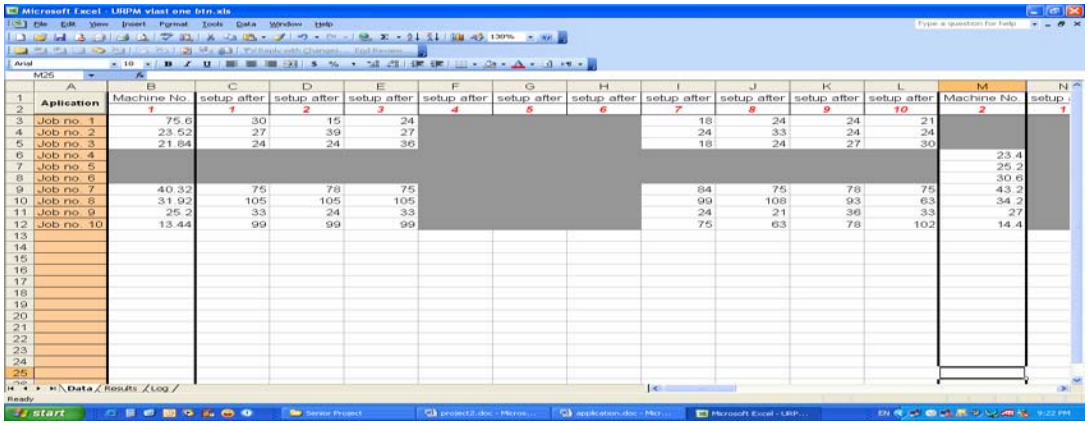

Figure 2: The data entry screen of the developed software (snapshot)

For future research in this area, researchers are advised to investigate further the following unrelated parallel machine scheduling issues:

- Seek further improvement, in terms of minimum makespan, through finding a better sequence-searching mechanism.
- Investigate the robustness and sensitivity of the proposed heuristic to the different input parameters, and also study their interaction effects.

- Build an economic model to compare the performance based on cost rather than completion time or makespan.
- Incorporate due date and release date in the scheduling problems.

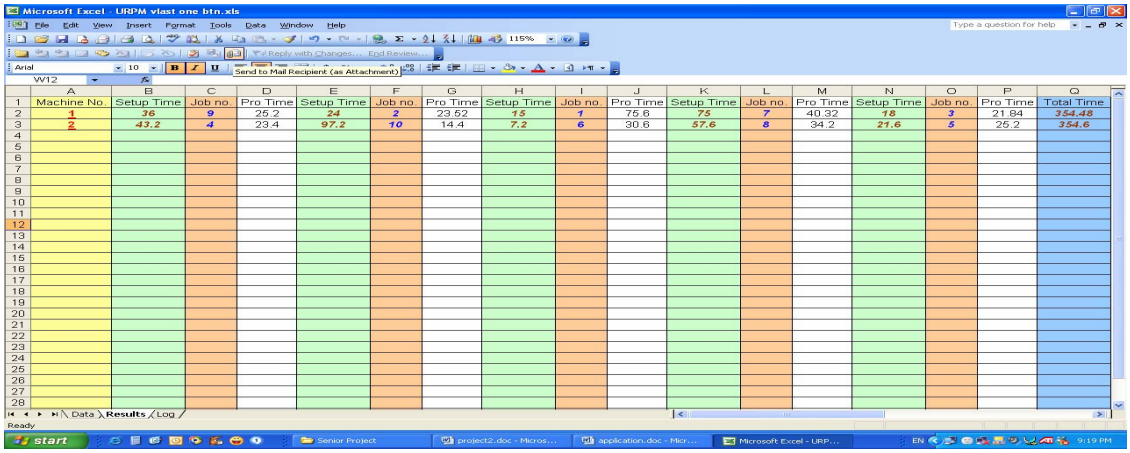

Figure 3: The software output: Job sequence per machine and its makespan (snapshot)

5. REFERENCES

[1] **Baker, K.R.** 1974. *Introduction to sequencing and scheduling*, John Wiley and Sons, Inc., New York.

[2] **Brown, S.** 1999. *Visual Basic 6 complete*, Sybex Inc.

[3] **Ignizo, J.P. and Cavalier, T.M.** 1994. *Linear programming*, Prentice Hall Inc.

[4] **Morton, T.E. and Pentico**, **D.W.** *Heuristic scheduling systems with applications to production systems and project management*, John Wiley and Sons, Inc., New York.

[5] **Rabadi, G.** 1999. *Minimizing the total earliness and tardiness for single-machine scheduling problem with common due date and sequence-dependent setup times*, Doctoral dissertation, Department of Industrial Engineering and Management Systems, University of Central Florida, USA.

[6] **Salem, A.** 1999. *Unrelated parallel machine scheduling with sequence-dependent setup times and machine eligibility restrictions for minimizing the makespan*, Doctoral dissertation, Department of Industrial Engineering and Management Systems, University of Central Florida, USA.

[7] **Salem, A. and Armacost, R.L.** 2002. Unrelated machine scheduling with machine eligibility restrictions, *Engineering Journal of the University of Qatar*, Vol. 15, pp. 193-210.

[8] **Sipper, D. and Bulfin, R., Jr.** 1998. *Production planning, control, and integration*, McGraw-Hill.