

A Decomposition Heuristic for Planning and Scheduling of Jobs on Unrelated Parallel Machines

Christoph Habla, Lars Mönch

University of Hagen, Department of Mathematics and Computer Science, 58097 Hagen, Germany
{Christoph.Habla, Lars.Moench}@FernUni-Hagen.de

Michele E. Pfund, John W. Fowler

Arizona State University, Department of Industrial Engineering, Tempe, AZ 85287-5906, USA
{Michele.Pfund, John.Fowler}@asu.edu

In this paper, we study a planning and scheduling problem for unrelated parallel machines. There are n jobs that have to be assigned and sequenced on m unrelated parallel machines. Each job has a weight that represents the priority of the corresponding customer order, a given due date, and a release date. An Automated Guided Vehicle is used to transport at maximum $Load_{max}$ jobs into a storage space in front of the machines in a given period of time. We consider T consecutive periods of time, and are interested in minimizing the total weighted tardiness (TWT) of the jobs across the T periods. To solve the problem, we present a mixed integer program (MIP) and a heuristic decomposition methodology. These methodologies are tested using stochastically generated test instances and compared. Results indicate that the decomposition approach performs comparably to the MIP while having reasonable solution times.

Keywords: Planning and Scheduling, Parallel Machines, Decomposition, Genetic Algorithms.

1. Introduction

Scheduling jobs in manufacturing is still challenging due to the NP-hardness of many scheduling problems, a lack of data, machine breakdowns, and multiple criteria [5,10]. Therefore, the development of efficient planning and control strategies is highly desirable for complex manufacturing systems. In the course of the development of new planning and control algorithms, the researchers and developers have to take into account the new opportunities of advanced software and hardware technologies.

In this paper, we model a planning and scheduling problem for unrelated parallel machines in a Printed Wiring Board (PWB) manufacturing environment. The process of PWB consists of multiple stages of production. Each board must pass through a pre-assigned sequence of manufacturing steps [16]. Scheduling jobs on parallel machines is a building block for any enterprise-wide scheduling approach for a PWB line. In the motivating PWB line, Automated Guided Vehicles (AGV's) are used to transport lots from one stage to another one.

We propose a multi-period formulation and show that the model can be solved very efficiently by means of genetic algorithms and dispatching rules. The multi-period formulation is a generalization of previous work done by Pfund *et al.* [11,16].

The paper is organized as follows. We describe the problem, related literature, and a MIP formulation in Section 2. Section 3 presents a decomposition approach that uses dispatching rules and genetic algorithms within the different phases of the decomposition. We show results of computational experiments in Section 4.

2. Problem and model

2.1. Problem Description

We consider a planning and scheduling problem for jobs on unrelated parallel machines in a PWB manufacturing environment. We assume that we know in advance which of the n jobs can be processed on the machines within a given horizon T . An AGV is used to transport at maximum $Load_{max}$

jobs each P hours to the machines. For simplicity reasons we assume that T is divided into t_{max} periods of size P . Therefore, the relation $T = Pt_{max}$ holds. We are interested in minimizing the total weighted tardiness (TWT) of the n jobs. The quantity TWT is defined as follows:

$$TWT = \sum_{j=1}^n w_j T_j = \sum_{j=1}^n w_j (c_j - d_j)^+ \quad (1)$$

Here we use for abbreviation the notation $x^+ := \max(x, 0)$. The customer specific priority of job j is denoted by w_j . The notation c_j is used for the completion time of job j whereas d_j denotes the due date of job j . The problem can be represented as

$$R_m / r_j, Load_{max} / TWT \quad (2)$$

applying the $(\alpha / \beta / \gamma)$ classification scheme for scheduling problems (cf. [4,3]). We use the notation R_m for unrelated parallel machines, i.e., the speed of the machines can be different and depends on the jobs (cf. [13] for the different types of parallel machine environments). We consider a dynamic planning and scheduling problem, i.e., each job has a ready time r_j . Throughout the rest of the paper, the ready time of a job is given by the period where the job can be processed for the first time.

The considered problem is NP-hard because the single machine problem $1 // TWT$ [6] is also NP-hard. Therefore, efficient heuristics are needed to solve reasonable sized problems.

We finally summarize the decisions that have to be made within a period t where $1 \leq t \leq t_{max}$:

- **Load decisions** for the guided vehicles: selection of maximum $Load_{max}$ jobs from the set of non-planned jobs,
- **Assignment decisions:** assignment of jobs to a specific machine,
- **Sequencing decisions:** after the assignment of jobs to machines the sequence of the jobs on each machine has to be determined. This can be done by determining the position of the jobs on a single machine.

We call the problem to be researched a planning and scheduling problem because we consider future time periods. This leads to a planning problem. When we take into account only a single period, we have to deal with a pure scheduling problem.

2.2. Related Literature

Unrelated parallel machine scheduling problems are widely discussed in the scheduling literature. We refer, for example, to the recent survey paper by Pfund, Fowler, and Gupta [12].

An efficient branch & bound algorithm that is based on several dominance rules is described in [7]. However, only a single-period problem is presented. A related problem for unrelated parallel machines in a PWB environment is solved in [16,11] wherein a MIP formulation is given and the MIP is solved by Lagrange relaxation. However, only a single period is considered.

Decomposition techniques are widely used to solve complex scheduling problems (cf. [10] for different decomposition techniques). Different multi-phase decomposition approaches for scheduling jobs on parallel batch machines are discussed in [9]. The authors consider the phase batch formation, assignment of batches to single machines, and finally a sequencing phase for each single machine. Genetic algorithms are used for the assignment phase and dispatching rules are used for the remaining phases. We solve the planning and scheduling problem in this paper by a similar decomposition approach.

2.3. Mixed Integer Programming Formulation

In this section, we present a mixed integer programming formulation of our planning and scheduling problem. We introduce first the necessary index sets. The index sets can be derived from the load, assignment, and sequencing decisions. We consider therefore the following index sets:

$j \in J$:	jobs,	$1, \dots, j_{max}$,
$i \in I$:	machines,	$1, \dots, i_{max}$,
$t \in T$:	periods,	$1, \dots, t_{max}$,
$k \in K^{(i)}$:	positions,	$1, \dots, k_{max}^{(i)}$,
$k_{max}^{(i)}$:	maximum number of jobs per period on machine i .	

We consider the set $T^{(j)} := \{t / r_j \leq t \leq t_{max}\}$ for each $j \in J$. Furthermore, we introduce the index set $J^{(t)} := \{j / j \in J \wedge r_j \leq t\}$ for each $t \in T$.

The model contains the following parameters:

P :	duration of a period,
$p_{j,i}$:	processing time of job j on machine i ,
a_i :	time of the first availability of machine i in period $t=1$.

The already defined quantities d_j , w_j , and $Load_{max}$ are further model parameters. We use the following decision variables within the model. We define the binary variable

$$x_{j,i,t,k} := \begin{cases} 1, & \text{when job } j \text{ is on machine } i \text{ in period } t \text{ at position } k \\ 0, & \text{otherwise} \end{cases} \quad (3)$$

for all $j \in J$, $i \in I$, $t \in T^{(j)}$, and $k \in K^{(i)}$. The variable $o_{i,t}$ measures for all $i \in I$ and $t \in T$ the overload on machine i in period t , i.e., the amount of time that is necessary to complete the last job on machine i in period t after the end of period t . The decision variable $c_{i,t,k}$ models the completion time of the job at position k in period t on machine i for all $i \in I$, $t \in T$, $k \in K^{(i)}$. The tardiness of job j is given by $T_j := (c_j - d_j)^+$ for all jobs $j \in J$. Note that the quantity $x_{j,i,t,k}$ is a binary variable whereas the remaining variables are real and non negative.

The objective function TWT has to be minimized. The optimization problem can be formulated as follows:

$$(P) \quad \min \sum_{j \in J} w_j T_j \quad (4)$$

subject to the following constraints

$$\sum_{i \in I} \sum_{t \in T^{(j)}} \sum_{k \in K^{(i)}} x_{j,i,t,k} = 1, \quad \forall j \in J, \quad (5)$$

$$\sum_{j \in J^{(t)}} x_{j,i,t,k} \leq 1, \quad \forall i \in I, t \in T, k \in K^{(i)}, \quad (6)$$

$$c_{i,t,k} = a_i + \sum_{n=1}^k \sum_{j \in J^{(t)}} x_{j,i,t=n} p_{j,i}, \quad \forall i \in I, t=1, k \in K^{(i)}, \quad (7)$$

$$c_{i,t,k} = (t-1)P + o_{i,t-1} + \sum_{n=1}^k \sum_{j \in J^{(t)}} x_{j,i,t=n} p_{j,i}, \quad \forall i \in I, t=2, \dots, t_{\max}, k \in K^{(i)}, \quad (8)$$

$$c_{i,t,k} - p_{j,i} \leq tP + t_{\max}P(1 - x_{j,i,t,k}), \quad \forall j \in J, i \in I, t \in T^{(j)}, k \in K^{(i)}, \quad (9)$$

$$o_{i,t} \geq c_{i,t,k} - tP, \quad \forall i \in I, t \in T, k = k_{\max}^{(i)} \quad (10)$$

$$T_j \geq c_{i,t,k} - d_j - (t_{\max} + 1)P(1 - x_{j,i,t,k}), \quad \forall j \in J, i \in I, t \in T^{(j)}, k \in K^{(i)}, \quad (11)$$

$$\sum_{j \in J^{(t)}} \sum_{i \in I} \sum_{k \in K^{(i)}} x_{j,i,t,k} \leq Load_{\max}, \quad \forall t \in T. \quad (12)$$

Constraint (5) assigns to each job a machine i , a period t , and a position k . Constraint (6) ensures that each position for each machine and period can be occupied only by at most one job. Constraints (7) and (8) calculate the completion time of the job at position k on machine i in period t . Constraint (9) ensures that the last job in period t on machine i starts with processing before period t is over. Constraint (10) determines the overtime in period t on machine i . Inequality (11) allows calculating the tardiness of job j . Constraint (12) makes sure that the maximum transportation quantity (capacity) of jobs for the AGV for each period is not violated.

3. Decomposition approach

Since the problem is NP-hard we develop a heuristic approach for the solution of large problem sizes. We suggest a three-phase decomposition algorithm:

- assignment of jobs to periods (load phase),
- assignment of jobs to the parallel machines for each single period in the second phase (assignment phase),
- solution of the resulting single machine sequencing problems of the second phase in the third phase (sequencing phase).

3.1. Load phase

The aim of the load phase consists of assigning $Load_{\max}$ jobs to single periods. Because we are interested in minimizing TWT we use the Apparent Tardiness Cost (ATC) dispatching rule [14] to select appropriate jobs for each period. We calculate for each job j the ATC index given by

$$I_j(t) := \frac{w_j}{p_{j,i}} \exp\left(-\frac{(d_j - p_{j,i} - t)^+}{k\bar{p}}\right), \quad (13)$$

where we denote by k a look-ahead parameter that is varied between 0.1 and 6.5. We use the notation \bar{p} for average processing time of the remaining jobs. The time t is the time when any of the available machines becomes free in a simple deterministic forward simulation. At this time t , all available jobs that are ready in the period are ordered in descending order of their ATC index and the job with the highest ATC index is assigned to the available machine. Note that we have to perform the forward simulation to obtain a dynamic version of the ATC rule by determining appropri-

ate values for t . We consider the next period after selecting $Load_{max}$ jobs. Note that it is crucial for the performance of the ATC rule to select appropriate k values [13]. In order to find appropriate values we run the pure dispatching based approach described in Section 3.4 for a large number of k from an equidistant grid on $[0,1,6.5]$.

3.2. Assignment phase

We use a genetic algorithm (GA) in order to assign the jobs in each single period to the parallel machines, evaluate each of the assignments by aggregating the total weighted tardiness of sequences (obtained on each machine by using sequencing heuristics for the single machine case) on all machines. The GA converges towards assignments that give good solutions. GA's are widely used for hard combinatorial optimization problems (cf. [2] for applications in manufacturing).

We used a job-based chromosome representation in the GA. A solution of the problem to assign jobs to machines is an array whose length is equal to the number of jobs. The s -th element of the array represents the machine that is used to process job s . Each chromosome is randomly initialized by generating a random number from $\{1, \dots, m\}$ for each single gene. A standard one-point crossover operator [8] is used for performing the crossover operations. According to a user-specific probability, a gene of a chromosome is randomly changed to a different machine number for mutation purposes. Each chromosome is evaluated with the sum of the total weighted tardiness value of the jobs in a given period and a second term that represents the overload on the machines in the period. After the jobs are assigned to machines, each machine is sequenced and the sum of the total weighted tardiness values of all machines and the overload for each machine and period is used as the fitness function of the GA. A steady state GA with overlapping populations is used (cf. [8]). The GA is controlled by a prescribed number of generations. The parameter setting of the GA is done by computational experiments.

3.3. Sequencing phase

For sequencing the jobs on each single machine within each single period we use again the ATC rule. We apply again deterministic forward simulation as described for the load phase to update the time t in the ATC index in a correct way.

3.4. Pure dispatching rule based approach

A simple dispatching rule based scheme can be constructed using the ATC rule as described in Section 3.1. After the assignment of at maximum $Load_{max}$ jobs to each period we use a list based scheduling approach to assign jobs to parallel machines. Note that we simultaneously determine a sequence for each single machine. As already described we have to iterate over a large number of k values to find values that lead to small TWT values. We denote the pure dispatching rule based approach by ATCH. The suggested decomposition approach is denoted by GA-ATC.

4. Computational results

4.1. Design of experiments

We use the experimental design shown in Table 1 for our computational experiments. The applied test data were generated according to an extension of the test data description given in [1] to the case of parallel machines and multiple periods. The test instances depend on the number of parallel machines, the number of jobs in each period on a machine, and the ready time and due setting mechanism. Totally, we solve 135 test instances.

All experiments were performed on a Pentium IV, 3.4 GHz PC. We used the object-oriented framework GaLib (cf. [15]) in order to implement the suggested GA in an efficient way. The computing time was less than 20 seconds for GA-ATC and less than 0.3 seconds for ATCH for all test instances analyzed in Section 4.2. That is both fast with respect to the planning horizon of several hours. We take six independent replications of the GA runs for one test instance to obtain stochastically significant results.

Table 1. Factorial Design

Factor	Level	Count
Capacity of the AGV $Load_{max}$	{18,24,30} for {3,4,5} machines	1
Number of Machines i_{max}	{3,4,5}	3
Number of Periods t_{max}	6	1
Average Number of Jobs per Period and Machine	4 $j_{max} = 72,96,120$	1
Length of a Period P	240	1
Efficiency v_i of the Machines	0.5 ($i = 1$) 1.0 ($1 < i < i_{max}$) 1.25 ($i = i_{max}$)	1
Processing Time of the Jobs for the Different Products (each of them with equal probability)	30, 45, 60 und 75	1
Ready Time	$r_j \sim 1 + \lfloor t_{max} * \alpha * U(0,1) \rfloor$ $\alpha = 0.25, 0.50, 0.75$	3
Due Date	$d_j \sim (r_j + \lceil t_{max} * \beta * U(0,1) \rceil) * P$ $\beta = 0.25, 0.50, 0.75$	3
Weight of the Jobs	$w_j \sim U(0,1)$	1
Number of Parameter Combinations		27
Number of Independent Problem Instances		5
Total Number of Problem Instances		135

4.2. Results of computational experiments

The results of the experiments are given in Table 2. We show the improvement ratio for the TWT value of the suggested decomposition approach to the TWT value of ATCH. Clearly, GA-ATC outperforms ATCH. As can be seen from the results the algorithm is sensitive to the ready time range that is controlled by α and the due date range that depends on the choice of β . Instead of comparing all test instances individually, the test instances were grouped according to the level of factors. For example, the results for $m=3$ are for all runs with three machines while all other factors have been varied at their different levels.

Table 2. Computational Results for GA-ATC

	ATCH	GA-ATC
$m = 3$	1.00	0.96
$m = 4$	1.00	0.94
$m = 5$	1.00	0.92
$\alpha = 0.25$	1.00	0.96
$\alpha = 0.50$	1.00	0.96
$\alpha = 0.75$	1.00	0.90
$\beta = 0.25$	1.00	0.97
$\beta = 0.50$	1.00	0.89
$\beta = 0.75$	1.00	0.96
Overall	1.00	0.94

In Table 3, we study the effect of different release and due date setting schemes. Therefore, we consider all possible α and β combinations. The choice of $\beta=0.75$ leads to the largest improvement rates for $\alpha=0.25$ because a larger β value causes a wider spread of due dates. In this situation the GA offers a lot of advantage. Larger values of α in combination with $\beta=0.50$ also lead to large improvements.

Table 3. Effect of Release Date and Due Date Factor Setting

	ATCH	GA-ATC
$\alpha = 0.25, \beta = 0.25$	1.00	0.98
$\alpha = 0.25, \beta = 0.50$	1.00	0.97
$\alpha = 0.25, \beta = 0.75$	1.00	0.92
$\alpha = 0.50, \beta = 0.25$	1.00	0.98
$\alpha = 0.50, \beta = 0.50$	1.00	0.94
$\alpha = 0.50, \beta = 0.75$	1.00	0.97
$\alpha = 0.75, \beta = 0.25$	1.00	0.94
$\alpha = 0.75, \beta = 0.50$	1.00	0.75
$\alpha = 0.75, \beta = 0.75$	1.00	1.00

In Table 4, we compare the solution quality of ATCH and GA-ATC with the solution quality obtained by an ILOG CPLEX implementation of the MIP presented in this paper. The last column shows the improvement rates obtained by the ILOG software compared to ATCH and GA-ATC. We use only small size test instances (24 jobs) for $m \in \{2,3\}$. The ILOG solver runs at maximum for two hours.

Table 4. TWT Values for ATCH, GA-ATC, and ILOG

α	β	M	ATCH	GA-ATC	ILOG -MIP (2 h)	Improvement [%]	
						ATCH	GA-ATC
0.25	0.25	2	606	579	579	4	0
0.25	0.25	3	933	895	898	4	0
0.25	0.50	2	314	280	280	11	0
0.25	0.50	3	179	151	148	17	2
0.50	0.25	2	146	140	139	4	0
0.50	0.25	3	219	204	204	7	0
0.50	0.50	2	84	78	78	7	0
0.50	0.50	3	58	51	52	10	0

We see from Table 4 that GA-ATC is able to find solutions with the same TWT value as obtained by the ILOG solver in almost all cases. But even ATCH performs quite well. Note that it is not possible to solve test instances from Table 1 with a reasonable amount of time by using the ILOG software.

5. Conclusion

In this paper, we presented a problem description for a combined planning and scheduling problem. We developed a mixed-integer programming formulation. Because of the computationally intractable mixed integer programs, we suggested a decomposition heuristic based on hybridized genetic algorithms. Based on stochastically generated test instances it turned out that the decompo-

sition approach outperforms a simple dispatching rule based approach. We found that even the dispatching rule based approach provided near to optimal solutions for small size test instances based on comparison with solutions obtained by the ILOG software with a two hour maximum computation time. There are several directions for future research. First of all it seems to be possible to study different modifications of the genetic algorithm. It seems to be possible to avoid the load phase in the beginning. In this case, first an assignment step for jobs to machines is necessary. Then, in a second step it is required to assign the jobs to periods and sequence them on each single machine. However, a careful investigation is necessary because of the considerably larger search space of the GA in this case. Instead of looking for TWT it seems to be possible to consider other performance measures and to look for different product mix scenarios. Due to the parallel structure of the problem, the second future research direction is given by using column generation techniques where the columns represent a job assignment to one of the parallel machines. Here, appropriate sub problem solution techniques have to be developed. However, carrying out all the details is part of future research.

References

- [1] M. S. Akturk, D. Ozdemir (2001), A New Dominance Rule to Minimize Total Weighted Tardiness with Unequal Release Dates, *European Journal of Operational Research* **135**, 394–412.
- [2] H. Aytug, M. Khouja, F. E. Vergara (2003), Use of Genetic Algorithms to Solve Production and Operations Management Problems: a Review, *International Journal of Production Research* **41**(17), 3955–4009.
- [3] J. Blazewicz, K. H. Ecker, E. Pesch, G. Schmidt, J. Weglarz (2001), *Scheduling Computer and Manufacturing Processes*, 2nd edition, Springer, Berlin.
- [4] R. L. Graham, E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan (1997), Optimization and Approximation in Deterministic Sequencing and Scheduling: a Survey, *Annals of Discrete Mathematics* **5**, 287–326.
- [5] S. Kreipl, M. Pinedo (2004), Planning and Scheduling in Supply Chains: An Overview of Issues in Practice, *Production and Operations Management* **13**(1), 77–92.
- [6] E. L. Lawler (1977), A “Pseudopolynomial” Time Algorithm for Sequencing Jobs to Minimize Total Weighted Tardiness, *Annals of Discrete Mathematics* **1**, 331–342.
- [7] C.-F. Liaw, Y.-K. Lin, C.-F. Cheng, M. Chen (2003), Scheduling Unrelated Parallel Machines to Minimize Total Weighted Tardiness, *Computers & Operations Research* **30**, 1777–1789.
- [8] Z. Michalewicz (1996), *Genetic Algorithms + Data Structures = Evolution Programs*, 3rd edition, Springer, Berlin, Heidelberg, New York.
- [9] L. Mönch, H. Balasubramanian, J. W. Fowler, M. E. Pfund (2005), Heuristic Scheduling of Jobs on Parallel Batch Machines with Incompatible Job Families and Unequal Ready Times, *Computers & Operations Research* **32**, 2731–2750.
- [10] I. M. Ovacik, R. Uzsoy (1997), *Decomposition Methods for Complex Factory Scheduling Problems*, Kluwer Academic Publishers, Boston.
- [11] M. E. Pfund (2002), *Evaluation of Uncertainty on Scheduling Algorithms in Printed Wiring Board Manufacturing*, PhD Dissertation, Arizona State University, Department of Industrial Engineering.
- [12] M. E. Pfund, J. W. Fowler, J. N. D. Gupta (2004), A Survey of Algorithms for Single and Multi-objective Unrelated Parallel-Machine Deterministic Scheduling Problems, *Journal of the Chinese Institute of Industrial Engineers* **21**(3), 230–241.
- [13] M. Pinedo (2002), *Scheduling: Theory, Algorithms, and Systems*, Prentice Hall, Second Edition, Englewood Cliffs, NJ.
- [14] A. Vepsäläinen, T. E. Morton (1987), Priority Rules and Lead Time Estimates for Job Shop Scheduling with Weighted Tardiness Costs, *Management Science* **33**, 1036–1047.
- [15] M. Wall (2007) *GaLib – a C++ Library of Genetic Algorithm Components*, <http://lancet.mit.edu/ga/>.
- [16] L. Yu, H. M. Shih, M. E. Pfund, W. M. Carlyle, J. W. Fowler (2002), Scheduling of Unrelated Parallel Machines: an Application to PWB Manufacturing, *IIE Transactions* **32**, 921–931.