Hybrid Approach for Machine Scheduling Optimization in Custom Furniture Industry

Juan C. Vidal, Manuel Mucientes, Alberto Bugarín, and Manuel Lama
Department of Electronics and Computer Science
University of Santiago de Compostela, E-15782 Spain
{juan.vidal,manuel.mucientes,elbugari,manuel.lama}@usc.es

Reza Sadigh Balay
Martínez Otero, S.L.
Avda. de Pontevedra 97, E-36689 A Estrada, Spain
r.balay@martinezotero.com

Abstract

Machine scheduling is a critical problem in industries where products are custom-designed. The wide range of products, the lack of previous experiences in manufacturing, and the several conflicting criteria used to evaluate the quality of the schedules define a huge search space. Furthermore, production complexity and human influence in each manufacturing step make time estimations difficult to obtain thus reducing accuracy of schedules. The solution described in this paper combines evolutionary computing and neural networks to reduce the impact of (i) the huge search space that the multi-objective optimization must deal with and (ii) the inherent problem of computing the processing times in a domain like custom manufacturing. Our hybrid approach obtains near optimal schedules through the Non-dominated Sorting Genetic Algorithm II (NSGA-II) combined with time estimations based on multilayer perceptron networks.

1. Introduction

Machine scheduling is a difficult problem in industrial environments such as custom furniture industry. This problem can be defined as finding an optimal sequence of operations for a set of resources and constraints. Machine scheduling is classified as NP-hard [9] because it has a combinatorial search space caused by the resources that must be allocated to maximize the utilization of machines and to minimize the time required to complete the scheduling process. Therefore, this problem can take a lot of computing time with exact methods such as branch and bound, dynamic programming or constraint logic programming. For this reason, solutions near the optimal are considered good solutions in this context and non-derivative methods, such as evolutionary algorithms, simulated annealing, tabu search or complex method, are better suited [2].

Machine scheduling problem is characterized by the presence of many conflicting objectives. Therefore, it is natural to look at scheduling as a multi-objective optimization problem that raises the issue about how different objectives should be combined to yield a final solution. Unfortunately, the processing times that constitute the essence of some scheduling process are difficult to obtain in custom manufacturing. The computing of these times depends on several factors such as machine, material and piece characteristics. The latter is the key point because materials, dimensions and shape of custom-designed pieces may take a wide range of different values. Since these times consist of a mixture of numerical simulations, analytical calculations and catalog selections, there is no precise way of calculating the objective functions of the scheduling.

Selection of the multi-objective optimization algorithm is closely related to the particular problem to be solved. This paper deals with a real-world scheduling problem that is close to *job shop scheduling problems* family. However, the scheduling constraints in our problem define a huge search space which may be even huger if the scheduling faced events such as machines breaking down, workers getting sick or new jobs appearing. In our approach we deal with these events through a new scheduling and not through rescheduling. That is, these events will only modify the conditions under which the new scheduling is performed. For this reason, our approach requires to be reliable but also time efficient since time is critical to have a faster response

978-0-7695-3326-1/08 $25.00 © 2008 IEEE
DOI 10.1109/HIS.2008.141

to customer orders. In this context, evolutionary algorithms are known to be a fast and robust solution in optimization problems such as scheduling [6, 4] because they facilitate finding a global optima, and do not get trapped on local optima as gradient methods might do.

The machine scheduling solution described in this paper combines evolutionary computing and neural networks to reduce the impact of *(i)* the huge search space that our multi-objective scheduling process must deal with and *(ii)* the inherent problem of computing the processing times in a domain like custom manufacturing. Our approach is based on the use of a multi-objective evolutionary algorithm where schedule evaluation is based on processing times modeled by neural networks. In essence, we apply the non-dominated sorting genetic algorithm II (NSGA-II) [7] for producing new schedules in each iteration, and a multilayer perceptron classifier [15] to estimate their processing times.

The paper is structured as follows: Section 2 introduces the problem of scheduling in custom wood-based furniture manufacturing industry. Section 3 describes the NSGA-II algorithm that solves the scheduling problem, how the problem has been encoded and which objective functions evaluate the fitness of the solutions, while Section 4 describes the neural network model we defined to compute the processing times of machines. Finally, Section 5 points out the conclusions and some results of the experiments in a real-world production environment.

2. Machine Scheduling Problem in Custom Wood-based Furniture Manufacturing

The time interval for completing all of the operations of a work order is known as throughput time. Its accurate estimation is hard in industries where custom-designed products are dominant, such as the furniture industry. The lack of previous manufacturing experiences and the complexity in the production makes time estimations difficult.

The reduction of throughput time has many benefits: lower inventory, reduced costs, improved product quality (problems in the process can be detected more quickly), faster response to customer orders, and increased flexibility. Much effort is therefore spent to reduce throughput time by improving manufacturing planning, control systems, and developing more sophisticated scheduling procedures [11, 12]. The objective is to define work plans that minimize the resources queuing and maximize resources capacity, constrained to the material availability and product requirements. In this context, the feasibility, time, and cost of the most promising plans is analyzed [10, 14]. Note that the throughput time has many components, including move, queue, setup, and processing times [1, 5, 16]. In this work we will address all these components by improving the machine scheduling task to produce feasible work plans such as depicted in Figure 1.

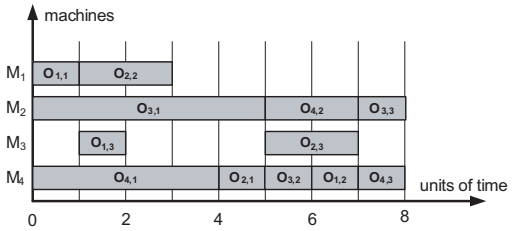


Figure 1. Gantt chart of a schedule. $O_{i,j}$ stands for the j-th operation in job J_i.

The real-world scheduling problem described in this work is close to the *job shop scheduling problem* (JSSP) family. A JSSP P of size $n \times m$ consists of n jobs $J = \{J_1, J_2, \ldots, J_n\}$ to be scheduled on a set of m machines $M = \{M_1, M_2, \ldots, M_m\}$. Each job $i \in J$ consists of a set of n_i operations $O_i = \{O_{i,1}, O_{i,2}, \ldots, O_{i,n_i}\}$ describing the processing order of the operations. Although the way in which we select the manufacturing route is out of the scope of this paper, we just mention that selection is based on a set of rules based on woodworking knowledge. These rules take into account the constructive decisions, joints used to assemble the furniture, and finishing or quality standards. The operation $O_{i,j}$ is the j-th operation of job i to be processed on a specific machine M_k with a processing time $p_{M_k,O_{i,j}}$. Furthermore, each job must be achieved before a due date d_i, and each machine has a *start time* $s_{M_k,O_{i,j}}$ before which no processing can be done on the machine for this operation. The following constraints must also hold although some of them modify the classical constraints of the JSSP:

- Although JSSP constraints that each job is scheduled on each machine only once, our problem needs that each job can be scheduled on each machine more than once. For example edge banding machines can be used for edge banding, trimming or sanding. Therefore, the same machine may be used more than once for a job.
- Two operations of the same job may be processed simultaneously.
- Contrary to the JSSP, jobs do not have to visit every machine in M.
- Each machine can process only one operation at a time.
- No machine can free an operation until it is finished.
- The total number of machines of each type is fixed and greater than one.

- No job can start before the processing of its parts is available such as gluing and assembly jobs.

Another difficulty our scheduling problem must deal with is the processing times estimation, which is a critical piece in each manufacturing step. The estimation of the processing time is one of the most important tasks in the product design life cycle. For example, time estimations are taken into account to redesigning the product if the predicted time is longer than expected. There are many models and techniques for estimating the processing times of a manufacturing step based on the product design [3]. For a detailed design, highly detailed process planning, manufacturing process simulation, or time estimation models can be employed [3, 14]. For a conceptual design, however, less detailed models must depend on a more limited set of critical design information [3]. Both approaches are applied in the furniture industry since the definition of a detailed design is cost and time expensive. Therefore, there are several ways of computing the processing time of a machine depending on the available input variables.

The processing time of an operation for a machine can depend on several input variables, like the dimensions of the product, the material, the speed of the machine for that kind of product, and so on. For example, the calibrating of wood pieces depends on the sanding machine that will perform the operation, apart from the piece variables. Calibrating machinery specifies feed speed, abrasive belt speed, abrasive belt size, working thickness, maximum and minimum workpiece dimensions and so on. Nevertheless, the processing time for a given operation may not always be influenced by all of these variables. Thus, a machine can have several regression functions for a class of operation and product characteristics. Therefore, a good estimation of the throughput time demands an accurate regression function as well as the selection of the appropriate function among all the regression functions of the machine. In this context, processing times estimation is difficult to obtain by standard methods or manufacturing experts and is a source of errors and uncertainty for the scheduling process.

3. NSGA-II Multi-Objective Approach for Machine Scheduling

Generally, we can describe a multi-objective scheduling problem as a multi-objective optimization problem:

$$min\ F(x) = \{f_1(x), f_2(x), ..., f_k(x)\}\ s.\ t.\ x \in S \quad (1)$$

where x is a solution, S is the set of feasible solutions, k is the number of objectives in the problem, $F(x)$ is the image of x in the k-objective space and each $f_i(x)$ for $i = 1, \ldots, k$ represents one objective. In custom furniture manufacturing, like in many other real-world problems, there are conflicting $f_i(x)$ objectives. For example, objectives like *minimize the cost of furniture* and *minimize the completion time* may be in conflict since it is usual that faster machines have a higher recovery cost. So, in contrast to a single-objective optimization problem, there is not one best solution, but several solutions to choose from (non-dominated set of solutions).

1. Initialize population
   (a) Generate schedules
2. for iteration = 1 to *maxIterations*
   (a) Neural networks-based schedules evaluation
   (b) Selection
   (c) Crossover
   (d) Mutation
3. Select best individuals

Figure 2. NSGA-II algorithm structure for machine scheduling

Our approach to this problem uses the multi-objective scheduling optimization through the NSGA-II algorithm [7]. NSGA-II is one of the most efficient multi-objective evolutionary algorithms using elitist approach: its computational complexity is in $O\left(MN^2\right)$, where M is the number of objectives and N the population size. The algorithm structure is described in Figure 2. NSGA-II has a fitness assignment scheme that consists in sorting the population in different fronts using the non-domination order relation. Therefore, it has two objectives: *(i)* to find a set of non-dominated solutions as close as possible to the Pareto-optimal front in each iteration but *(ii)* maintaining the set of solutions as diverse as possible covering or nearby the Pareto-optimal front. Each iteration returns a new population that combines the current population and its offspring generated with the crossover and mutation. Finally, the best individuals in terms of non-dominance and diversity are chosen.

3.1. Problem Encoding

Choosing a good representation is a vital component for solving search problems. In this paper we selected a direct chromosome representation so no transformation is needed from the chromosome to the schedule. Specifically, we represent chromosomes through the *Parallel Job Encoding* [13]. Figure 3 depicts this encoding for a 4 jobs and 4 machines scheduling problem. Each row of the matrix is an ordered sequence of operations $O_{i,j}$, $i = 1, \ldots, 4$, $j = 1, \ldots, 3$. Each element in a row contains two terms: *(i)* the machine M_k, $k = 1, \ldots, 4$, which performs the opera-

tion and *(ii)* the starting time $t_{M_k,O_{i,j}}$ of operation $O_{i,j}$ in machine M_k. For example, the first operation in job J_2 is performed on machine M_4 at time 4.

	O_1	O_2	O_3
J_1	$(M_1, 0)$	$(M_4, 6)$	$(M_3, 1)$
J_2	$(M_4, 4)$	$(M_1, 1)$	$(M_3, 5)$
J_3	$(M_2, 0)$	$(M_4, 5)$	$(M_2, 7)$
J_4	$(M_4, 0)$	$(M_2, 5)$	$(M_4, 7)$

Figure 3. Chromosome representation of the 4×4 schedule example

This encoding directly produces a feasible solution. It contains the machines that will execute each operation and at which time. Time is set to zero when the machine cannot perform the operation. Therefore, a schedule is easily created knowing the operations routing and the processing times. Figure 4 complements the 4×4 scheduling example and provides *(i)* the due date of each job J_i, *(ii)* the operations order in the job and *(iii)* the time (the sum of the setup and processing time) of each operation $O_{i,j}$ on each machine of the production plant. For example, job J_2 has a due date of 7 time units and defines the $O_{2,2}, O_{2,1}, O_{2,3}$ operations order. The Gantt chart depicted in Figure 1 represents the schedule that can be inferred from the chromosome depicted in Figure 3 and the data of Figure 4.

	d_i	Order	$O_{i,j}$	M_1	M_2	M_3	M_4
J_1	6	1,3,2	$O_{1,1}$	1	4	6	9
			$O_{1,2}$	3	2	5	1
			$O_{1,3}$	4	1	1	3
J_2	7	2,1,3	$O_{2,1}$	4	8	7	1
			$O_{2,2}$	2	10	4	5
			$O_{2,3}$	6	11	2	7
J_3	8	1,2,3	$O_{3,1}$	8	5	8	9
			$O_{3,2}$	9	3	6	1
			$O_{3,3}$	7	1	8	5
J_4	9	1,2,3	$O_{4,1}$	5	10	6	4
			$O_{4,2}$	4	2	3	8
			$O_{4,3}$	7	3	12	1

Figure 4. Due dates, operations order and processing times of operations on machines

3.2. Crossover

Figure 5 shows the two crossover operators adapted to the described encoding. Both operators cross two randomly selected chromosomes to generate a new schedule although row crossover affects jobs whereas column crossover operations:

- Row crossover: the child chromosome consists of the $1, \ldots, k$ jobs of *parent*$_1$ and the $k+1, \ldots, n$ jobs of *parent*$_2$.
- Column crossover: the child chromosome is made up of the $1, \ldots, k$ operations of *parent*$_1$ and the $k+1, \ldots, n$ operations of *parent*$_2$.

where k is a randomly selected cross point.

3.3. Mutation

The parallel job encoding used in this work forces mutations to act on machine assignments since it does not allow to interchange jobs or operations in the chromosome. Taking this constraint into account, two mutation operators have been used, however, only one of them is applied per mutation:

- Random mutation: a different machine is randomly selected to perform operation $O_{i,j}$.
- Load balancing mutation: a different machine is selected to perform operation $O_{i,j}$. The selection is based on the load of the machine in the schedule so this operator looks for balancing the machine loads.

where i-th row and j-th column are randomly selected. Note that both mutation operators always generate feasible schedules.

3.4. Objective Functions

Many time performance measures exists for JSSP, such as makespan, total flow-time, total lateness, total tardiness, and so on. Our aim is to minimize the following two objectives:

- C_{max}: Makespan. This measure returns the maximum completion time of the jobs:

$$C_{max} = max\,(C_i) \tag{2}$$

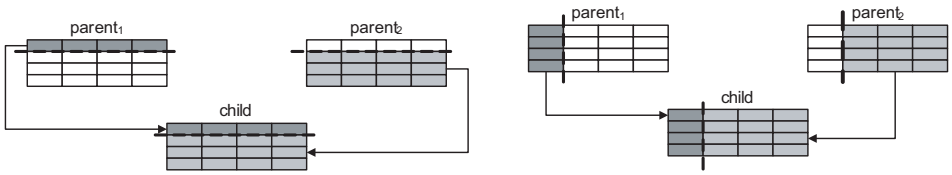
  where $i \in \{1, \ldots, n\}$ and the completion time of a job i is:

$$C_i = max\left\{s_{M_k,O_{i,j}} + p_{M_k,O_{i,j}}\right\} \tag{3}$$

  for $j \in \{1, \ldots, o\}$, and $k \in \{1, \ldots, m\}$.

- T_Σ: Total tardiness. The tardiness measures how much later than the deadline the job finishes. If the job finished earlier than d_i, it is assigned a negative lateness:

$$T_\Sigma = \sum_{i=1}^{n} max\,(0, C_i - d_i) \tag{4}$$



(a) Row crossover (b) Column crossover

Figure 5. Two crossover operators for the Parallel Job Encoding

Makespan and *total tardiness* are the most important objectives when evaluating a schedule in the context in which this work has been developed. Cost is also relevant but it can be considered a time-dependent variable and like another objectives, such as the communication capacity between two manufacturing steps, has little influence in the selection of the most suitable schedule. Note that we limited the number of objective because NSGA-II is not as successful in dealing with large dimensional problems and extremely disconnected Pareto fronts.

For the schedule example depicted in Figure 1, the makespan is $C_{max} = max\{7, 7, 8, 8\} = 8$ and total tardiness of the $T_\Sigma = 1 + 0 + 0 + 0 = 1$. Note that both objective functions depend on the processing time of machines. The estimation of this time is an important task in any manufacturing industry [11] and has a huge impact on the quality of the work plans produced by the scheduling process.

4. Neural Network Approach for Processing Times Estimation

The approach presented in this section looks for a high accuracy regression model for processing times estimation. Processing times of a machine can be described as polynomials of several input variables where variables can be combined in many different ways:

$$\sum_i \alpha_i \cdot \prod_{j=1}^{na} x_j^{\delta_{i,j}} \tag{5}$$

where α_i are the coefficients, x_j are the input variables for $j = 1, \ldots, na$, and $\delta_{i,j}$ is an indicator variable defined by:

$$\delta_{i,j} = \begin{cases} 1 & \textit{if } x_j \in \textit{ i-th term of the polynomial} \\ 0 & \textit{otherwise} \end{cases} \tag{6}$$

Note that for a given machine, there can coexist different polynomials, each one representing the estimation of the processing time of a class of the input variables. For example, in an specific machine, processing times of pieces with a thickness over a threshold could be estimated with a polynomial, and under that threshold with another polynomial.

Summarizing, the learning model must have the capacity to approximate non linear functions, to capture complex relationships in the data, and to identify the regression function to apply for each class of input variables for a machine and operation. Furthermore, the learning model must be reactive to changes in the supply chain configuration. That is, it may learn new polynomials for each updating of machinery or new operation a machine can perform. Taking into account the previous conditions, a neural network approach has been used. Neural networks are universal approximators and can easily be trained to map multidimensional nonlinear functions because of their parallel architecture.

4.1. Multilayer Perceptron

We approach each time estimation through a Multilayer Perceptron (MLP). The computations performed by the network for a single hidden layer can be written as:

$$\mathbf{y} = f(\mathbf{x}) = \mathbf{B}\varphi\left(\mathbf{AS} + \mathbf{a}\right) + \mathbf{b} \tag{7}$$

where $\mathbf{s}$ is a vector of inputs, $\mathbf{x}$ a vector of outputs, $\mathbf{A}$ and $\mathbf{a}$ are the weight matrix and the bias vectors of the first layer whereas $\mathbf{B}$ and $\mathbf{b}$ are the weight matrix and the bias vectors of the second layer. The function φ denotes a sigmoidal function $\varphi = 1/1 + \exp^{-x}$.

The standard backpropagation (BP) learning algorithm was adopted in the learning process of time estimations [8]. However, MLP networks trained with BP suffer some disadvantages: *(i)* they are easily trapped into local minima, *(ii)* they have slow convergence, and *(iii)* network topology must be determined by trial and error. To get around the first problem, the learning algorithm simply trained multiple nets and pick the best. The second and third problems are mitigated with a network optimization scheme. Specifically, the training method implements a network growing approach: the layout starts with a small network with only a single hidden unit. The network is trained until the improvement in the error over one epoch falls below some threshold. Then the method adds an additional hidden unit, with weights from inputs and to outputs (weights have been randomly initialized) and resume training. The process continues until no significant gain is achieved by adding an extra unit.

5. Results and Conclusions

A solution for multi-objective machine scheduling in the custom furniture industry has been presented. The solution is based on a multi-objective evolutionary approach together with a neural network. The system uses the NSGA-II algorithm to produce reliable schedules which are optimized on the basis of their makespan and total tardiness. Initial population combines random generation with the generation of initial schedules following priority rules. The evolutionary implementation has a parallel chromosomes representation and has tuned the NSGA-II parameters so that *(i)* the objectives are satisfied, *(ii)* the entire Pareto frontier is covered, promoting the diversity of work plans and *(iii)* machines load is balanced.

Schedules evaluation is based on multilayer perceptron networks trained with backpropagation algorithm. These neural networks learn non linear functions to approach processing times estimation. Their accuracy will determine work plan quality since objective functions of the multi-objective problem are based on these times. The system performs a continuous and automatic improvement of these estimations each time a significant number of furniture orders have been manufactured. In each learning step, neural networks are trained again taking real times (obtained from the new manufacture) into account. Note that training data selection is random which may be a problem in the presence of noise. As an improvement, a future approach will look for reducing the noise variance in order to achieve better learning results.

Although the system is still being validated, first results showed that it has reduced the differences between real and planned processing times. Note that first iterations of the learning process have a greater impact because they usually add new classes of input variables (for a machine and operation) to the training data. When most of the classes have been included, the improvement in the error becomes more stable. It should be remarked that some time estimations are still being computed with polynomials defined by experts so the whole system configuration has not been tested. Test data for training neural networks are difficult to obtain and require time measurements. However, the comparison of experts-defined polynomials with trained networks have shown a 10% lower mean error. Tests also demonstrated a time efficient response both to a huge work load and to changes in the environment, such as the definition of new client orders or changes in production due dates.

References

[1] A. Allahverdi, T. C. E. Ng, C. T. Cheng, and M. Y. Kovalyov. A survey of scheduling problems with setup times or costs. *European Journal of Operational Research*, 2006.

[2] J. Andersson. *Multiobjective Optimization in Engineering Design*. PhD dissertation, Linköpings University, Division of Fluid and Mechanical Engineering Systems, Linköping, Sweden, 2001.

[3] G. Boothroyd, P. Dewhurst, and W. Knight. *Product Design for Manufacture and Assembly*. Marcel Dekker, February 1994.

[4] C. A. Brizuela Rodríguez. *Genetic Algorithms for Shop-scheduling Problems: Partial Enumeration and Stochastic Heuristics*. PhD dissertation, Kyoto Institute of Technology, Japan, 2000.

[5] T. C. E. Cheng, Q. Ding, and B. M. T. Lin. A concise survey of scheduling with time-dependent processing times. *European Journal of Operational Research*, 152:1–13, 2004.

[6] C. A. Coello Coello. *Evolutionary Multiobjective Optimization: Theoretical Advances And Applications*, chapter Recent Trends in Evolutionary Multiobjective Optimization, pages 7–32. Springer-Verlag, 2005.

[7] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, April 2002.

[8] S. I. Gallant. Perceptron-based learning algorithms. *IEEE Transactions on Neural Networks*, 1(2):179–191, 1990.

[9] M. R. Garey, D. S. Johnson, and R. Sethi. The complexity of flowshop and jobshop scheduling. *Mathematics of Operational Research*, 1(2):117–129, 1976.

[10] S. K. Gupta and D. S. Nau. A systematic approach for analyzing the manufacturability of machined parts. *Computer Aided Design*, 27(5):323–342, 1995.

[11] J. W. Herrmann and M. M. Chincholkar. Reducing throughput time during product design. *Journal of Manufacturing Systems*, 20(6):416–428, 2001.

[12] A. Kusiak and W. He. Design of components for schedulability. *European Journal of Operational Research*, 164:185–194, 1999.

[13] K. Mesghouni, S. Hammadi, and P. Borne. Evolutionary algorithms for job-shop scheduling. *International Journal of Applied Mathematics and Computer Science*, 14(1):93–103, 2004.

[14] I. Minis, J. W. Herrmann, G. Lam, and E. Lin. A generative approach for concurrent manufacturability evaluation and subcontractor selection. *Journal of Manufacturing Systems*, 18(6):383–395, 1999.

[15] F. Moller. A scaled conjugate gradient algorithm for fast supervised learning. *Neural Networks*, 6:525–533, 1990.

[16] D. Shabtay and G. Steiner. A survey of scheduling with controllable processing times. *European Journal of Operational Research*, 155:1643–1666, 2007.