Optimizing Production Schedules and Energy Consumption with an Evolutionary Algorithm

Bogdan Filipič
Department of Intelligent Systems
Jožef Stefan Institute
Jamova 39, SI-1000 Ljubljana, Slovenia
E-mail: bogdan.filipic@ijs.si

Keywords: evolutionary algorithm, production scheduling, energy consumption, peak energy demand, cost minimization, automobile industry

Received: November 15, 2004

We present an evolutionary algorithm approach to schedule optimization for a group of production lines in a car factory. Schedules are evaluated with respect to the energy consumption over peak demand periods, while the task is to minimize the energy costs by appropriately scheduling the interruptions of processes on the lines. Tests on real problem instances show this approach gives near-optimal schedules in acceptable time.

Povzetek: v članku opisujemo učinkovito sestavljanje urnikov obratovanja z evolucijskim algoritmom in optimiranje porabe energije na proizvodnih linijah v avtomobilski tovarni.

1 Introduction

Scheduling deals with allocating activities to resources over time in such a way that given objectives are optimized, while temporal constraints and resource limitations are satisfied. Problems of this kind appear in manufacturing, timetabling, vehicle routing, design of computer operating systems and other domains. Because of its great practical importance, scheduling has permanently attracted research interests. Following the attempts in the fields of Operations Research and Artificial Intelligence with limited success in practice, Evolutionary Computation [1] has recently offered means of producing near-optimal schedules for complex problems at reasonable computational costs [3]. A number of applications of evolutionary algorithms in scheduling have been reported [2, 6, 11]. Nevertheless, there are still open issues to be addressed in the development of evolutionary scheduling systems. Above all, real-world problems and realistic criteria for schedule optimization should be considered [5].

Using evolutionary computation techniques, we deal with a class of real-world problems with schedule cost related to resource management. Our previous application oriented studies include scheduling of operations in a production unit of a textile factory, where the objective was to ensure optimal energy consumption [7], and scheduling activities in ship repair in order to balance the work load for workers of various trades [8].

In this paper we describe production scheduling on a group of production lines of an automobile factory. The problem is non-typical in two respects. First, it requires process interruptions to be scheduled rather than processes themselves. Second, the optimality criterion is not based on a traditional schedule performance measure, such as overall processing time, but related to energy consumption. The objective is to schedule interruptions of the running processes in such a manner that energy consumption over the peak demand periods is minimized. In addition, the schedules are subject to time and resource constraints that have to be strictly satisfied.

Design of an evolutionary scheduling system for this problem and the initial practical results were presented in [9, 10], while here the focus is on an improved version of the system and its evaluation. The paper explains the scheduling problem and the schedule cost that is related to energy consumption, describes the employed scheduling system, provides the results of its evaluation on real problem instances, and discusses them in view of regular exploitation at the plant.

2 Production Scheduling Based on Energy Costs

Production systems relying on intense energy consumption, such as steel plants and other heavy industries, are faced with peak demand periods. These are the time periods over which their power demand exceeds a given limitation and the excess has to be paid at a higher rate. This measure is imposed by the energy supplier to minimize the total energy consumption over critical periods. There are several ways of reducing the peak power demand: activation of internal energy sources, interruption of energy-intensive processes, and appropriate production scheduling.

2.1 The Scheduling Task

The focus of energy consumption management in the considered factory is in the car-body production unit. The unit consists of six lines of hydraulic presses that perform cutting and shaping. A line in operation is regarded as an individual work process. Power demands of the processes vary from 20kW to 370kW. The unit operates according to a daily production plan that specifies which of the lines are in operation and what is their work time. Power demand of the unit equals to the sum of power demands of the running processes. Other energy consumers at the plant contribute to the so called background power demand, P_b. The total power demand of the plant, P, consists of the demand of the pressing unit and the background demand. To asses the energy costs, the total demand is related to the prescribed limitation $P_{\max}$, also called the target load. Figure 1 shows an example of daily profiles for background demand, total power demand and target load.

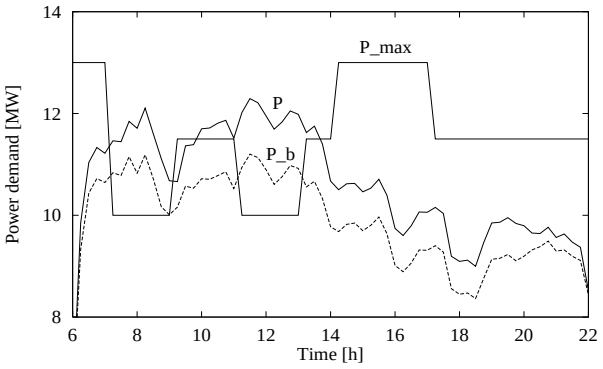

Figure 1: Background demand P_b, total power demand P, and target load $P_{\max}$ on the production lines

The efforts to reduce the target load excess are concentrated on line production in the pressing unit, since it is an intense energy consumer and also more suitable for scheduling than background processes. Two approaches are combined in the unit: process interrupting and scheduling. Process interruptions are either intended as breaks for the staff or can be spent to change machine tools and perform maintenance on the lines. The idea is to schedule these activities in such a way that the daily production plan is realized, while the contribution of the unit to the target load excess is minimized.

To balance between the conflicting requirements of plan fulfilment and reduction of the target load excess, the following constraints have been imposed on schedules:

- duration of process interruptions, T_0,
- minimum period of time between two interruptions of a process, T,
- maximum number of processes that can be interrupted simultaneously, M.

Taking into account these constraints, process interruptions have to be scheduled so as to minimize the target load excess contributed by the production lines.

2.2 Schedule Cost

The schedule cost to be minimized is formally defined as follows. Let $P_i(t)$, $i = 1, \ldots, N$, denote the power demands of the operating production lines in the pressing unit. The total power demand of the plant is

$$P(t) = \sum_{i=1}^{N} P_i(t) + P_b(t) \tag{1}$$

where $P_b(t)$ represents the background demand. Then the contribution of the considered processes to the target load excess at time t is

$$P_{\text{exc}}(t) = \begin{cases} \sum_{i=1}^{N} P_i(t); & P_b(t) \geq P_{\max}(t) \\ P(t) - P_{\max}(t); & P_b(t) < P_{\max}(t) \;\&\; P(t) > P_{\max}(t) \\ 0; & \text{otherwise} \end{cases} \tag{2}$$

and the energy consumption resulting from the target load excess equals to

$$W_{\text{exc}} = \int_t P_{\text{exc}}(t)\, dt. \tag{3}$$

W_{exc} represents the cost of interruption schedules which is to be minimized. It is to be noted, however, that power demands are in practice sampled using certain time interval Δt and integral (3) is approximated by

$$\sum_t P_{\text{exc}}(t)\, \Delta t. \tag{4}$$

3 The Scheduling System

The scheduling system generates daily schedules of process interruptions and calculates the expected reduction of the target load excess. It accepts the following input information:

- estimates of power demand profiles for the processes to be executed,
- an estimate of the background demand profile,
- the target load profile, and
- constraints to be considered in schedule construction.

The power demand estimates are based on data recorded over previous days and on production plan for the current day. As the production does not change rapidly, the estimates are rather accurate and make it possible to generate realistic production schedules.

The scheduling algorithm is designed to solve problem instances with arbitrary power demand profiles and can operate at various time discretizations. The core of the algorithm is a $(\mu + \lambda)$ evolution strategy [14]. It iteratively improves the schedules through the following sequence of steps:

Step 0: Generate an initial population of μ schedules by randomly assigning starting times to process interruptions.

Step 1: Generate λ descendants from μ parents by applying local transformations to schedules.

Step 2: Select μ best solutions out of $\mu + \lambda$ available, and make them parents for the next generation.

Step 3: If maximum number of generations is reached, exit, otherwise go to Step 1.

The best schedule found during this search process is returned as a suboptimal solution to the problem. Both, inserting the interruptions into a schedule at the initialization step and local schedule transformations are performed in such a way that constraints imposed on schedules remain satisfied. This is achieved by maintaining a direct representation of schedules within the algorithm and checking the constraints. Schedules are represented as two-dimensional arrays with the number of rows equal to the number of running processes, and columns to time intervals considered during scheduling. Each element of the array holds a value denoting the process status at the corresponding time interval. The status can be: *interrupted*, which means the process is interrupted, *interruption possible*, which means the process is running and it is possible to interrupt it, or *interruption not possible*, which means the process is running but cannot be interrupted due to the constraints.

Schedule transformations are carried out on random basis and include:

– inserting an interruption into a schedule,

– deleting an interruption from a schedule,

– shifting an interruption within a schedule.

Insertion of an interruption consists of finding a random time slot in the schedule with *interruption possible*, changing its status to *interrupted* and updating the status of the slots affected through constraint values T and M to *interruption not possible*. Deletion of an interruption includes random selection of an *interrupted* slot, changing its status to *interrupt possible*, and updating the status of the slots that are no more effected through constraint values T and M to *interruption possible*. Shifting of an interruption consists of its deletion at the current time slot and insertion at another time slot.

4 Evaluation and Results

4.1 Tests on Real Scheduling Problems

The scheduling system was initially tested on a set of problem instances based on real data recorded at the plant. The data were used as input to optimize daily schedules for the production lines. The constraints for schedule construction were set as follows. Duration of process interruptions, T_0, was 30 minutes. Each process had to run continuously for at least four hours between two interruptions ($T = 240$ min), and at most three process interruptions were permitted to take place simultaneously ($M = 3$). Time step used during search for assigning starting times to interruptions was 5 minutes.

The scheduling algorithm was run for 200 generations. The population size and the number of offspring generated in each generation were $\mu = \lambda = 20$. For each problem instance, the algorithm was executed 10 times, and both the best and average results were recorded. The optimized schedules of process interruptions were produced in the form shown in Table 1.

Table 1: An optimized interruption schedule for the production lines

Line number	Interruption times	
1	8:00–8:30	12:15–12:45
2	7:25–7:55	11:55–12:25
3	7:00–7:30	12:20–12:50
4	7:15–7:45	11:45–12:15
5		11:00–11:30
6		11:30–12:00

The evaluation confirmed that schedule optimization can substantially contribute to the decrease of energy costs in the production unit. Energy consumption resulting from the target load excess on the lines was reduced by at least 25% on workdays, but in most cases by about 30%. Table 2 shows the achieved reduction averaged over 10 runs of the optimization algorithm for each day in a two-week period. The reproducibility of the reduction in kWh obtained in 10 runs for each problem instance was within 2%.

4.2 On the Optimality of Schedules

Additional numerical experiments were carried out to check how close the optimized schedules are to the true optimal ones. For this purpose a selected scheduling problem representing a typical situation at the plant was used. All six production lines were required to operate and estimates of power demand profiles shown in Fig. 1 were used. If case of no process interruptions, the target load excess would amount to 3218.3 kWh. For this power demand situation, test problem instances of various complexity were defined. Their complexity was varied through constraint values T_0, T, and M.

Table 2: Average reduction of the target load excess obtained by the scheduling system

Day	Target load excess [kWh]	Reduction [kWh]	[%]
Mon	2616.5	1000.4	38.2
Tue	2569.6	970.5	37.8
Wed	3218.3	1012.6	31.5
Thu	2892.2	926.2	32.0
Fri	3055.1	931.6	30.5
Sat	655.0	413.0	63.0
Sun	0.0	0.0	0.0
Mon	2461.2	810.1	32.9
Tue	2117.7	636.6	30.1
Wed	2910.3	836.8	28.7
Thu	2752.8	803.3	29.2
Fri	2523.5	869.8	34.5
Sat	0.0	0.0	0.0
Sun	0.0	0.0	0.0

To denote a problem instance with particular constraint values, will use the notation (T_0, T, M), where times T_0 and T are given in minutes, and $M \in [1..N]$. The test set of problem instances consisted of $(30, 240, 3)$, $(30, 240, 1)$, $(30, 120, 3)$, $(30, 120, 1)$. Note that $(30, 240, 3)$ is the default setting of constraint values used at the plant, while additional settings resulting in more demanding problems were chosen to further check the performance of the developed scheduling system.

For the evaluation purposes, optimal schedules for the selected problem instances were produced by the Constraint Logic Programming approach. Constraint Logic Programming (CLP, [4, 12]) is a generalization of logic programming [13] where unification is replaced by a more general mechanism of constraint satisfaction over a specific computation domain, such as Boolean, finite or real. It is capable of finding optimal solutions to the problems of manageable size. We used the ECL^{i}PS^e CLP environment and its finite domain solver CLP($\mathcal{F}$). Unfortunately, the scheduling task introduced in Subsection 2.1 is too complex to be treated generally. However, particular problem instances can be handled individually by considering their specificities during problem solving.

Schedule costs found by this tool and by the evolutionary scheduling system are compared in Table 3. More clear picture of the evolutionary algorithm performance can be obtained from Table 4 which shows the deviation of the schedule improvement from the optimum gained with CLP.

These results are very informative for practical assessment of the evolutionary scheduling algorithm. While the initial tests on real problems showed that potential decrease of energy costs is at the expected level, we now have an absolute measure of the scheduling algorithm performance. It is particularly encouraging, that under constraint setting $(30, 240, 3)$, which is usually used at the plant, the result is very close to the optimum. For hypothetical problems with more complex spaces the gap to the optimum increases, but we believe that initial results given in this paper can still be improved.

Table 3: Optimal schedule costs in kWh found with CLP and suboptimal costs obtained with the evolutionary algorithm (EA)

Problem instance (T_0, T, M)	CLP	EA best	EA average
(30, 240, 3)	2209.1	2211.5	2213.1
(30, 240, 1)	2374.3	2385.1	2419.7
(30, 120, 3)	2185.8	2187.1	2187.4
(30, 120, 1)	2345.8	2365.8	2390.4

Table 4: Deviation of schedule cost improvement by the EA from the optimal improvement obtained with CLP

Problem instance (T_0, T, M)	EA best	EA average
(30, 240, 3)	0.2%	0.4%
(30, 240, 1)	1.3%	5.4%
(30, 120, 3)	0.1%	0.2%
(30, 120, 1)	2.3%	5.1%

Certainly, one may ask whether it is possible to apply the CLP system for regular scheduling at the plant. It turns out that its advantage of guaranteed optimal solutions comes at some other costs. Solving problems of this type with CLP is only efficient on individual basis, where additional constraints for schedules are derived from input data (e.g. feasible time intervals for process interruptions) and implemented to prune the search space. Further increase of problem complexity would sooner or later exceed the capabilities of the system. The CPU time spent to obtain optimal solutions with the CLP system depends very much on the problem instance, and ranges from a few minutes to several hours on a Pentium computer. On the other hand, the execution of the evolutionary algorithm on the same computer requires about half a minute for each problem instance, and only slightly increases with problem complexity.

5 Conclusion

An evolutionary algorithm was developed to schedule process interruptions on car-body production lines in an automobile factory where the objective is to decrease power demand over critical periods. In a comparative study the results of the evolutionary algorithm were assessed with regards to optimal results found by a CLP system. The comparison was beneficial in that it confirmed the evolutionary algorithm is capable of finding near-optimal results for a typical scheduling task appearing on the lines.

The approach was implemented as a process scheduling

module within a system for energy consumption management at the plant. It facilitates monitoring and control of energy consumption, while its primary role is to assist the process supervisor in preparing daily production schedules for the pressing unit.

In practical exploitation it has turned out the system is beneficial under certain amount of work load for the production lines. When the amount of orders is low, the lines are not heavily loaded and the resulting power demand does not exceed the target load. Hence there is no need for interruption scheduling and energy consumption optimization. On the other hand, the plant may get numerous orders and tight deadlines to accomplish them, and therefore deliberately decides not to interrupt the line processes as the additional energy costs are less than the penalties for not fulfilling the orders in time. Between these two extremes, there are however modes of operation where the system is regularly used and contributes to the decrease of production costs.

Acknowledgement

This work was supported by the Ministry of Education, Science and Sport of the Republic of Slovenia, and by the companies Revoz and INEA.

References

[1] T. Bäck, D. B. Fogel, Z. Michalewicz (Eds.): *Handbook of Evolutionary Computing*. Institute of Physics Publishing, Bristol, and Oxford University Press, New York, 1997.

[2] J. Biethahn, V. Nissen (Eds.): *Evolutionary Algorithms in Management Applications*. Springer-Verlag, Berlin, 1995.

[3] R. Bruns: Scheduling. Chapter F1.5 in T. Bäck, D. B. Fogel, Z. Michalewicz (Eds.), *Handbook of Evolutionary Computing*. Institute of Physics Publishing, Bristol, and Oxford University Press, New York, 1997.

[4] J. Cohen: Constraint logic programming languages. *Communications of the ACM*, 33 (7), 52–68, 1990.

[5] D. Corne, P. Ross: Practical issues and recent advances in job- and open-shop scheduling. In D. Dasgupta, Z. Michalewicz (Eds.), *Evolutionary Algorithms in Engineering Applications*, pp. 531–546. Springer-Verlag, Berlin, 1997.

[6] D. Dasgupta, Z. Michalewicz (Eds.): *Evolutionary Algorithms in Engineering Applications*. Springer-Verlag, Berlin, 1997.

[7] B. Filipič: Enhancing genetic search to schedule a production unit. In B. Neumann (Ed.), *Proceedings of the 10th European Conference on Artificial Intelligence ECAI '92*, pp. 603–607, Vienna, Austria, 1992. John Wiley, Chichester. Also published in J. Dorn, K. A. Froeschl (Eds.), *Scheduling of Production Processes*, pp. 61–69. Ellis Horwood, Chichester, 1993.

[8] B. Filipič: A genetic algorithm applied to resource management in production systems. In J. Biethahn, V. Nissen (Eds.), *Evolutionary Algorithms in Management Applications*, pp. 101–111. Springer-Verlag, Berlin, 1995.

[9] B. Filipič: A hybrid optimization algorithm for energy consumption management at a motor plant. In *Proceedings of the 5th European Congress on Intelligent Techniques and Soft Computing EUFIT '97*, vol. 1, pp. 717–721, Aachen, Germany, 1997.

[10] B. Filipič, D. Zupanič: Near-optimal scheduling of line production with an evolutionary algorithm. In *Proceedings of the 1999 Congress on Evolutionary Computation CEC 99*, pp. 1124–1129, Washington, D.C. IEEE, Piscataway, 1999.

[11] K. S. Hindi, H. Yang, K. Fleszar: An Evolutionary Algorithm for Resource-Constrained Project Scheduling. *IEEE Transactions on Evolutionary Computation*, 6 (5), 512–518, 2002.

[12] J. Jaffar, J.-L. Lassez: Constraint logic programming. In *Proceedings of the 14th ACM Symposium on Principles of Programming Languages*, pp. 111–119, Munich, Germany, 1987.

[13] J. W. Lloyd: *Foundations of Logic Programming*. Springer-Verlag, Berlin, 1987.

[14] H.-P. Schwefel: *Evolution and Optimum Seeking*. John Wiley, New York, 1995.