Available online at www.sciencedirect.com
SCIENCE DIRECT®

Computers in Industry 56 (2005) 161–168

COMPUTERS IN INDUSTRY

www.elsevier.com/locate/compind

Genetic algorithms for integrated preventive maintenance planning and production scheduling for a single machine

N. Sortrakul, H.L. Nachtmann, C.R. Cassady*

Department of Industrial Engineering, University of Arkansas, 4207 Bell Engineering Center, Fayetteville, AR 72701, USA

Received 3 March 2003; received in revised form 3 November 2003; accepted 28 June 2004
Available online 16 December 2004

Abstract

Despite the inter-dependent relationship between them, production scheduling and preventive maintenance planning decisions are generally analyzed and executed independently in real manufacturing systems. This practice is also found in the majority of the studies found in the relevant literature. In this paper, heuristics based on genetic algorithms are developed to solve an integrated optimization model for production scheduling and preventive maintenance planning. The numerical results on several problem sizes indicate that the proposed genetic algorithms are very efficient for optimizing the integrated problem.

Keywords: Genetic algorithms; Preventive maintenance; Production scheduling; Optimization

1. Introduction

Production scheduling and preventive maintenance (PM) planning are among the most common and significant problems faced by the manufacturing industry. Production schedules are often interrupted by equipment failures, which could be prevented by proper preventive maintenance. However, recommended PM intervals are often delayed in order to expedite production. Despite the trade-offs between the two activities, they are typically planned and executed independently in real manufacturing settings even if manufacturing productivity can be improved by optimizing both production scheduling and PM planning decisions simultaneously.

Numerous studies have been conducted in these two areas in the past decades. Shapiro [1] and Pinedo [2] reviewed various papers in production scheduling. Similarly, Sherif and Smith [3] and Dekker [4] reviewed several studies using maintenance optimization models. However, almost all relevant studies considered production scheduling and PM planning as two independent problems and therefore solve them separately.

Only a few studies have tried to combine and solve both problems simultaneously. Graves and Lee [5] presented a single-machine scheduling problem with the objective to minimize the total weighted

* Corresponding author. Tel.: +1 479 575 6735;
fax: +1 479 575 8431.
E-mail address: cassady@engr.uark.edu (C.R. Cassady).

0166-3615/$ – see front matter
doi:10.1016/j.compind.2004.06.005

completion time of jobs. However, only one maintenance activity can be performed during the planning horizon. Lee and Chen [6] extended Graves and Lee's research to parallel machines, but still permitting only one maintenance action. Qi et al. [7] considered a similar single-machine problem with the possibility for multiple maintenance actions, but the risk of not performing maintenance is not explicitly included in the model. Cassady and Kutanoglu [8] developed an integrated mathematical model for a single-machine problem with total weighted expected completion time as the objective function. Their model allows multiple maintenance activities and explicitly captures the risk of not performing maintenance.

In this paper, we develop genetic algorithm heuristics to solve the integrated production scheduling and preventive maintenance planning problem for a single machine introduced in Cassady and Kutanoglu [8]. The following section, Section 2, contains an overview of the integrated production scheduling and PM planning problem. Section 3 briefly describes the proposed genetic algorithm procedures. The experimental results of multiple problem sizes appear in Section 4. The conclusions are summarized in Section 5.

2. Integrated production scheduling and PM planning problem

This section describes the integrated model and proposed solution procedures for a single-machine production scheduling and PM planning problem presented by Cassady and Kutanoglu [8].

2.1. Production scheduling problem

The deterministic single-machine scheduling problem with the objective to minimize the total weighted completion time is considered. Assuming that every job is ready at the beginning of production period and the preemption of one job for another is prohibited, the optimal solution for this problem can simply be obtained from the WSPT (weighted shortest processing time) rule. As mentioned in Pinedo [2], jobs are scheduled under this rule in descending order of the ratio of weight to processing time.

2.2. Preventive maintenance planning problem

Suppose the time to failure of a machine is governed by a Weibull probability distribution having scale parameter η and shape parameter β greater than 1. When a machine fails, minimal repair is performed to restore the machine back to its operating condition without altering its effective age. Since $\beta > 1$, the machine failure rate increases over time. Therefore, PM may be used to stop the increasing risk of machine failure by restoring it back to a "good as new" condition.

Since PM restores the machine to a "good as new" condition, machine performance can be modeled using a renewal process, where the renewal points correspond to the completion of a PM activity. Since repair is minimal, failures occur during each "cycle" of the renewal process according to a non-homogeneous Poisson process (NHPP) having intensity function $z(t)$ where $z(t)$ corresponds to the hazard function of a new machine. As a result, the expected value of the number of failures that occur during a single cycle of the renewal process is

$$m(\tau) = \int_0^\tau z(t)\,dt = \int_0^\tau \frac{\beta}{\eta^\beta} t^{\beta-1}\,dt = \left(\frac{\tau}{\eta}\right)^\beta \quad (1)$$

Therefore, an "average" cycle of the renewal process includes τ time units of operation, $m(\tau)$ machine repairs of length t_r, and a single PM action of length t_p. The resulting steady-state machine availability is

$$A(\tau) = \frac{\tau}{\tau + m(\tau)t_r + t_p} \quad (2)$$

Differentiation and algebraic analysis reveal the optimal PM interval, which maximizes machine availability, to be

$$\tau^* = \eta\left[\frac{t_p}{t_r(\beta-1)}\right]^{1/\beta} \quad (3)$$

2.3. Integrated problem

In the integrated problem, a machine possesses all production requirements defined in Sections 2.1 and 2.2. In addition, jobs cannot be preempted for PM. Also, jobs interrupted by machine failure can be resumed after machine repair without any additional time penalty. For this problem, there are two major tasks that must be considered simultaneously. The first one is to find the best sequence of n jobs out of $n!$ possible sequences. The other one is to decide whether to perform the PM before processing each job in the sequence. Thus, there are 2^n possible sets of PM decisions for each job sequence. Therefore, the integrated problem has a total of $(n!)2^n$ feasible solutions.

The job completion times of the integrated problem are stochastic since the machine may or may not fail during each job and PM decisions can change the stochastic process governing machine failure. Let $p_{[i]}$ denote the processing time of the ith job in the sequence, $\bar{a}_{[i-1]}$ denote the age of machine immediately before processing the ith job, $a_{[i]}$ denote the age of machine immediately after finishing the ith job, and

$$y_{[i]} = \begin{cases} 1 & \text{if PM is performed prior to the } i\text{th job} \\ 0 & \text{otherwise} \end{cases} \quad i = 1, 2, \ldots, n \tag{4}$$

Since PM renews the machine and repair is minimal,

$$\bar{a}_{[i-1]} = a_{[i-1]}(1 - y_{[i]}) \quad i = 1, 2, \ldots, n \tag{5}$$

$$a_{[i]} = \bar{a}_{[i-1]} + p_{[i]} \quad i = 1, 2, \ldots, n \tag{6}$$

Note that $a_{[0]}$ is the age of the machine prior to making any decisions. Since failures occur according to an NHPP, the expected value of the completion time of the ith job can be obtained by

$$E\left(C_{[i]}\right) = \sum_{k=1}^{i} \left\{ t_p y_{[k]} + p_{[k]} + t_r \left[m\left(a_{[k]}\right) - m\left(\bar{a}_{[k-1]}\right) \right] \right\} \quad i = 1, 2, \ldots, n$$

Let $w_{[i]}$ be the weight of the ith job in the sequence. The objective function, the total weighted expected completion time (TWECT), of the integrated problem is

$$\sum_{i=1}^{n} w_{[i]} E(C_{[i]}) \tag{8}$$

In Cassady and Kutanoglu [8], the results of using a total enumeration procedure to solve thousands of instances of small-size (3-job) problems indicate that 96% of the optimal solutions of integrated problems utilize the WSPT job sequence. However, only 6% of integrated solutions utilize both the WSPT job sequence and the independently obtained optimal PM interval (τ^*), which are the results that would be obtained from solving these two problems independently. Thus, it is beneficial to solve the integrated problem to determine more effective PM planning. Based on these findings, a heuristic, which uses the WSPT job sequence and only performs total enumeration to find the best PM decisions for the WSPT job sequence, is proposed for solving larger problems.

3. Genetic algorithm procedures

Initially, genetic algorithm (GA) procedures were mainly applied to research topics in the area of artificial intelligence. However during the past decade, GA has become one of the most well-known search heuristics and is widely used in many combinatorial optimization problems including machine scheduling. Chambers [9] reviewed applications of GA in several research areas. For the single-machine scheduling problem, Gupta et al. [10] applied GA to minimize the variance of flow time. Liu and Tang [11] proposed a modified GA for a single-machine scheduling problem with job ready time constraints.

In general, GA works by keeping a population of a fixed number of candidate solutions (or chromosomes), which are represented by a sequence of numbers. Each element (or gene) of a chromosome is used to represent a certain feature of the solution. A population at a specific time is called a generation. In each generation, the fitness value (related to the objective function of the problem) of each chromosome in the current population is calculated. Based on a predetermined reproductive strategy, certain chromosomes are selected as parents and mated using a crossover operation to create a new set of offspring chromosomes. Selected offspring are mutated according to a predetermined mutation scheme. The next generation is then created from the combination of parents and offspring. This process is repeated until a specified stopping condition is reached.

3.1. Genetic algorithms for integrated problem

Three GA-based heuristics are developed for solving the integrated problem. The first heuristic is similar to the heuristic proposed by Cassady and Kutanoglu [8] in order to benefit from the knowledge that 96% of optimal solutions of small experimental problems utilize the WSPT job sequence. This heuristic (GA1) utilizes the WSPT job sequence and only uses GA to determine the PM decisions. As discussed in Section 3.1.1, the second heuristic (GA2) applies GA procedure to both the job scheduling and PM planning problems simultaneously. The third heuristic (GA3) is a modification of GA2 where GA for job scheduling is performed every time that a fixed number of generations of GA for PM planning have been evaluated. This fixed number of generations is referred to as the ‘‘PM per job sequence ratio’’, or R_{pms}. Hence, GA3 becomes identical to GA2 if R_{pms} is equal to 1. It is important to note that GA3 has more opportunity than GA2 to find good PM decisions for the current job sequence chromosomes before it moves on to the next generation of job sequences. In this study, the heuristics terminate when a fixed number of generations are evaluated. The GA characteristics implemented for the above heuristics are described next.

3.1.1. Chromosome representation

For the 3-job integrated problem, the chromosome {$\underline{1}3\underline{0}2\underline{1}1$} can be used to represent the solution that job 3 is the first job to be processed followed by job 2 and then job 1 where PM is scheduled before job 3 and job 1. Note that underlined binary number indicates whether or not PM is performed before the *i*th job. However, the crossover and mutation operators on this type of chromosome are not obvious. Therefore, two separate chromosomes, one for the job sequence and the other for the PM decisions, are proposed as an alternative in this study. The chromosome in previous example can simply be replaced by the job sequence chromosome {321} and the PM chromosome {101}. In this research, the same crossover point and mutation point are used for both chromosomes.

3.1.2. Reproductive strategy

Fitness value may simply be used to generate the probability distribution for selecting chromosome for parenthood. However, Reeves [12] mentioned that selecting a chromosome based solely on its fitness value is rarely effective as a ranking approach. The ranking mechanism, where a probability to be selected of a chromosome is derived from its ranking rather than its fitness value discussed in Reeves [13], is used to select a certain number of parent chromosomes into the mating population. In other words, the better fitness (higher ranking) chromosome has more chance to be selected than the lower ranking chromosome. The elitist strategy, in which the new generation consists of the best solution (chromosome) of the current generation and the remaining members of the population selected from the offspring, is also implemented.

3.1.3. Crossover

The one-point crossover, where the string after the randomly generated crossover point is directly replaced by the string from the other parent, is applied to the binary coding PM decisions chromosome. The C1 crossover explained in Reeves [14] is implemented on the job sequence chromosome. The C1 crossover creates offspring with a feasible job sequence while preserving the absolute position of the jobs before the crossover point from one parent and the relative position of jobs taken from the other parent. For example, C1 crossover on two parents, P1, {12345}, and P2, {42531}, assuming that the random crossover point is after the second job yields the offspring O1, {12453}, and O2, {42135}. The crossover probability, P_{c}, is set to 1 in this study.

3.1.4. Mutation

For the PM chromosome, the one-point mutation, where the value of the gene at the randomly selected mutation point is changed from 0 to 1 or vice versa, is utilized. A shift mutation, where a randomly chosen gene is shifted to the right for a random number of places, is applied to the job sequence chromosome. In this version of shift mutation, if the number of places to move is beyond the last job in the sequence, the count continues from the first job and then proceeds to the next job in the sequence. Reeves [13,14] mentioned that several studies indicate that GA works better if the probability to perform mutation is changed as the iteration number increases. Therefore, this research investigates both fixed mutation probability and adaptive mutation probability. In the adaptive case, the mutation probability (P_{m}) of next generation is reduced by the mutation reduction rate (R_{m}). However, P_{m} is restored to the original value if the ratio $v_{\min}/v_{\max}$ is higher than the diversity limit (D) where v is the fitness of each chromosome in a certain generation.

3.1.5. Population size

The population size of GA should be large enough to adequately cover the solution space without incurring excessive computation effort. Reeves [13] reported that many authors suggest that a population size as small as 30 is enough to produce satisfactory results. Therefore, a population of 30 chromosomes is used in all problems.

3.1.6. Initial population

Although many GA heuristics choose the initial population at random, this study also investigates the performance of GA2 and GA3 with both random initial population and good initial population obtained by the WSPT rule.

4. Results

In this section, the proposed GA-based heuristics (GA1, GA2, and GA3) are compared with the total enumeration approach (TE) and the heuristic (H) suggested by Cassady and Kutanoglu [8]. Three problem sizes (small, medium, and large) are used to compare the performance of the algorithms. All algorithms are evaluated for the small size problems. Because of the excessive computation of the enumerative strategies, TE is only considered for the small size problems and H is not considered for the large size problems.

In this paper, an experimental structure based on the 2^4 factorial design similar to the structure used by Cassady and Kutanoglu [8] is utilized. Each experiment consists of a certain number of randomly generated problems where,

- The initial age of the equipment $a_{[0]}$ is a discrete uniform random variable over the integers 1, 2, . . ., 100.
- The weight of a job is a discrete uniform random variable over the integers 1, 2, . . ., 10.
- The processing time of a job is a discrete uniform random variable over the integers 1, 2, . . ., P_{max}.
- The four controlled factors, P_{max}, β, t_p, and t_r, of each experiment are shown in Table 1.
- The scale parameter η is set to 175.

In order to select an effective mutation scheme for the integrated problem, both fixed and adaptive mutation schemes with various mutation probability are evaluated. Table 2 summarizes the effect of different mutation probabilities (P_m) for both the fixed and adaptive cases on GA2 using 1600 instances of 6-job problems (100 instances per experiment). For each instance, the initial population is randomly generated and GA2 is executed using a population size of 30 for

Table 1
Controlled factors of experimental design

Trial	β	t_p	t_r	P_{max}
1	2	5	15	50
2	2	5	15	100
3	2	5	25	50
4	2	5	25	100
5	2	10	15	50
6	2	10	15	100
7	2	10	25	50
8	2	10	25	100
9	3	5	15	50
10	3	5	15	100
11	3	5	25	50
12	3	5	25	100
13	3	10	15	50
14	3	10	15	100
15	3	10	25	50
16	3	10	25	100

Table 2
The effect of different mutation probabilities (P_m) on 6-job problems

Scenario	Average TWECT	Optimal solution found (%)	Average CPU time (s)
$P_m = 0.05$ fixed	3251.490	92.06	0.2810
$P_m = 0.05$ adaptive	3251.488	91.94	0.2818
$P_m = 0.10$ fixed	3251.065	97.00	0.2829
$P_m = 0.70$ fixed	3250.784	99.81	0.2981
$P_m = 0.80$ fixed	3250.779	100.00	0.2999
$P_m = 0.80$ adaptive	3250.780	99.94	0.2925
$P_m = 0.90$ fixed	3250.779	100.00	0.3021

Table 3
Average algorithm performance for 6-job problems

Algorithms	Average TWECT	Optimal found (%)	Deviation from optimal (%)	CPU time (s)
TE	3250.779	100.00	0.000	0.3168
H	3251.078	93.00	0.007	0.0004
GA1	3251.078	93.00	0.007	0.3335
GA2	3250.779	100.00	0.000	0.3550
GA3	3250.779	100.00	0.000	0.3244

Table 4
Average algorithm performance for 15-job problems

Algorithms	Average TWECT	Same result as H (%)	Better result than H (%)	Deviation from H (%)	CPU time (s)
H	17436.584	100.00	0.00	0.0000	0.5765
GA1	17436.672	99.25	0.00	0.0003	3.7919
GA2_1	17439.071	49.50	25.00	0.0108	4.1344
GA2_2	14438.054	50.50	27.13	0.0069	4.0640
GA3_1	17434.725	65.00	30.75	−0.0096	3.7790
GA3_2	17434.396	65.75	31.50	−0.0109	3.7801

500 generations. For the adaptive P_m case, $D = 0.99$ and $R_m = 0.95$ are implemented. The results indicate that the computation time increases when P_m increases due to additional mutation operations. The average weighted completion time (objective function) generally improves when P_m is increased. In fixed P_m case, both $P_m = 0.8$ and 0.9 find the optimal solution for all instances, however, detailed analysis of the results indicates that $P_m = 0.9$ requires more generations to find the optimal solution. Although the adaptive $P_m = 0.8$ does not obtain the optimal solution in one instance, it yields the optimal solution in every instance if 600 generations are used. The adaptive $P_m = 0.8$ with $D = 0.99$ and $R_m = 0.95$ is chosen.

The results for 6-job (small size) problems are summarized in Table 3. In this scenario, 100 instances of problems for all 16 experiments presented in Table 1 are generated. The adaptive $P_m = 0.8$ with $D = 0.99$, $R_m = 0.95$, a population size of 30, and 600 generations are applied to all three GA heuristics. $R_{pms} = 30$ is used in GA3. GA2 and GA3 perform very well and yield the optimal solution in all problems. Since GA1 assumes the WSPT job sequence, the best solution that it can obtain is the solution of H. In this testing, GA1 is able to provide the same solution as H in every problem. Although the CPU time of H is much smaller than GA1, the CPU time of H increases dramatically when the problem size grows.

The comparisons for 15-job (medium size) problems are shown in Table 4. For each experiment, 50 instances are created. All three GA heuristics utilize the adaptive $P_m = 0.8$ with $D = 0.99$, $R_m = 0.95$, and a population size of 30. Preliminary experiments indicate that 3000 generations and $R_{pms} = 30$ yield good results with GA3. GA1 still performs comparatively well against H since GA1 obtains the H solution in 99.25% of tested instances. GA2 and GA3 with a random initial population are referred as GA2_1 and GA3_1, respectively. GA2 and GA3 with the entire initial population using the WSPT job sequence are denoted as GA2_2 and GA3_2, respectively. In GA2_2 and GA3_2, the algorithm can focus on searching for a good PM schedule for the WSPT job sequence during the early generations and then gradually diversify to consider other good job sequences by mutation and crossover operators in later generations. In general, GA3_2 combines the ability to find good PM decisions for the WSPT job sequence of GA1 with the diversity in searching the solution space of GA2_2. GA2_1 and GA2_2 perform considerably well compared to H. GA2_1 yields the same solution as H in 49.5% and a better solution than H in 25.0% of tested problems. Similarly, GA2_2 obtains the same solution in 50.5% and a better solution in 27.13% of tested problems. However, the average objective function values and the positive deviations from H of GA2_1 and GA2_2 indicate that GA2 is not as efficient as H. On average, both GA3_1

Table 5
Average algorithm performance for 30-job problems

Algorithms	Average TWECT	Best solution (%)	Worst solution (%)	CPU time (s)
GA1	63656.273	41.63	31.63	7.3488
GA2_2	63676.185	28.25	57.13	7.9685
GA3_2	63650.292	59.75	13.63	7.4965

Table 6
Average algorithm performance for 8-job problems

Algorithms	Average TWECT	Optimal solution (%)	Deviation from optimal (%)	CPU time (s)
TE	6132.505	100.00	0.0000	93.9158
H	6132.704	85.63	0.0031	0.0024
GA3_2	6132.505	100.00	0.0000	2.1163

Table 7
Average algorithm performance for 19-job problems

Algorithms	Average TWECT	Same result as H (%)	Better result than H (%)	Deviation from H (%)	CPU time (s)
H	29534.565	100.00	0.00	0.0000	11.8633
GA3_2	29532.241	56.88	29.38	0.0049	4.7872

and GA3_2 perform better than H as seen by the average objective function values and the negative deviations from H. GA3 requires less CPU time than GA2 since it does not perform the GA operation on the job sequence chromosome in every generation. The results also indicate that GA2_2 and GA3_2 are more effective than GA2_1 and GA3_1, respectively. Hence, the WSPT job sequence initial population is applied to the rest of the test problems.

For large problem sizes, 30-job problems are used to investigate the performances of three GA heuristics (GA1, GA2_2, and GA3_2). Table 5 summarizes the results of 800 test problems (50 instances for each experiment). All three GA heuristics utilize the adaptive $P_m = 0.8$ with $D = 0.99$, $R_m = 0.95$, a population size of 30, and 3000 generations. In addition, GA3_2 utilizes $R_{pms} = 30$. The results show that GA3_2 yields the best (lowest) average objective function value. Compared among the solutions of three heuristics, GA3_2 obtains the best solution in 59.75% and the worst solution in 13.63% of tested instances. Overall, we believe that GA3_2 heuristic performs the best among all proposed GA heuristics.

Since the computational time increases in a non-polynomial fashion, 8-job problems are the largest that can be solved by TE in a reasonable amount of time. Similarly, 19-job problems are the upper limit for utilizing H. Therefore, 1608-job and 16019-job problems (10 problem instances per experiment) are used to perform a final demonstration of the effectiveness of GA3_2. A population size of 303,000 generations and $R_{pms} = 30$ are used in all problems. The results are summarized in Tables 6 and 7. In the 8-job problems, GA3_2 provides positive savings of TWECT compared to H. GA3_2 obtains the optimal solution in all problems and uses much less CPU time than TE. In the 19-job problems, GA3_2 obtains the same solution as or a better solution than H in more than 86% of tested problems. GA3_2 yields a positive savings of TWECT compared to H and requires less CPU time.

5. Conclusions

In this study, the genetic algorithm procedure is successfully applied to the integrated optimization model for production scheduling and preventive maintenance planning proposed by Cassady and Kutanoglu [8]. Three GA-based algorithms are developed. Their performance is evaluated using multiple instances of small, medium, and large size problems. Based on the results, we conclude that the proposed genetic algorithms can be used to effectively solve the integrated problem. Future work includes the development of similar GA-based heuristics to solve extensions of this problem such as due-date related objective functions and multiple machine systems.

Acknowledgement

The authors thank the anonymous referees andthe members of the Editorial Board for the comments and insights. Their feedback led to significant improvements to the content and presentation of this work.

References

[1] J.F. Shapiro, Mathematical programming models and methods for production planning and scheduling, in: S.C. Graves, A.H.G. Rinnooy Kan, P.G. Zipkin (Eds.), Handbooks in Operations Research and Management Science, Logistics of Production and Inventory, vol. 4, North-Holland, Amsterdam, 1993, pp. 371–439.
[2] M. Pinedo, Scheduling: Theory, Algorithms and Systems, Prentice-Hall, Englewood Cliffs, 1995.
[3] Y.S. Sherif, M.L. Smith, Optimal maintenance models for systems subject to failure – a review, Naval Research Logistics Quarterly 28 (1981) 47–74.
[4] R. Dekker, Application of maintenance optimization models: a review and analysis, Reliability Engineering and System Safety 51 (1996) 229–240.
[5] G.H. Graves, C.Y. Lee, Scheduling maintenance and semiresumable jobs on a single machine, Naval Research Logistics 46 (1999) 845–863.

[6] C.Y. Lee, Z.L. Chen, Scheduling jobs and maintenance activities on parallel machines, Naval Research Logistics 47 (2000) 145–165.
[7] X. Qi, T. Chen, F. Tu, Scheduling the maintenance on a single machine, Journal of the Operational Research Society 50 (1999) 1071–1078.
[8] C.R. Cassady, E. Kutanoglu, Integrating preventive Maintenance Planning and Production Scheduling for a Single Machine, Working Paper.
[9] L.D. Chambers, The Practical Handbook of Genetic Algorithms: Applications, CRC Press, Boca Raton, 2001.
[10] M.C. Gupta, Y.P. Gupta, A. Kumar, Minimizing flow time variance in a single machine system using genetic algorithms, European Journal of Operational Research 70 (1993) 289–303.
[11] J. Liu, L. Tang, A modified genetic algorithm for single machine scheduling, Computers and Industrial Engineering 37 (1999) 43–46.
[12] C.R. Reeves, Genetic algorithms for the operations researcher, INFORMS Journal on Computing 9 (1997) 231–250.
[13] C.R. Reeves, Genetic algorithms, in: C.R. Reeves (Ed.), Modern Heuristic Techniques for Combinatorial Problems, Wiley, New York, 1993, pp. 151–188.
[14] C.R. Reeves, A genetic algorithm for flowshop sequencing, Computers and Operations Research 22 (1995) 5–13.

Navadon Sortrakul is an optimization engineer team leader at Transplace, Inc. Before his current position, he has worked at J.B. Hunt Logistics, Inc. He received his PhD in Industrial Engineering from the University of Arkansas, his MS in Industrial Engineering from the Iowa State University, and his BS in Industrial Engineering from the Chulalongkorn University, Thailand. His primary research interests are in applied operations research in production scheduling, preventive maintenance optimization, and logistics. He is a member of IIE, INFORMS, and Phi Kappa Phi.

Heather Nachtmann is an assistant professor of Industrial Engineering at the University of Arkansas. She joined the UofA faculty in 2000. She received her PhD in Industrial Engineering from the University of Pittsburgh. Her research interests include economic decision analysis, engineering valuation, and applied operations research. Her primary research applications are intermodal transportation networks and logistic systems. She holds leadership positions in ASEE and IIE. She is a member of AACE International, ASEM, and INFORMS.

C. Richard Cassady is an associate professor in the Department of Industrial Engineering at the University of Arkansas. Prior to joining the faculty at the UofA, he served on the industrial engineering faculty at Mississippi State University. He received his BS summa cum laude, MS and PhD, all in industrial and systems engineering, from Virginia Tech. His primary research interests are in repairable systems modeling, including the evaluation and optimization of equipment maintenance policies. He also conducts research in reliability engineering, statistical quality control and applied operations research. He is a Senior Member of IIE and a Member of ASEE, ASQ, INFORMS and SRE. He is also a member of the RAMS Management Committee.