A BRKGA for the Integrated Scheduling Problem in FMSs

S. Mahdi Homayouni
LIAAD, INESC TEC,
Porto, Portugal.
smh@inesctec.pt

Dalila B.M.M. Fontes
LIAAD, INESC TEC, & Faculdade de Economia, Universidade do Porto,
Porto, Portugal.
dfontes@inesctec.pt

Fernando A.C.C. Fontes
SYSTEC - ISR, & Faculdade de Engenharia, Universidade do Porto,
Porto, Portugal.
faf@fe.up.pt

ABSTRACT

This work proposes a biased random key genetic algorithm (BRKGA) for the integrated scheduling of manufacturing, transport, and storage/retrieval operations in flexible manufacturing systems (FMSs). Only recently, research on this problem has been reported; however, no heuristic approaches have yet been reported. The computational results show the BRKGA to be capable of finding good quality solutions quickly.

CCS CONCEPTS

• **Computing methodologies** → **Planning for deterministic actions**; • **Applied computing** → **Industry and manufacturing**; **Computer-aided manufacturing**;

KEYWORDS

Integrated scheduling, Storage/retrieval operations, Biased random key genetic algorithm.

ACM Reference Format:
S. Mahdi Homayouni, Dalila B.M.M. Fontes, and Fernando A.C.C. Fontes. 2019. A BRKGA for the Integrated Scheduling Problem in FMSs. In *Genetic and Evolutionary Computation Conference Companion (GECCO '19 Companion), July 13–17, 2019, Prague, Czech Republic.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3319619.3321933

1 INTRODUCTION

In an FMS environment a set of multipurpose machines, a set of automated guided vehicles (AGVs), and an automated storage/retrieval system (AS/RS) cooperate under the control of a central computer system. In such systems, a part (job) stored in a cell of the AS/RS is retrieved by the shuttle and taken to the load/unload (LU) area, from where an AGV takes it to the machine processing its first operation. (A job consists of a set of manufacturing operations.) When a machine finishes processing an operation, an AGV takes the job to the machine processing the next one or to the LU area, if it is its last operation. Once the job is back at the LU area, the shuttle transports and delivers it to the corresponding storage cell. Since manufacturing operations, transport tasks, and storage/retrieval (s/r) are interrelated they need to be scheduled simultaneously.

GECCO '19 Companion, July 13–17, 2019, Prague, Czech Republic

ACM ISBN 978-1-4503-6748-6/19/07…$15.00
https://doi.org/10.1145/3319619.3321933

Jawahar et al. [7] address storage allocation within FMSs considering that machine processing time includes transportation time; thus assuming an unlimited number of AGVs. Chetty and Reddy [3] develop a two-stages approach that first solves the job scheduling problem and then schedules the needed shuttle tasks. Gnanavelbabu et al. [4] determine the sequence of jobs (rather than operations) both for machines and AGVs. An AGV is dedicated to a job until its completion and then returned to the LU to be allocated to the next job. The sequence of s/r operations is determined afterwards.

Previous approaches address the problem components separately, which may lead to sub-optimal solutions or even infeasible ones. As far as we are aware of, the only work addressing the integrated scheduling of machines, AGVs, and s/r operations is that of Homayouni and Fontes [6], which proposes a mixed integer linear programming (MILP) model. However, heuristic approaches are yet to be proposed. Thus, our main contribution is the development of a heuristic approach, a biased random key genetic algorithm (BRKGA), capable of quickly finding good quality solutions for the integrated scheduling problem.

2 PROBLEM DESCRIPTION

In the manufacturing area J independent jobs are to be processed, each job comprises n_j, $j \in J$, ordered operations. Each operation is characterized by a processing machine and a processing time. For each job, there are potentially $n_j - 1$ transport tasks between machines and two between the LU and the machines processing the first and last operations of the job. The transport tasks are performed by identical AGVs carrying one job at the time with known travelling times.

Each job is stored at a known storage cell of the AS/RS and is returned to it once all its operations have been completed. The AS/RS is comprised of two racks of storage cells and a shuttle moves along the aisle between the racks to pick up and drop off the jobs. As in [7] the shuttle travelling time is calculated using the Chebyshev distance between two storage cells or between a storage cell and the LU area. We assume a sufficiently large buffer at both the machines and the LU area, where jobs may wait if necessary.

3 THE PROPOSED BRKGA

Random keys genetic algorithms (RKGAs) were originally proposed by Bean [1] specifically for problems involving sequencing. RKGAs represent a solution to the problem as a vector of real numbers in the interval $[0, 1]$, which needs to be decoded into a solution to the original problem. The main advantages of using random keys are: i) high locality and heritability, ii) relative importance between operations, and iii) ensuring solution feasibility. The BRKGA proposed here uses the framework of [5], which can be applied to a wide range of problems, e.g. [8], as it searches the solution space of the optimization problem indirectly by searching the continuous n-dimensional unit hypercube. Solutions in the hypercube are mapped onto solutions to the optimization problem, where the fitness is evaluated, by a decoder. Thus, to specify a BRKGA we only need to define the chromosome representation and the decoder.

For the problem being addressed a chromosome has two parts: the operations sequence (part X) with $N + 2J$ genes and the vehicle assignment (part Y) with $N + J$ genes, where $N = \sum_{j=1}^{J} n_j$. To decode part X, we use the smallest position value (SPV) rule [1] to sort the vector of random keys. Then, we convert the indices of the sorted random keys into job numbers. Finally, job numbers are translated into operations by associating the first job appearance with its first operation, the second job appearance with its second operation, and so on. For part Y, the interval $[0, 1]$ is equally divided into V sub-intervals, where V is the number of available AGVs. A random key in $\left[\frac{i-1}{V}, \frac{i}{V}\right]$ is translated into AGV i, with $i = 1, \ldots, V$. From left to right in part Y, the specified AGV is assigned to transport the job for the corresponding operation in part X, except for the first operation of each job, since it corresponds to a retrieval operation. Figure 1 illustrates a chromosome and its decoding for an instance with two jobs, each with two manufacturing operations, and two AGVs.

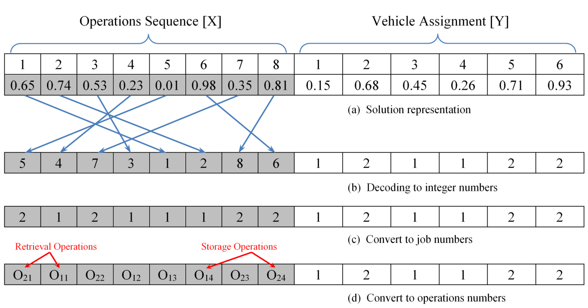

Figure 1: A sample solution and its decoding procedure.

Following on the framework proposed in [5], the initial population is randomly generated. Once the fitness of each individual of a generation, say k, has been calculated, the population is partitioned into a small group of elite solutions and the rest of population. The population of generation $k + 1$ is obtained by: i) copying the elite solutions, ensuring monotonically improvement; ii) randomly generating a small number of new solutions (mutants), avoiding local optima; and iii) producing the rest of the population through mating. Bias is introduced in the selection process as it selects one parent from the elite group and in the crossover as the probability of each allele being inherited from the elite parent is larger.

4 RESULTS AND CONCLUSIONS

Computational experiments were carried out on a modified version of the 10 instances proposed in [2], each having four to eight jobs and 13 to 21 operations, using the first layout. We adapted these instances to our problem by incorporation an AS/RS served by a single shuttle and has two storage racks with 100 storage cells each. The storage-job allocation was randomly generated (data can be found in https://fastmanufacturingproject.wordpress.com).

Table 1 reports the optimal makespan (C^*_{max}) and corresponding CPU time (seconds), obtained by solving the MILP model [6]. For the BRKGA, it reports the best makespan (C_{max}), the optimality gap ($GAP = \frac{C_{max} - C^*_{max}}{C^*_{max}} (\%)$), and the average optimality gap ($\overline{GAP} = \frac{mean - C^*_{max}}{C^*_{max}} (\%)$), percentage makespan standard deviation ($\sigma\%$), and average CPU time (seconds) over 15 runs.

Table 1: Computational results.

Instances		MILP [6]		BRKGA				
Name	M-J-O	C^*_{max}	Time	C_{max}	GAP	$\overline{GAP}$	$\sigma\%$	Time
EX11	4-5-13	203	6.5	**203**	0.00	2.73	5.25	4.96
EX51	4-5-13	207	7.7	**207**	0.00	4.64	5.60	4.63
EX91	4-5-17	296	7.7	**296**	0.00	4.80	7.42	7.43
EX41	4-5-19	224	15.8	230	2.68	6.70	6.64	9.85
EX21	4-6-15	270	92.6	**270**	0.00	3.68	8.79	7.85
EX31	4-6-16	226	116.6	230	1.77	6.19	5.46	8.41
EX61	4-6-18	258	172.8	**258**	0.00	4.83	6.90	10.57
EX81	4-6-20	247	137.8	251	1.62	6.34	6.97	13.36
EX101	4-6-21	298	221.3	300	0.67	4.38	8.63	16.57
EX71	4-8-19	286*	23219	286	0.00	5.76	7.75	16.61
Average					0.67	5.00	6.94	10.03

M - number of machines, J - number of jobs, and O - and number of operations

Results in Table 1 show that the BRKGA obtains the optimal C_{max} for five of the nine instances with a known optimal one, always in less time than the MILP. In addition, for the instance EX71 for which the MILP was not able to find an optimal solution within its 50 000 seconds limit, the BRKGA matched the best known C_{max}. Furthermore, for the remaining instances the gap ranges from 0.67% to 2.68%. Finally, the average makespan and its standard deviation over 15 runs range from 2.73% to 6.70% and from 5.25% to 8.79%, respectively, which allows for inferring the BRKGA robustness.

The proposed BRKGA, in addition to being novel, is capable of quickly finding good solutions for small-sized problem instances (its CPU time is always below 17 seconds). This work is particularly relevant as no heuristic approaches exist for the joint production, transport, and s/r scheduling in FMSs, which is a NP-hard problem.

ACKNOWLEDGMENTS

This work is financed by FEDER/COMPETE2020/ NORTE2020/FCT through grants NORTE-01-0145-FEDER-000020, POCI-01-0145-FEDER 031821 and 031447, and PTDC/EEI-AUT/2933/2014.

REFERENCES

[1] JC Bean. 1993. Genetics and random keys for sequencing amd optimization. (1993).
[2] Ü Bilge and G Ulusoy. 1995. A time window approach to simultaneous scheduling of machines and material handling system in an FMS. *Operations Research* 43, 6 (1995), 1058–1070.
[3] OVK Chetty and MS Reddy. 2003. Genetic algorithms for studies on AS/RS integrated with machines. *The International Journal of Advanced Manufacturing Technology* 22, 11-12 (2003), 932–940.
[4] A Gnanavelbabu, J Jerald, A Noorul Haq, and P Asokan. 2009. Multi objective scheduling of jobs, AGVs and AS/RS in FMS using artificial immune system. In *Proc. of National conference on Emerging trends in Engineering and Sciences.* 229–239.
[5] JF Gonçalves and MGC Resende. 2011. Biased random-key genetic algorithms for combinatorial optimization. *Journal of Heuristics* 17, 5 (2011), 487–525.
[6] SM Homayouni and DBMM Fontes. 2017. Integrated Scheduling of Machines , Vehicles , and Storage Tasks in Flexible Manufacturing Systems. In *MISTA 2017, Kuala Lumpur, Malaysia.* 5–8.
[7] N Jawahar, P Aravindan, and SG Ponnambalam. 1998. Optimal random storage allocation for an AS/RS in an FMS. *The International Journal of Advanced Manufacturing Technology* 14, 2 (1998), 116–132.
[8] LAC Roque, DBMM Fontes, and FACC Fontes. 2011. A biased random key genetic algorithm approach for unit commitment problem. In *Lecture Notes in Computer Science*, Vol. 6630. Springer, 327–339.