ENSEMBLE POPULATION BASED DISCRETE HARMONY SEARCH ALGORITHM FOR THE JOB SHOP SCHEDULING PROBLEM

Lubos Matejcik and Donald Davendra,
Department of Computer Science
Faculty of Electrical Engineering and Computer Science
VŠB-Technical University of Ostrava
17. listopadu 15, 708 33 Ostrava-Poruba
Czech Republic.
Email: {lubos.matejcik.st,donald.davendra}@vsb.cz

KEYWORDS

Ensemble population, Harmony Search algorithm, Job shop scheduling

ABSTRACT

A novel ensemble population based Harmony Search (HS) algorithm based on different operating parameters is proposed in this paper. A sub-population based HS based on differing scaled pitch adjusting rate and bandwidth is devised, with the objective of searching different search space in the fitness landscape. The algorithm is tested on the job shop scheduling problem and favourably compared with the reported optimal values in literature.

INTRODUCTION

The Harmony Search (HS) algorithm, proposed by (Geem et al., 2001), uses musical phenomena such as pitch and tune to emulate a evolutionary algorithm. The objective then, as in any orchestra, is to obtain a perfect pitch; an optimal solution. This algorithm has gained traction due to the inordinate amount of scientific research into different formulation and applications (Al-Betar et al., 2015).

One of the recent trends in evolutionary algorithm design has been the ensemble population structure. The canonical algorithms generally had a singular population, where its size was generally problem dependent. This provided the algorithm with a large diversity of selectable "gene pool" materials. While this is a valid strategy, generally the extent of interaction of the different individuals in the population and the need to obtain optimal "operating" parameters of the algorithm, in order to maximise the interaction of the individuals in the population, has led to the ensemble approach.

The ensemble approach, is not a novel concept, having first appeared in the form of the "islands model" (Corcoran and Wainwright, 1994), where the population is divided into sub-populations, which each sub-population tasked with a different objective. These sub-populations were combined into a structure, such as ring, star or random, where the exchange of information regarding the population would then occur. Periodically, the islands interact using a migration process which is responsible for sending and receiving certain individuals across islands controlled by migration *rate* and migration *frequency*. If there is no migration, an island model is nothing more than a set of separate runs and thus migration is very important (Al-Betar et al., 2015).

Recent research trends have taken different algorithms as ensemble (Tasgetiren et al., 2010), different constraints within the objective functions as an ensemble approach (Mallipeddi and Suganthan, 2010) alongside niching algorithms (Yu and Suganthan, 2010) and sub-populations (Pan et al., 2010). Harmony search has also been applied using the "islands model" (Al-Betar et al., 2015), where the researchers have conducted an islands model based on the concept of (Pan et al., 2010). The main idea of this approach is the convergence behaviour of each subpopulation applied to real-domain problems.

This research takes a different track to the aforementioned paper, as the main onus here is the tuning of parameters for different sub-populations applied to a combinatorial optimisation problem. HS is perturbed by changing the operating parameters independently for each sub-population, thereby the search space occupied by the sub-groups is generally non-overlapping, with a lower level of stagnation.

The paper is organised as follows: section 1 introduces the HS algorithm and section 2 introduces the ensemble HS approach. Section 3 described the job shop scheduling problem. The experiment is given in section 4 and the research is concluded in section 5.

1 HARMONY SEARCH

HS is meta-heuristic algorithm developed by Geem (Geem et al., 2001) and was originally inspired by the improvisation process of Jazz musicians. A musical instrument improvisation corresponds to a decision variable in optimization. Its pitch range corresponds to a value range. HS is good at identifying the high performance regions of the solution space at a reasonable time. HS algorithm uses stochastic random searches, so derivative is unnecessary.

Proceedings 29th European Conference on Modelling and Simulation ©ECMS Valeri M. Mladenov, Petia Georgieva, Grisha Spasov, Galidiya Petrova (Editors)
ISBN: 978-0-9932440-0-1 / ISBN: 978-0-9932440-1-8 (CD)

HS algorithm contains a number of operating parameters, including harmony memory size (HMS), harmony memory considering rate (HMCR), pitch adjusting rate (PAR), number of improvisation (NI) and distance bandwidth (BW).

$$\left[\begin{array}{cccc|c} x_1^1 & x_2^1 & \dots & x_D^1 & f(X^1) \\ x_1^2 & x_2^2 & \dots & x_D^2 & f(X^2) \\ \vdots & \dots & \dots & \dots & \vdots \\ x_1^{HMS} & x_2^{HMS} & \dots & x_D^{HMS} & f(X^{HMS}) \end{array}\right]$$

HS uses memory storage, named harmony memory (HM), which consists of HMS harmony vectors, where HMS is the harmony memory size parameter. The harmony vectors and corresponding objective function values are stored in the harmony memory (HM) expressed as a matrix.

In the HS algorithm, each solution is called a "harmony" and represented by a *D*-dimension vector. The initial population of harmony vectors are randomly generated and stored in HM.

$$x_i^{new} \leftarrow \begin{cases} x_i(k) \in \{x_i(1), x_i(2), ..., x_i(k)\} & r_1 > HMCR \\ x_i(k) \in \{x_i^1, x_i^2, ..., x_i^{HMS}\} & r_1 \leq HMCR \\ x_i(k) \in \{x_i(k) + r_3 * BW\} & r_1 \leq PAR \end{cases}$$

In each improvisation, a new harmony is improvised, and if it is better than the worst vector in the harmony vector, the worst is replaced by the new vector. A new harmony vector x^{new} is improvised by applying three rules: a memory consideration, a pitch adjustment and random selection. r_1 is random number generated in the range [0, 1]. If r_1 is less than HMCR, the decision variable $x^{new}(k)$ is chosen from harmony memory; otherwise, $x^{new}(k)$ is obtained by a random re-initialization between the search bounds. Each new harmony vector can be adjusted by perturbance with probability of PAR.

PAR and BW in HS algorithm are very important parameters in fine-tuning of optimized solution vectors. The traditional HS algorithm uses fixed value for both PAR and BW - PAR and BW values adjusted in the initialization step and cannot be changed during new improvisations. Improvement here is to dynamically change these two parameters according to the improvisation number. Small BW values in final improvisations increase the fine-tuning of solution vectors, but in early improvisations BW must take a bigger value to enforce the algorithm to increase the diversity of solution vectors. Large PAR values with small BW values usually cause the improvement of best solutions in final generations which causes the algorithm to converge to the optimal solution vector. Therefore, we propose to change parameters PAR and BW as given in Equations (1), (2) and (3).

$$BW(i) = BW_{max} * \exp(c * i), \tag{1}$$

$$c = \frac{\ln\left(\frac{BW_{min}}{BW_{max}}\right)}{NI} \tag{2}$$

$$PAR(i) = PAR_{min} + \frac{(PAR_{max} - PAR_{min})}{NI} * i \tag{3}$$

where $i \in \{1, 2, \dots, NI\}$, with NI being the number of total improvisations.

```
initialize parameters
for p = 1 to n do
    for i = 1 to HMS do
        randomly generate harmony vector X_p^i
        calculate f(X_p^i)
    end
end
for imp = 1 to NI do
    for p = 1 to n do
        PAR_p = PAR(imp, p)
        BW_p = BW(imp, p)
        if r_1 < HMCR_p then
            X^new = X_p^{r_2}
            if r_2 < PAR_p then
                perturbate X^new of BW_p
            end
        end
        randomly generate X^new
        calculate f(X^new)
        if f(X^new) is better than f(X_p^worst) then
            replace X_p^worst by X^new
        end
    end
end
```

Algorithm 1: Harmony Search algorithm

2 ENSEMBLE POPULATION APPROACH

Obtaining the optimal parameter values is a complicated process, and generally empirically driven. One of approach as to how to find good parameter values is tuning, but this can be applied only to specific problem instances. Basically, parameter tuning leads to a next subsequent optimization task. After several tunings with different parameter values, we can obtain sufficient good values.

Ensemble population approach run algorithm with different parameter values in each population as shown in Figure 2. Convergence rates in populations are not the same and individuals which can lead to better solutions are not abandoned due to different parameter settings.

Tuned parameter values obtained by stochastic method for the problem instance of ft06 is given in Table 1. These parameter values were subsequently used for all experimentations in this paper.

3 JOB SHOP SCHEDULING

A job shop scheduling problem (JSP) is a problem where the *route* of the job is fixed, however not necessarily the same for each job. If a job has to visit certain machines more than once, the job is said to recirculate (Pinedo, 1995). The problem designation is

$$Jm \parallel C_{\max}$$

The JSP can be described by a set of n jobs $\{J_i\}_{1 \leq j \leq n}$ which is to be processed on a set of m machines $\{M_r\}_{1 \leq r \leq m}$. The problem can be characterized as follows:

1. Each job must be processed on each machine in the order given in a pre-defined technological sequence of machines.

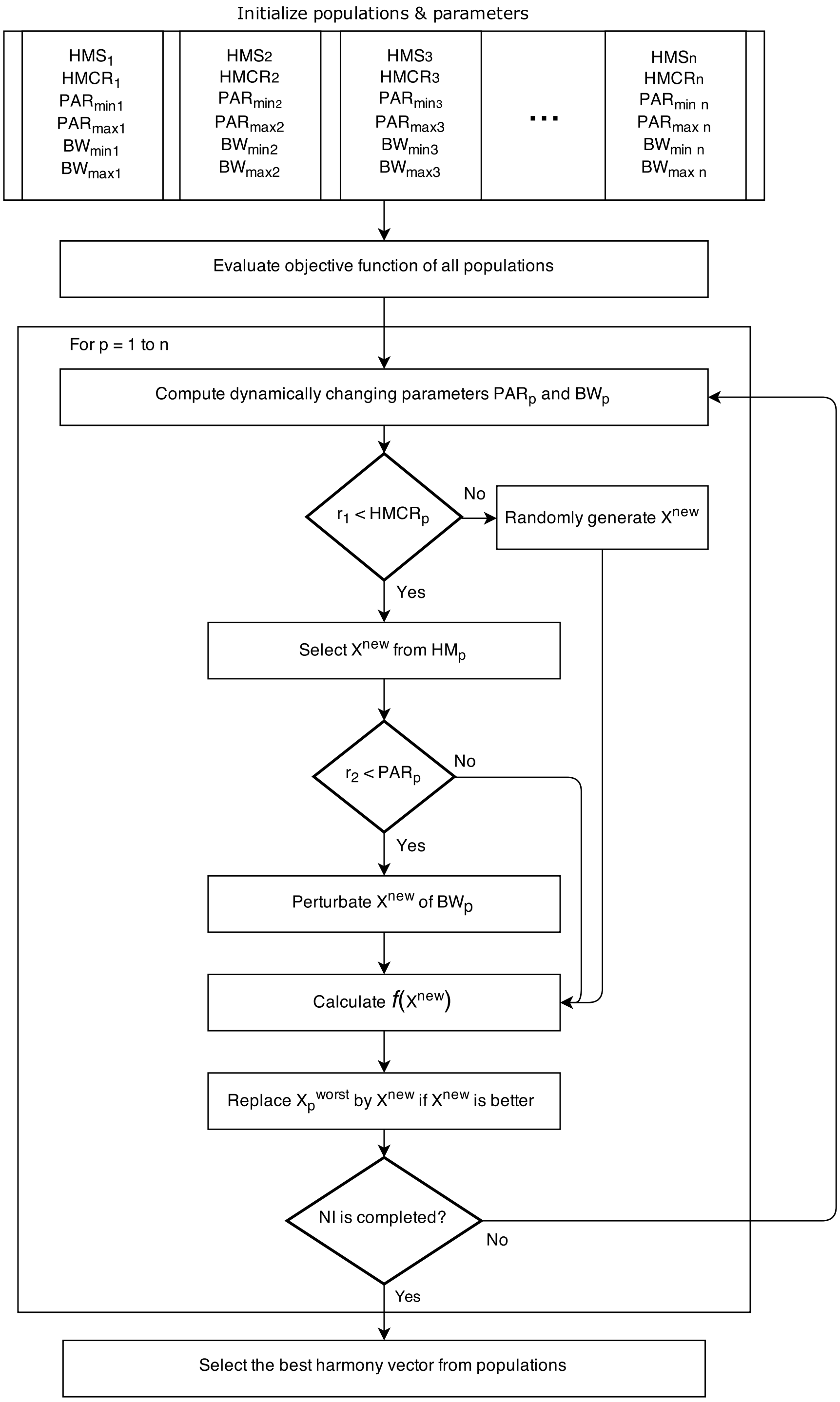

Figure 1: Flow diagram

Table 1: Tuned parameter values for HS with 4 populations for ft06 instance (S is schedule size)

HMS_1	13	$HMCR_1$	0.42	PAR_1	$\langle 0.4, 0.8\rangle$	BW_1	$\langle 1, 2 \cdot S\rangle$
HMS_2	14	$HMCR_2$	0.75	PAR_2	$\langle 0.5, 0.99\rangle$	BW_2	$\langle 2, 3 \cdot S\rangle$
HMS_3	3	$HMCR_3$	0.99	PAR_3	$\langle 0.1, 0.6\rangle$	BW_3	$\langle 5, 4 \cdot S\rangle$
HMS_4	7	$HMCR_4$	0.147	PAR_4	$\langle 0.3, 0.7\rangle$	BW_4	$\langle 1, 2 \cdot S\rangle$

2. Each machine can process only one job at a time.
3. The processing of job J_j on machine M_r is called the operation O_{jr}.
4. Operation O_{jr} requires the exclusive use of M_r for an uninterrupted duration p_{jr}, its processing time; the preemption is not allowed.
5. The starting time and the completion time of an operation O_{jr} is denoted as s_{jr} and c_{jr} respectively. A schedule is a set of completion times for each operation $\{c_{jr}\}_{1\leq j\leq n, 1\leq r\leq m}$ that satisfies above constraints.
6. The time required to complete all the jobs is called the *makespan*, which is denoted as $C_{\max}$. By definition, $C_{\max} = \max_{1\leq j\leq n, 1\leq r\leq m} c_{jr}$.

The problem is "general", in the sense that the technological sequence of machines can be different for each job as implied in the first condition and that the order of jobs to be processed on a machine can be also different for each machine. The predefined technological sequence of each job can be given collectively as a matrix $\{T_{jk}\}$ in which $T_{jk} = r$ corresponds to the k-th operation O_{jr} of job J_i on machine M_r. The objective of optimizing the problem is to find a schedule that minimizes $C_{\max}$ (Yamada, 2003).

4 EXPERIMENTATION

The experimentation was conducted on ten selected problem from the Operations Research library (Beasley, 2009) and Taillard's scheduling instances (Taillard, 2015) with sizes from ten jobs and ten machines for the abz5, abz6, ft10 and la16 instances, ten jobs and five machines for the la01, la02, la03, la04 and la05 instances, six jobs and six machines for the ft06 instance and larger instances from the Taillard's sets.

Thirty experiments were conducted on each instance to get statistical variance. The results are presented in Table 2 and Table 3. Four parameters are of interest, the best makespan obtained, the average makespan from 30 runs of the experiment, the standard deviation of the makespan of these 30 runs and the average time.

From the obtained results, for four instances of ft06, la01, la04 and la05, the optimal result was obtained by EPHS. Additionally, the average value for these instances is also the optimal and the standard deviation is zero. For the other instances, the highest standard deviation is 5.584 for the ft10 instance and the lowest is 0.456 for the la02 instance. The lowest average time is 2773.33 milliseconds for the ft06 instance and 8410.13 milliseconds for the ft10 instance. The cumulative average standard deviation is 1.495 and time is 5304.48 milliseconds for all instances.

The percentage relative difference (PRD) between the optimal and the $EPHS_{best}$ can be computed as:

$$PRD = \frac{(Optimal - EPHS_{best})}{Optimal} \cdot 100$$

and the result is presented in Table 4. The highest PRD is for the ft10 instance at -5.913, and the minimum is -0.458 for the la02 instance. The average PRD is -1.553 for all instances.

For the Taillard instance, the PRD is taken between the lower bound and $EPHS_{best}$ and is given in Table 5. The differences are from -18.2 to -60.8. The average PRD for the Taillard's instances is -29.32.

5 CONCLUSION

A ensemble population based HS algorithm based on different parameter values is proposed in this paper to solve the job shop scheduling problem. Based on the "islands" model, with non-interactive, or no-exchange topology, concentrated sub-populations are generated randomly and then driven with different tuned parameters of evolving pitch adjusting rate and bandwidth.

The new algorithm (EPHS) is tested on 19 different instances of the job shop scheduling problem based on two classifications. In the first set, out of the 10, EPHS obtains the optimal value for four instances and the average PRD of all instances is -1.553, with the highest being -3.605 for the abz6 instance. In the second set, the best instance is 50 x 20 and the worst is 20 x 20, with average being -29.32. Form the second experiment, we confer that neighbourhood search techniques would be required inside HS in order to sample the best regions and also to utilise some good starting seed solutions using NEH, which could provide a catalyst for initial exploration.

Some of the unique features of the algorithm is the separate ensemble population evolution, a wide mapping of operating parameters and an exploitation of constrained search space, without penalisation of "inferior" solutions. These attributes improve the general behaviour of the HS algorithm.

Future direction of this research is its application to different constraints within the job shop problem such as tardiness, local search routines and the parallel application of the algorithm using OpenMP and CUDA to speed up the execution time.

ACKNOWLEDGEMENT

This work was principally supported by the Grant of SGS SP2015/141 and Grant Agency of the Czech Republic - GAČR P103/15/06700S.

REFERENCES

Al-Betar, M. A., Awadallah, M. A., Khader, A. T., and Abdalkareem, Z. A. (2015). Island-based harmony search for optimization problems. *Expert Systems with Applications*, 42(4):2026 – 2035.

Beasley, J. (2009). Operations reserach library. `http://people.brunel.ac.uk/~mastjjb/jeb/info.htm`.

Table 2: Summarised results

Instances	n x m	Optimal	$EPHS_{best}$	$EPHS_{avg}$	Std dev	Avg time
abz5	10 x 10	**1234**	1242	1329.85	0.644	7599.067
abz6	10 x 10	**943**	977	1010.8	3.480	7553.133
ft06	6 x 6	**55**	**55**	55	0.000	2773.333
ft10	10 x 10	**930**	985	1069.37	5.584	8410.133
la01	10 x 5	**666**	**666**	669	0.000	3823.633
la02	10 x 5	**655**	658	693.926	0.456	3862.367
la03	10 x 5	**597**	608	639.906	1.809	3847.533
la04	10 x 5	**590**	**590**	590	0.000	3807.133
la05	10 x 5	**593**	**593**	593	0.000	3851.633
la16	10 x 10	**945**	974	1021.757	2.977	7526.900

Table 3: Summarised results (Taillard's instances)

n x m	Lower bound	Upper bound	$EPHS_{best}$	$EPHS_{avg}$	Std dev	Avg time
15 x 15	1005	1231	1316	1390.1	18.05	90240
20 x 15	1254	1376	1562	1653.8	25.66	125418
20 x 20	1217	1663	1957	2025.0	8.9	168207
30 x 15	1764	1770	2126	2189.2	16.9	202414
30 x 20	1850	2064	2532	2648.5	30.9	273151
50 x 15	2760	2760	3336	3419.1	23.3	404720
50 x 20	2868	2868	3391	3463,9	40.5	553399
100 x 20	5464	5464	6671	6823.7	12.8	2094680

Table 4: Percentage relative difference

Instances	Optimal	$EPHS_{best}$	PRD
abz5	**1234**	1242	-0.648
abz6	**943**	977	-3.605
ft06	**55**	**55**	0
ft10	**930**	985	-5.913
la01	**666**	**666**	0
la02	**655**	658	-0.458
la03	**597**	608	-1.842
la04	**590**	**590**	0
la05	**593**	**593**	0
la16	**945**	974	-3.068

Table 5: Percentage relative difference (Taillard's instances)

n x m	Lower bound	Upper bound	$EPHS_{best}$	PRD
15 x 15	1005	1231	1316	-30.9
20 x 15	1254	1376	1562	-24.5
20 x 20	1217	1663	1957	-60.8
30 x 15	1764	1770	2126	-20.5
30 x 20	1850	2064	2532	-36.8
50 x 15	2760	2760	3336	-20.8
50 x 20	2868	2868	3391	-18.2
100 x 20	5464	5464	6671	-22.1

Corcoran, A. L. and Wainwright, R. L. (1994). A parallel island model genetic algorithm for the multiprocessor scheduling problem. In *Proceedings of the 1994 ACM Symposium on Applied Computing*, SAC '94, pages 483–487, New York, NY, USA. ACM.

Geem, Z., Kim, J., and Loganathan, G. (2001). A new heuristic optimization algorithm: Harmony search. *Simulation*, 76(2):60–68.

Mallipeddi, R. and Suganthan, P. N. (2010). Ensemble of constraint handling techniques. *IEEE Trans. on Evolutionary Computation*, 14(4):561–579.

Pan, Q.-K., Suganthan, P. N., Liang, J. J., and Tasgetiren, M. F. (2010). A local-best harmony search algorithm with dynamic subpopulations. *Engineering Optimization*, 42(2):101–117.

Pinedo, M. (1995). *Scheduling: theory, algorithms and systems*. Prentice Hall, New Jersey.

Taillard, E. (2015). Taillard job shop instances. `http://mistic.heig-vd.ch/taillard/problemes.dir/ordonnancement.dir/ordonnancement.html`.

Tasgetiren, M. F., Suganthan, P. N., and Pan, Q. K. (2010). An ensemble of discrete differential evolution algorithms for solving the generalized traveling salesman problem. *Applied Mathematics and Computation*, 215(9):3356–3368.

Yamada, T. (2003). *Studies on Metaheuristics for Jobshop and Flowshop Scheduling Problems*. PhD thesis, Kyoto University.

Yu, E. and Suganthan, P. (2010). Ensemble of niching algorithms. *Information Sciences*, 180(15):2815 – 2833.

AUTHOR BIOGRAPHIES

LUBOS MATEJCIK is a graduate student in Informatics at the Computing Science at the Technical University of Ostrava. His email address is `lubos.matejcik.st@vsb.cz`.

DONALD DAVENDRA is an Associate Professor of Computing Science at the Technical University of Ostrava. He has a Ph.D. in Technical Cybernetics from the Tomas Bata University in Zlin. His email address is `donald.davendra@vsb.cz`.