Journal of Fundamental and Applied Sciences

Research Article

ISSN 1112-9867

Special Issue

Available online at http://www.jfas.info

THE IMPLEMENTATION FRAMEWORKS OF META-HEURISTICS HYBRIDIZATION WITH DYNAMIC PARAMETERIZATION

S. Masrom[1,*], A. S. A. Rahman[2], S. Z. Z. Abidin[3], N. Omar[3] and Z. I. Rizman[4]

[1]Faculty of Computer and Mathematical Sciences, UniversitiTeknologi MARA, Perak, Malaysia

[2]Faculty of Science and Information Technology, UniversitiTeknologi PETRONAS, Perak, Malaysia

[3]Faculty of Computer and Mathematical Sciences, UniversitiTeknologi MARA, Shah Alam, Selangor, Malaysia

[4]Faculty of Electrical Engineering, UniversitiTeknologi MARA, Dungun, Terengganu, Malaysia

Published online: 10 November 2017

ABSTRACT

The hybridization of meta-heuristics algorithms has achieved a remarkable improvement from the adaptation of dynamic parameterization. This paper proposes a variety of implementation frameworks for the hybridization of Particle Swarm Optimization (PSO) and Genetic Algorithm (GA) and the dynamic parameterization. In this paper, taxonomy of the PSO-GA with dynamic parameterization is presented to provide a common terminology and meworks are possible to be adapted. Furthermore, different algorithms that used the implementation frameworks with sequential scheme and dynamic parameterizations approaches are tested in solving a facility layout problem.

Author Correspondence, e-mail: suray078@perak.uitm.edu.my

doi: http://dx.doi.org/10.4314/jfas.v9i6s.41

The results present the effectiveness of each tested algorithm in comparison to the single PSO and constant parameterization.

Keywords: hybridization; PSO; GA; implementation frameworks; dynamic parameterization.

1. INTRODUCTION

One promising way to effectively solve optimization problem is by using meta-heuristics algorithms. In this paper, the concern is to introduce a set of implementation frameworks that can be used for the design of meta-heuristics hybridizations involving two well-known meta-heuristics namely Particle Swarm Optimization (PSO)[1-2] and Genetic Algorithm (GA)[3-4]. These two meta-heuristics have gained widespread appeal amongst researchers to solve optimization problems in a variety of application domains.

Research that combine PSO and GA algorithms is very progressing[5-7]. As a kind of meta-heuristics algorithm, PSO and GA hybrid techniques can be generally classified as high-level or low-level hybridization [4-5]. In high-level hybridization (HLH), the components from the different algorithms are not strongly connected because the hybrid algorithms can be independently run from each other. The interaction among the different algorithms can be accomplished through a well-defined interface or protocol [8]. However, the low-level hybridization (LLH) involves original structure modifications of the different algorithms [9-10]. In other words, LLH creates a new algorithm that combines the components from the different hybrid algorithms. These components are strongly interdependent and need to fit well together to solve a particular problem. As a result, the implementation of LLH tends to be more difficult than HLH in terms of the algorithm design. Responding to the difficulty, the attempt of this paper is to introduce a set of simple implementation frameworks that can be used to design a variety of PSO-GA with the LLH techniques. In addition, since research has identified that dynamic parameterization provides benefit for improving the PSO performance[11], the implementation frameworks also been designed to allow different set of dynamic parameterization setting.

2. BACKGROUND OF THE STUDY

2.1. Meta-Heuristics Implementation Framework

In order to establish a common understanding from the implementation variant of meta-heuristics for the purpose of consistency and repeatability experiment [12], many researchers have introduced different implementation frameworks, models or taxonomies. In this research, the frameworks are defined as an abstraction of techniques that are derived based on the proposed taxonomy. Therefore, in this section, the existing works that attempt to generalize and to classify the implementation variant of meta-heuristics in a form of taxonomy, models [36-37] or implementation frameworks are reviewed in accordance to the type of meta-heuristics paradigm and classification elements.

Among the works that has been widely acknowledged is a taxonomy for meta-heuristics hybridization introduced by[13]. In the classification scheme, the researcher has divided meta-heuristics hybridizations according to High-level Teamwork, High-level Relay, Low-level Teamwork and Low-level Relay. The term level is used to describe the combination or cooperation strength among components from the hybrid algorithms. In a more general view, in[14] have divided the hybrid meta-heuristics schemes into three general forms, which are component exchange among meta-heuristics, cooperative search from different meta-heuristics and integration with other methods. The researchers described that the component exchange among meta-heuristics is identical to the LLH while the cooperative search is a kind of HLH. Similarly, in addition to the hybridization level, other types of classification of hybrid meta-heuristics introduced by [9] are hybrid algorithms, order of execution and control strategy. In this taxonomy, major implementation techniques for the HLH are classified by the order of execution and the control strategy. Parallel meta-heuristics are also considered as meta-heuristics hybridization as there is occurrence of cooperation among different meta-heuristics[4, 6]. Several taxonomies for parallel meta-heuristics have been proposed, but they are more significant to HLH. For examples, a taxonomy that categorized the implementation as operation parallelization, search space decomposition and multi-search threads [15-16]. In [17], the researchers have introduced a new taxonomy specific for cooperative search algorithm under the parallel meta-heuristics. The classification is divided according to the algorithm types and the space decomposition.

To summarize, the following Table 1 lists the existing frameworks in relation to the meta-heuristics and its paradigm (Single, LLH, HLH). In addition, it is significant to this paper to review each of the related works with the element of dynamic parameterization.

Table 1. Meta-heuristics implementation framework

Research	Meta-Heuristics	Single	LLH	HLH	Dynamic Parameterization
[9]	All	/	/	/	x
[10]	All	x	/	x	x
[12]	PSO	/	x	x	x
[13]	All	/	/	/	x
[14]	All	/	/	/	x
[15]	All	x	/	x	x
[16]	All	x	/	x	x
[17]	All	x	/	x	/
[18]	Tabu Search	x	/	x	x
[19]	PSO and DE	x	/	/	x
[20]	ACO	/	x	x	x
[21]	EAs	/	/	/	/
[22]	PSO	x	/	/	x
[23]	EAs	/	/	x	/
[24]	EAs	/	x	x	x

As seen in Table 1, the majority of researchwere focused on the HLH rather than HLH for the meta-heuristics hybridization. In the works of [13-14, 9], the researchers have generally classified the meta-heuristics paradigms not to specific for the PSO and GA. The introduction of taxonomy that is specific for PSO and Differential Evolution (DE) in [19] has been found to be a worthwhile for users [34]. Due to the specific meta-heuristics, the taxonomy can be directly used as a tool to analyze the hybridization strategies between the PSO and DE. It is also useful as a reference for designing new optimizers that combines the two meta-heuristics. Furthermore, the specific PSO taxonomy with the HLH and LLH introduced in [22] is only beneficial for the homogeneous implementation. A main drawback of the meta-heuristics

frameworks listed in the Table 1 is the limitation to address dynamic parameterization as one of the implementation elements. In [24] has found that even after many years of research into the Evolutionary Algorithms (EAs), there are no straightforward parameterization guidelines have been provided for the meta-heuristics. The researcher consequently has proposed dynamic parameterization frameworks specific for the EAs.

2.2. Dynamic Parameterization

The achievement of an algorithm like PSO in solving a particular problem domain is always subjected to the suitability of parameters properties, which can be constant or changeable along the iteration of search. Dynamic parameterization refers to the changeable value of parameters, which is derived from different approaches, namely random, time-vary or adaptive. The time-vary approach use iteration number as a main factor of formulation while adaptive approach relies more on the PSO performances indicators such as particle fitness, global fitness and particle position. Details about the time-vary and adaptive formulations used in this papers are described in [25-26] respectively.

3. THE IMPLEMENTATION FRAMEWORKS

The composition of implementation frameworks was created based on a general taxonomy. The taxonomy for LLH of PSO-GA is generally divided into two main groups of elements, namely component and implementation. Fig. 1 shows the taxonomy.

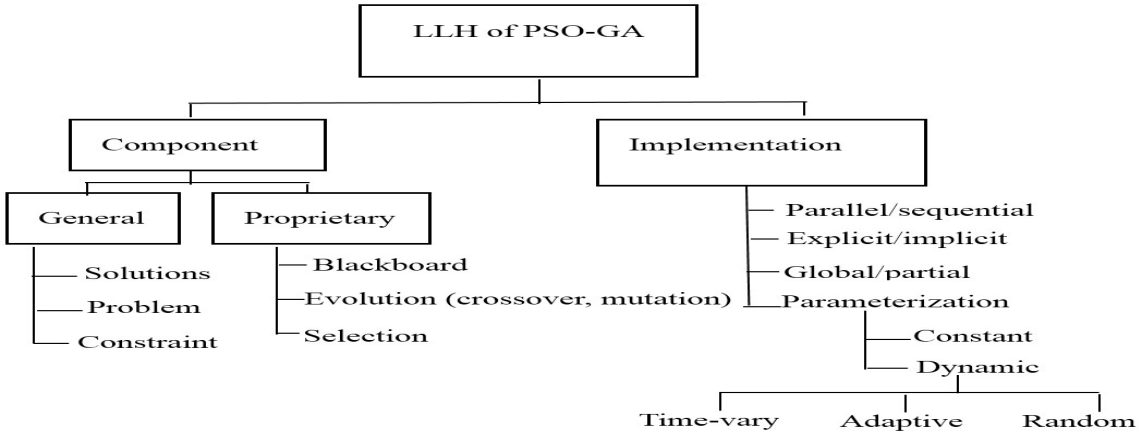

Fig.1. The taxonomy of LLH

3.1. Component

As meta-heuristics, PSO and GA have general and proprietary components. The general components include problem to be solved, a group of solutions for the problem and the solutions constraints. The problem is defined through one or more objective functions while solutions constraints can be derived with constraint functions. The solutions for the problems in the search space are represented according to the algorithm. PSO uses particle for representing solutions while in GA in the form of chromosome.

The PSO proprietary components are based on blackboard type while GA components comprised of evolution and selection categories. The evolution approach uses operators (e.g., mutation and crossover) to reproduce new population of solutions while blackboard type uses shared memory concept to generate new population. PSO uses its shared memory in the form of personal and global best. Selection technique is common to the GA algorithm. It is a process to choose different solutions probabilistically for being crossover usually in proportion to their fitness.

3.2. Implementation

Implementation refers to the execution methods for the LLH components. For example, the solutions component in the search space can be composed into several sub-search spaces which can be explored in parallel or sequential. If the encoding method for the solutions representation is identical in each sub-search spaces, it is categorized as explicit. Otherwise, it is classified as implicit decomposition. Furthermore, each algorithm might solve on global or partial problem. The problem is global if both PSO and GA solve the same optimization problem while partial problem occurs if the problem is different for each algorithm. Fig. 2 shows the elements of implementation.

3.3. Parameterization

Other than the components and its implementation, consideration should be given on parameter configurations of the algorithm. The proposed framework allows dynamic parameterization as described in the section 2.2.

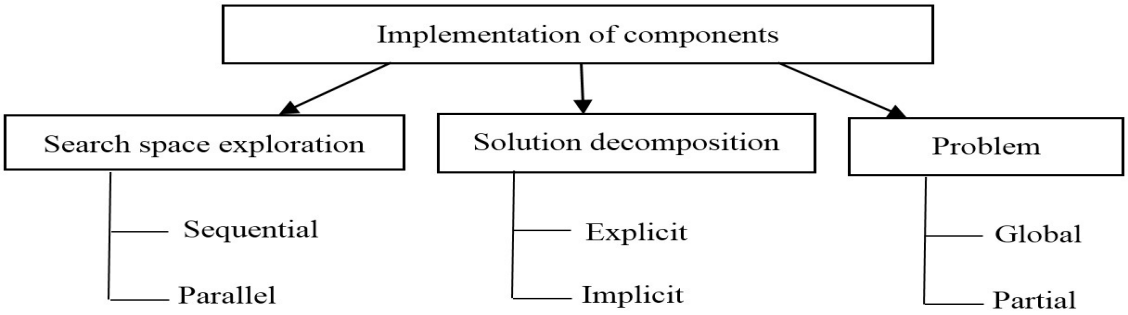

Fig.2. The implementation elements

3.3. The Composition of Implementation Frameworks

This part shows the composition of several implementation frameworks from the previous taxonomy as illustrated in Fig. 3.

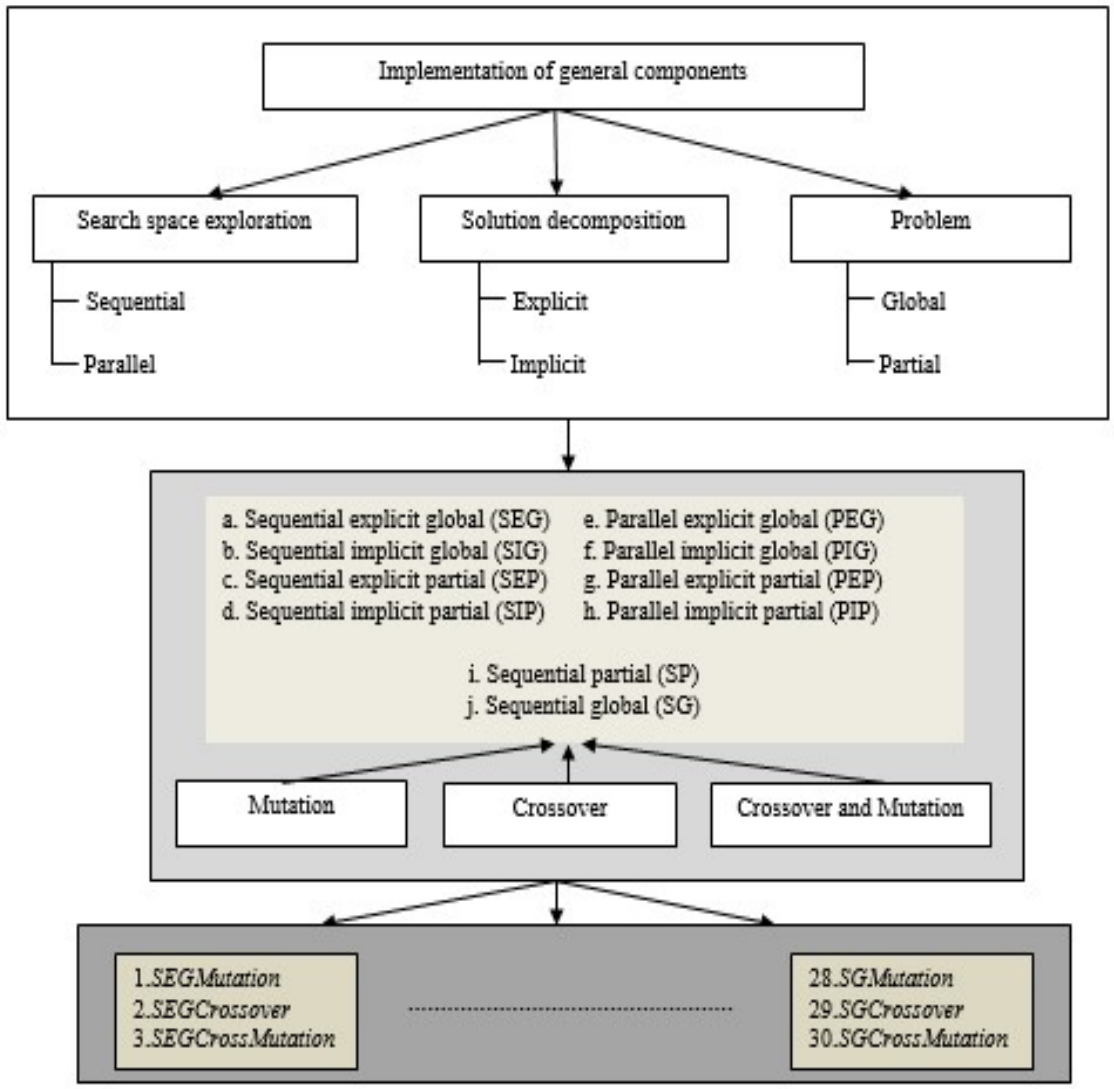

Fig.3. The composition of implementation frameworks

There are 30 implementation frameworks (from SEGMutation to SGCrossMutation) are possible to be prepared. The composition begins with relatively match each general component into each implementation element, which creates ten different LLH schemes (a-j). Each scheme is categorized relatively to search space exploration (parallel or sequential), solution decomposition (explicit or implicit) and problem (global or partial).

Then, each scheme can be adapted with the proprietary components of GA, either with mutation, crossover and the combination as well. In the basic GA paradigm, selection component was used for selecting potential individual for crossover process. Based on the literature, the implementation frameworks with sequential global (SG) scheme is the most widely used in many kinds of optimization problem. Fig. 4 provides general abstraction of the SG scheme.

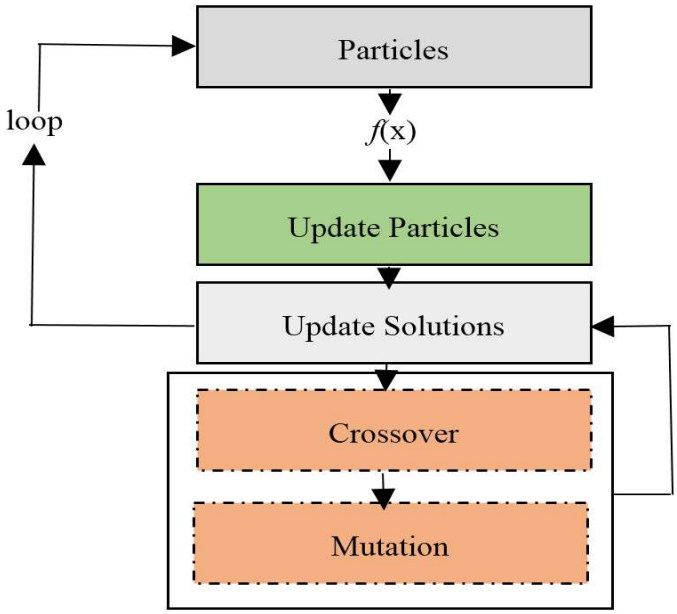

Fig.4. The abstraction of SG scheme

The dotted represent optional to be included in the algorithm. It can be PSO with crossover, PSO with mutation or PSO with both crossover and mutation. The examples of configurations for the SGMutation is presented in the following Fig. 5.

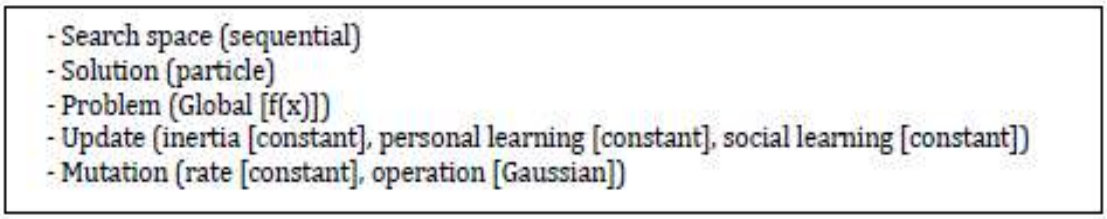

Fig.5. Example of configuration for SGMutation

The search space configuration consists of sequential configuration without percentage of division. Since PSO is master-metaheuristic [35], particle is the only solutions representation in the search space. The following Fig. 6 is the example of configurations for SGCrossover.

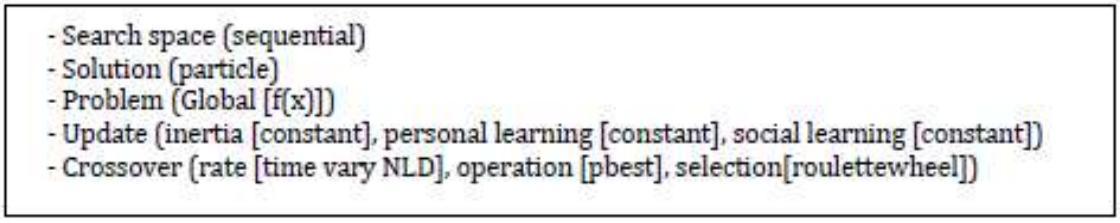

Fig.6. Example of configuration for SGCrossover

Then, the SGCrossoverMutation that includes both crossover and mutation with the SG scheme is drawn in the following Fig. 7.

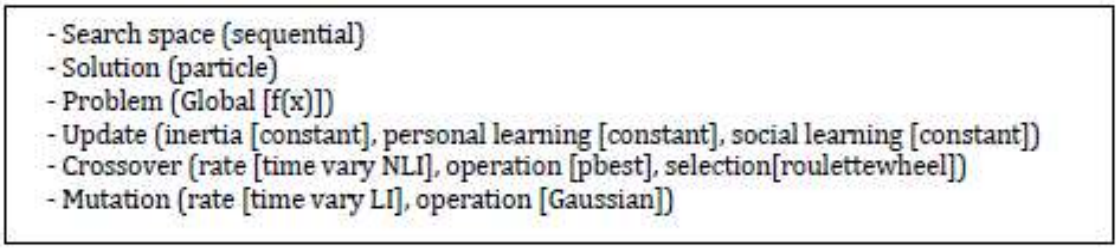

Fig.7. Example of configuration for SGCrossmutation

The crossover between particles are chosen probabilistically in proportion to their fitness. Each position between the first and the maximum are then swapped according to the dynamic crossover probability Cp with adaptive parameterization. Inspired by [27], the periodic crossover was replaced with the adaptive crossover probability. Based on the above configurations, Fig. 8 presents the algorithm.

The threshold value r and r1 are set to random number in the interval [0,1]. It is compared to each particle's probability *Cp* of crossover to decide whether this particle's randomly chosen position d should be altered using *pbest* crossover. As defined in [27], the adjustment to

dimension d is made using an average of two particles' relevant *pbest*values. In the algorithmic listing at Fig. 8, the crossover probability *Cp* and the mutation probability*Mp* of all particles are adapted at line 2 and line 3 respectively. The mutation operation used a Gaussian function that returns a random value within the range of the particle dimension. The α is bounded within 0.1 times of the particle dimension.

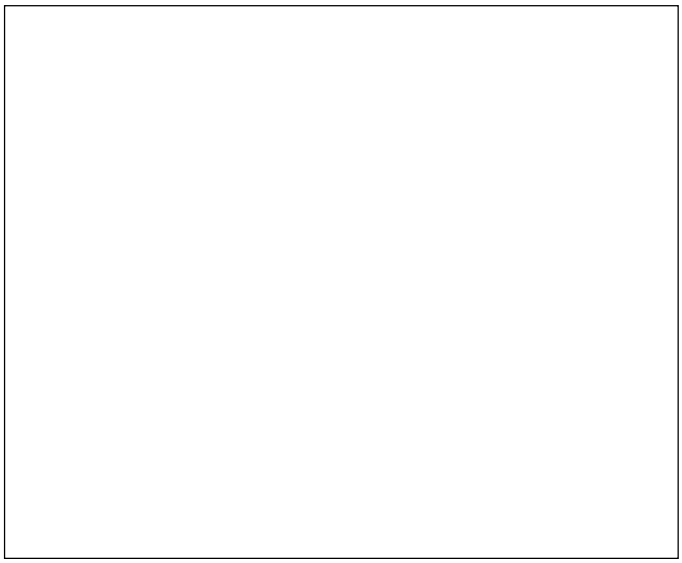

Fig.8. Algorithm of the dynamic crossover and mutation

4. EMPIRICAL EXPERIMENTS

Many experiments have been carried out to observe the performances of different LLHs of PSO-GA with dynamic parameterization in accordance to the implementation frameworks. The experiments are divided into three different sets of applications. Each application consists of three different LLHs that are developed based on the three selected implementation frameworks (SGMutation, SGCrossover, SGCrossMutation) and one single PSO. All the algorithms in the first two applications employ dynamic parameterization (time-vary and adaptive) while all the algorithms in the third application employ constant parameterization. The algorithms have been previously tested on several benchmark functions[25-26] and the Vehicle Routing Problem with Time Windows (VRPTW)[28]. At this time, the interest has been coined to test the algorithms in Facility Layout Problem (FLP)[29].

The FLP is a well-studied combinatorial optimization problem that emerged in a variety of problems such as layout design of hospitals, schools, airports, networking and backboard

wiring. The most common objective in FLP is to minimize the facility resources costs that are determined based on the flow between the facilities and the distance between each facilities location. Due to the dynamic and impulsive environment in today's industry operations, dynamic FLP appears to be very important. The dynamic FLP extends the static FLP by involving the changes in resources flow over multiple periods as well as the costs of rearranging the layout.

4.1. Particle Encoding

Based on particle encoding introduced by [29], each particle represents a valid permutation where each dimension of the particle represents a location and each value represent the corresponding facility. For example, for n=4, the particle {3 4 2 1} indicates that the third facility is assigned to the first location, fourth facility to second location, second facility to third location and first facility to fourth location. The solution encoding involves the 0-1 binary integer decision variables, enabling a randomly generated solution. Fig. 9 illustrates the definition of particles.

			Facility (*M*)			
			1	2	3	4
Location (*L*)	Period *1*	1	0	0	1	0
		2	0	0	0	1
		3	0	1	0	0
		4	1	0	0	0
	Period *2*	1	0	1	0	0
		2	1	0	0	0
		3	0	0	1	0
		4	0	0	0	1
		...	...	...	...	...
		...	...	...	...	...
	Period *P*	1	0	0	0	1
		2	0	1	0	0
		3	1	0	0	0
		4	0	0	1	0

Fig.9. Particles encoding

4.2.Experiment Setting

Each experiment was repeated for 30 times with 2000 iterations. Therefore, regardless of each

algorithm, each of the 30 trials was allowed an equal number of 60000 evaluations (30 particles X 2000 iterations). As to create a fair comparison of all the algorithms, the same seed has been fixed for random number generation so that the initial population is same for all the algorithms. The algorithms were tested on 16 sets of problem obtained from[29]. In this paper, only the datasets with facility M=6 and Period P=10 are used for comparisons. Based on the preliminary experiment, the ranges of parameter values were tested as in the following Table 2.

Table 2. Parameter setting

Parameter	Value
Number of iteration	2000
Number of particles	30
PSO personal and social learning rate, (c1, c2)	1.25
Inertia weight	0.6

5. RESULTS AND DISCUSSION

The result of each algorithm in each application are compared with single PSO algorithm as well as with the result obtained by [29] that used Dynamic Programming (DP) approach. The name of each algorithm is SGM for SGMutation, SGC to denote SGCrossover and SGCM as to present SGCrossMutation.

5.1. Adaptive Parameterization

The results in Table 3 and Table 4 show that all the PSO hybrids obtained the best solutions for all the 16 problems with 6 numbers of facility at 5 periods. It is constantly shown in the results that PSO hybrids with adaptive mutation (mainly from SGM) outperforms other algorithms for the 16 test problems. Although SGM mostly outperforms single PSO (problem 3,4, 5, 6, 8), it never performs as well as adaptive mutation-based hybrids either with SGM or with SGCM.

Table 3. Solution results for problems with M=6, P=5

Prob.	SGM	SGC	SGCM	Single PSO	DP[31]
1	**106401**	106411	**106401**	106411	106419
2	**104799**	104838	104818	104838	104834
3	**104291**	104301	**104291**	104309	104320
4	**106382**	106380	**106382**	106491	106509
5	**105621**	105614	105651	105678	105628
6	**103885**	103921	**103885**	103993	103985
7	**106396**	106405	**106396**	106405	106447
8	**103619**	103619	103628	103631	103771

Table 4. Solution results for problems with M=6, P=10

Prob.	SGM	SGC	SGCM	Single PSO	DP[31]
9	**214299**	214310	214301	214311	214313
10	**21221**	21230	21225	212340	21234
11	**207985**	207987	**207985**	207986	207987
12	**212702**	212711	**212702**	212730	212741
13	**211012**	**211012**	**211012**	211019	211022
14	**209928**	209932	**209928**	2099302	209932
15	**214232**	214252	214238	214252	214252
16	**2125811**	**2125868**	**2125811**	212585	2125888

5.2. Time-Vary Parameterization

Similar with adaptive parameterization, all PSO hybrids with time-vary parameter setting in Table 5 and Table 6 have obtained better output than the single PSO.

Table 5. Solution results for problems with M=6, P=5

Prob.	SGM	SGC	SGCM	Single PSO	DP[31]
1	**106401**	**106405**	**106398**	106411	106419
2	**104789**	**104734**	**104712**	104838	104834
3	**104278**	**104267**	**104208**	104309	104320
4	**106355**	**106357**	**106357**	106491	106509
5	**105602**	**105600**	**105523**	105678	105628
6	**103712**	**103714**	**103727**	103993	103985
7	**106396**	**106387**	**106304**	106405	106447
8	**103666**	**103653**	**103791**	103631	103771

Table 6. Solution results for problems with M=6, P=10

Prob.	SGM	SGC	SGCM	Single PSO	DP[31]
9	**214200**	**214215**	**214201**	214311	214313
10	**212218**	**212308**	**212243**	212340	21234
11	**207985**	**207945**	**207912**	207986	207987
12	**212711**	**212711**	**212719**	212730	212741
13	**211023**	**211022**	**211022**	211019	211022
14	**209788**	**209788**	**209782**	2099302	209932
15	**214054**	**214051**	**214054**	214252	214252
16	**2121671**	**2121651**	**2121531**	212585	2125888

In this case, the inclusions of single mutation or crossover, either SGM or SGC do not have too much different results with the SGCM for all problems 1 to problem 16. In addition, almost all algorithms of PSO-GA hybrids with time-vary parameterization have produced better results than the adaptive parameterization.

5.3. Constant Parameterization

It is clearly presented in Table 7 and Table 8 that all algorithms with constant parameterization have produced larger values than the previous dynamic parameterizations. The results have very slight different with single PSO [30].

Table 7. Solution results for problems with M=6, P=5

Prob.	SGM	SGC	SGCM	Single PSO	DP[31]
1	106418	106411	106401	106411	106419
2	104799	104838	104818	104838	104834
3	104291	104301	104300	104309	104320
4	106481	106488	106489	106491	106509
5	105621	105677	105669	105678	105628
6	103917	103985	103999	103993	103985
7	106400	106405	106400	106405	106447
8	103628	103619	103625	103631	103771

Table 8. Solution results for problems with M=6, P=10

Prob.	SGM	SGC	SGCM	Single PSO	DP[31]
9	214305	214310	214298	214311	214313
10	212338	212337	212334	212340	21234
11	207985	207987	207985	207986	207987
12	212702	212711	212702	212730	212741
13	211012	211012	211012	211019	211022
14	209928	209932	209928	2099302	209932
15	214232	214252	214238	214252	214252
16	2125811	2125868	2125811	212585	2125888

6. CONCLUSION

In this paper, the composition of a set of implementation frameworks for the LLH of PSO-GA [31] with dynamic parameterization has been presented. The implementation frameworks are developed based on a general taxonomy that classify the common terminology of the LLH. Based on the implementation review of the existing LLH of PSO-GA [32], it was found that the implementation frameworks with sequential global (SG) scheme is the most widely used in practice. This scheme consists of three implementation frameworks namely SGMutation, SGCrossover and SGCrossmutation. Several algorithms based on the three-implementation framework with different parameter setting have been tested on different datasets of facility layout problem. The results have indicated some improvements from the PSO-GA [33] hybrids with dynamic parameterizations compared to the constant parameterization. As these experiments only focuses on the SG scheme, the future works should extend the evaluations to other kinds of implementation frameworks that have been introduced in this paper.

7. ACKNOWLEDGEMENTS

The authors would like to thank Ministry of Education Malaysia and UniversitiTeknologi MARA for their financial support to this project under FRGS Grant No. FRGS/1/2015/ICT01/UITM/02/1.

8. REFERENCES

[1] Du K L, Swamy M N. Particle swarm optimization. In Search and optimization by metaheuristics.Cham: Springer, 2016, pp. 153-173

[2] Zambrano-Bigiarini M, Clerc M, Rojas R. Standard particle swarm optimisation 2011 at CEC-2013: A baseline for future PSO improvements. In IEEE Congress on Evolutionary Computation, 2013, pp. 2337-2344

[3] Affenzeller M., Winkler S., Wagner S., Beham A. Genetic algorithms and genetic programming-Modern concepts and practical applications.Florida: CRC Press, 2009

[4] Mirzaei A, Maimun A, Priyanto A, Fitriadhy A. Mooring pattern optimization using a genetic algorithm. Jurnal Teknologi,2014, 66(2):189–193

[5] Hamed H N A, Shamsuddin S M,Salim N. Particle swarm optimization for neural network learning enhancement. Jurnal Teknologi, 2008,49(D):13–26

[6] Ouyang P, Pano V. Comparative study of DE, PSO and GA for position domain PID controller tuning. Algorithms, 2015, 8(3):697–711

[7] Martínez-Soto R, Castillo O, Aguilar A T, Rodriguez A.A hybrid optimization method with PSO and GA to automatically design Type-1 and Type-2 fuzzy logic controllers.International Journal of Machine Learning and Cybernetics, 2013, 6(2):175–196

[8] Talbi E.G. Metaheuristics: From design to implementation. New Jersey: John Wiley and Sons, 2009

[9] Raidl G R, Puchinger J, Blum C. Metaheuristic hybrids. In M. Pardalos, H. Panos, P. Van, & M. Milano (Eds.), Handbook of metaheuristics.New York: Springer, 2010, pp. 305–335

[10] Blum C, Roli A, Alba E. An introduction to metaheuristic techniques. In E. Alba (Ed.), Parallel metaheuristic: A new class of algorithms.New Jersey: John Wiley and Sons, 2005, pp. 3–42

[11] Nickabadi A, Ebadzadeh M M, Safabakhsh R.A novel particle swarm optimization algorithm with adaptive inertia weight.Applied Soft Computing, 2011, 11(4):3658-3670

[12] Pace S S, Cain A, Woodward C J. A consolidated model of particle swarm optimisation variants. In EEECongress on Evolutionary Computation, 2012, pp. 1–8

[13] Talbi E G. A taxonomy of hybrid metaheuristics.Journal of Heuristics, 2002, 8(5):541–564

[14] Christian B, Andrea R, Metaheuristics in combinatorial optimization: Overview and conceptual comparison.ACM Computing Surveys, 2003, 35(3):268–308

[15] Cung V D, Martins S,Ribeiro C, Roucairol C. Strategies for the parallel implementation of metaheuristics.In Essays and surveys in metaheuristics.Massachusetts: Springer, 2002, pp. 263–308

[16] Crainic T G,Toulouse M.Parallel strategies for meta-heuristics. In F. Glover, & G. A. Kochenberger (Eds.), Handbook of metaheuristics.New York: Springer, 2003, pp. 475–513

[17] El-Abd M, Kamel M.A taxonomy of cooperative search algorithms.In 2nd International Workshop on Hybrid Metaheuristics, 2005, pp. 32–41

[18]Crainic T G. Parallel computation, co-operation, tabu search. In R. Sharda, S. Voß, C. Rego, & B. Alidaee (Eds.), Metaheuristic optimization via memory and evolution.Massachusetts: Springer, 2005, pp. 283–302

[19]Xin B, Chen J, Zhang J, Fang H, Peng Z H. Hybridizing differential evolution and particle swarm optimization to design powerful optimizers: A review and taxonomy. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 2012, 42(5):744–767

[20] García-Martínez C, Cordón O, Herrera F.A taxonomy and empirical analysis of multiple objective ant colony optimization algorithms for the bi-criteria TSP.Journal of Operational Research, 2007, 180(1):116–148

[21] Dower S,Woodward C J. ESDL: A simple description language for population-based evolutionary computation.In 13th Annual Conference on Genetic and Evolutionary Computation, 2011, pp. 1045–1052

[22] El-Abd M, Kamel M S. A taxonomy of cooperative particle swarm optimizers. International Journal of Computational Intelligence Research, 2008, 4(2):137-144

[23] Herrera F, Lozano M, Sánchez A M.A taxonomy for the crossover operator for real-coded genetic algorithms: An experimental study.International Journal of Intelligent Systems, 2003,18(3):309–338

[24] Aleti A. An adaptive approach to controlling parameters of evolutionary algorithms.Victoria: Swinburne University of Technology, 2012

[25] Masrom S, Abidin S Z Z, Omar N, Nasir K. Time-varying mutation in particle swarm

optimization. In 5th Asian Conference on Intelligent Information and Database Systems,pp.31-40

[26] Masrom S, Moser I, Jontgomery J, Abidin S Z Z, Omar N. Hybridization of particle swarm optimization with adaptive genetic algorithm operators. In IEEE International Conference on Intelligent Systems Design and Applications, 2013, pp. 1–6

[27] Chen S. Particle swarm optimization with pbest crossover. InIEEE Congress on Evolutionary Computation,2012,pp. 1–6

[28] Masrom S, Abidin S Z Z, Omar N, Nasir K, Abd Rahman A S. Dynamic parameterization of the particle swarm optimization and genetic algorithm hybrids for vehicle routing problem with time window. International Journal of Hybrid Intelligent Systems, 2015, 12(1):13-25

[29] Balakrishnan J, Cheng C H, Conway D G. An improved pair-wise exchange heuristic for the dynamic plant layout problem.International Journal of Production Research, 2000,38(13):3067–3077

[30] Indera N I, Yassin I M, Zabidi A, Rizman Z I. Non-linear autoregressive with exogeneous input (NARX) Bitcoin price prediction model using PSO-optimized parameters and moving average technical indicators. Journal of Fundamental and Applied Sciences, 2017, 9(3S):791-808

[31] Zabidi A, Yassin I M, Tahir N M, Rizman Z I, Karbasi M. Comparison between binary particles swarm optimization (BPSO) and binary artificial bee colony (BABC) for nonlinear autoregressive model structure selection of chaotic data. Journal of Fundamental and Applied Sciences, 2017, 9(3S):730-754

[32] Yassin I M, Zabidi A, Ali M S, Tahir N M, Hassan H A, Abidin H Z, Rizman Z I. Binary particle swarm optimization structure selection of nonlinear autoregressive moving average with exogenous inputs (NARMAX) model of a flexible robot arm. International Journal on Advanced Science, Engineering and Information Technology, 2016, 6(5):630-637

[33] Masrom S, Abidin S Z, Omar N, Rahman A S, Rizman Z I. Dynamic parameterizations of particle swarm optimization and genetic algorithm for facility layout problem. ARPN Journal of Engineering and Applied Sciences, 2017, 12(10):3195-3201

[34] Mahtar S N A M, Masrom S, Omar N, Khairudin N, Rahim S K N A, RizmanZ I. Trust aware recommender system with distrust in different views of trusted users.Journal of

Fundamental and Applied Sciences, 2017, 9(5S):168-182

[35] Masrom S, Abidin S Z Z, Omar N, RizmanZ I. Software framework for optimization problems and meta-heuristics based on scripting language.Journal of Fundamental and Applied Sciences, 2017, 9(5S):49-57

[36] Ibrahim R, Leng NS, Yusoff RC, Samy GN, Masrom S, Rizman ZI. E-learning acceptance based on technology acceptance model (TAM). Journal of Fundamental and Applied Sciences, 2017, 9(4S):871-889

[37] Ibrahim R, Masrom S, Yusoff R C, Zainuddin N M, Rizman Z I. Student acceptance of educational games in higher education. Journal of Fundamental and Applied Sciences, 2017, 9(3S):809-829

How to cite this article:
Masrom S, Rahman A S A, Abidin S Z Z, Omar N, Rizman Z I. The implementation frameworks of meta-heuristics hybridization with dynamic parameterization. J. Fundam. Appl. Sci., 2017, *9(6S), 558-576.*