An International Journal
JSCDSS
http://www.jscdss.com
Vol.4 No.6 December 2017: 1-6
Article history:
Accepted 28 October 2017
Published online 28 October 2017

Journal of Soft Computing and Decision Support Systems

E-ISSN: 2289-8603

Metaheuristic Algorithms: Guidelines for Implementation

Ashkan Memari [a,*], Robiah Ahmad [a], Abd. Rahman Abdul Rahim [a]
[a] Department of Engineering, UTM Razak School of Engineering and Advanced Technology
Universiti Teknologi Malaysia, Jalan Sultan Yahya Petra, 54100, Kuala Lumpur, Malaysia

*** Corresponding author email address**: ashkan.memari@utm.my

Abstract

This paper presents a quick review of the basic concepts and essential steps for implementing of metaheuristic algorithms. It can be therefore used as a roadmap to shed light on solving an optimization problem using a metaheuristic algorithm. We provide a brief review of the topics, including general concepts for metaheuristics, the need to design metaheuristics, the need for further improvement of metaheuristics, parameters tuning and performance assessment of metaheuristic algorithms. Finally, the paper ends with a guideline framework which aims to assist new researchers for solving optimization problems via metaheuristics.

Keywords: Metaheuristic algorithms, Optimization, Literature review, Performance assessment, Guidelines for implementation

1. Introduction

Heuristic means 'to find' or 'to discover solutions by trial and error'. Heuristic methods aim at searching feasible solutions for large-scale problems (Yang, 2010). They have proven to overcome many shortcomings of traditional optimization methods, however, they are typically experience-based methods. Heuristic methods find the solution which hopes to be near optimal solution, rapidly. However, heuristic techniques are not considered effective for the complex problems. There are two main reasons for this:

i. Although heuristic techniques are usually (not always) provide a decent and acceptable answer in solving complex problems; they do not promise and guarantee the optimal solution for solving complex problems (Chinneck, 2004).

ii. Heuristic techniques are highly depending on experience and mathematical knowledge of the model developer (Coello et al., 2007).

The development of heuristic algorithms is named metaheuristic algorithms. The term "meta" means "beyond" or "higher level". Metaheuristics often perform better compare to simple heuristics. Basically, metaheuristic algorithms use certain trade-offs of randomization and local search (Yang, 2010). Since metaheuristic algorithms conduct a more thorough search, they have rapidly become the favoured methodology for generating solutions to complex real-life problems which exact methods are unable to solve (Glover and Kochenberger, 2003). There are several available classifications for metaheuristic algorithms in the literature. Fig. 1 presents a general classification of metaheuristic algorithms.

2. The need to design a metaheuristic algorithm: complexity of problems

The main branch of the theory of computation, which is known as computational complexity theory, aims at sorting problems concerning their inherent difficulty. The complexity associated with a problem is actually equivalent to the finest algorithm tackling that problem. Furthermore, the complexity theory concerns with the decisions which they always have a 'yes' or 'no' answer. In this line, a problem is called easy or tractable if a polynomial-time algorithm can solve it. If no polynomial-time algorithm can solve a problem, it is known as a difficult or intractable problem.

Another goal in computational theory is to sort problems into specified complexity classes. For a given amount of computational resources, a complexity class corresponds to the set of all problems that can be solved. *P (*polynomial time*)* and *NP* (nondeterministic polynomial time) are two main classes of problems in complexity classes (Coello et al., 2007).

A problem is in *P* class if a deterministic algorithm can solve all the decision problems in polynomial time and a problem is *NP* if a non-deterministic algorithm can solve all decision problems in polynomial time (Talbi, 2009). In this line, one of the central open questions is whether *P* = *NP* or not? The answer would have a broad impact on

computational complexity theory. Clearly, $P \subseteq NP$, since for any problem in class P there is a non-deterministic algorithm solving it. On the other hand, the following surmise $P \subset NP$ is still an open question (Talbi, 2009; Coello et al., 2007).

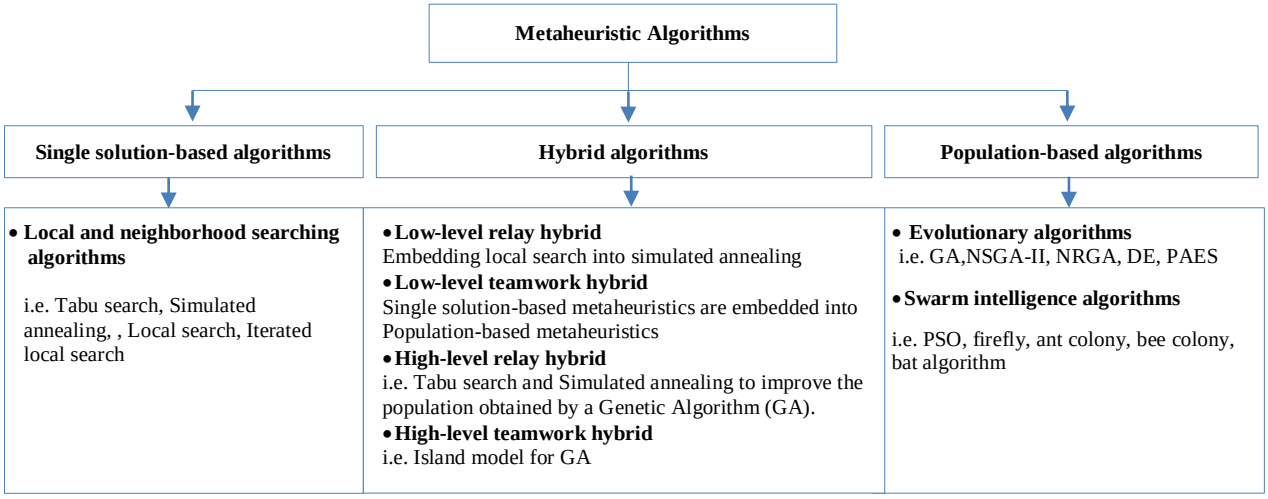

Fig. 1. General classification of metaheuristic algorithms.

In order to reduce a decision problem polynomially, suppose A and B are decision problems. A is reduced to B in polynomial time, if for all input instances IA for A, it is always possible to construct an input instance IB for B in polynomial time function to the size L (IA) of the input IA, such that IA is a positive instance of A if and only if IB is a positive instance of B (Talbi, 2009).

A decision problem $A \in NP$ is called an NP-complete problem if all of other NP-class problems are reduced to the problem A in polynomial time. If an NP-complete problem can be solved by a polynomial deterministic algorithm, then all problems in class NP may be solved polynomially. The optimization problems with NP-complete decision problems are called NP-hard problems. The majority of real-life optimization problems belong to NP-hard class, and in order to solve NP-hard problems, there are no demonstrably efficient algorithms. NP-hard problems can be optimally solved in exponential time (unless $P = NP$). One of the main alternatives to solve this class of problems is metaheuristic algorithms (Talbi, 2009; Coello et al., 2007).

3. The need for further improvement of metaheuristics: hybridization

Presently, upon solving a problem, the primary objective is actually to find the best possible way in solving, rather than promoting a specific metaheuristic. One of the key alternatives to improve metaheuristic algorithms is hybridization. The hybrid algorithms are believed to benefit from synergy. This means that deciding on a sufficient combination of complementary algorithmic concepts could be one of the keys to achieve the highest performance in solving many difficult optimization problems (Blum et al., 2011). Adding a local searcher into evolutionary algorithms, sometimes known as memetic algorithms, has shown to be successful (Boussaïd et al., 2013; Blum et al., 2011; Lozano and García-Martínez, 2010). Besides, Blum et al. (2008) published the first textbook that specifically devoted to different hybrid metaheuristic algorithms.

4. Parameters tuning of metaheuristics

In implementing of metaheuristic algorithms, the obtained results are sensitive to algorithms' parameters. Therefore, the parameters require being fine-tuned in order to obtain solutions with better quality (fitness function value). There are two different approaches for parameters tuning of metaheuristics: the online parameter tuning and the off-line parameter initialization approach. In the online strategy, the parameters are updated and controlled dynamically or adaptively during the metaheuristic execution, whereas, in off-line parameter initialization (or meta-optimization), the values of different parameters are fixed before the metaheuristic is executed. For further information about advantages and disadvantages of different parameter tuning strategies, please refer to (Talbi, 2009).

5. Performance assessment of metaheuristic algorithms

With fast growing of metaheuristic algorithms, the challenges in evaluating their performance have also become more and more highlighted. In performance assessment of single-objective optimization, empirical performance assessment usually consists of the obtained solutions' quality and the computational resources to generate these solutions. However, the main difficulty in performance evaluation of multi-objective optimization arises when the optimization output is not a single solution;

E-ISSN: 2289-8603

however, a set of solutions demonstrating a Pareto front approximation of a multi-objective problem.

For the performances' assessment of multi-objective metaheuristics, comparison of the non-dominated solutions set is an essential (Fig. 2). In literature, different metrics have been suggested for non-dominated sets; however, a generally accepted standard for performance assessment does not exist. Quality metrics may be categorized by different properties (Talbi, 2009).

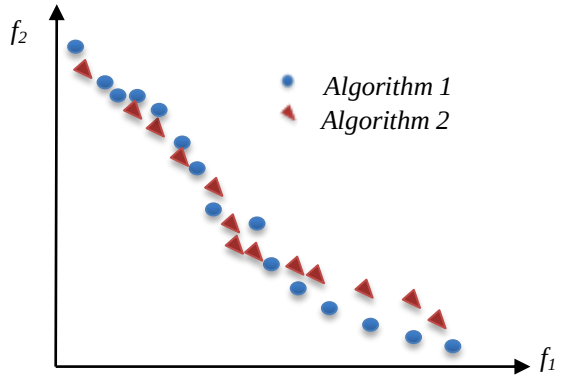

Fig. 2. Comparing two sets of Pareto front solutions in a bi-objective optimization case.

Fig. 2 shows some solutions of Algorithm 1 dominated solutions of Algorithm 2 and vice versa. Besides, some of the both algorithms solutions are incomparable.

In a multi-objective optimization problem, there are two distinct goals. The first goal is to search for those solutions that they are as close as possible to the Pareto-optimal solutions and the second goal is to find the solutions as diverse as possible in the obtained non-dominated front. The first goal needs to discover towards the Pareto-optimal front whereas the second goal needs to discover along the Pareto optimal front. These goals, in some sense, are usually orthogonal together, as depicted in Fig. 3. Therefore, at least two performance metrics should be considered for both these goals.

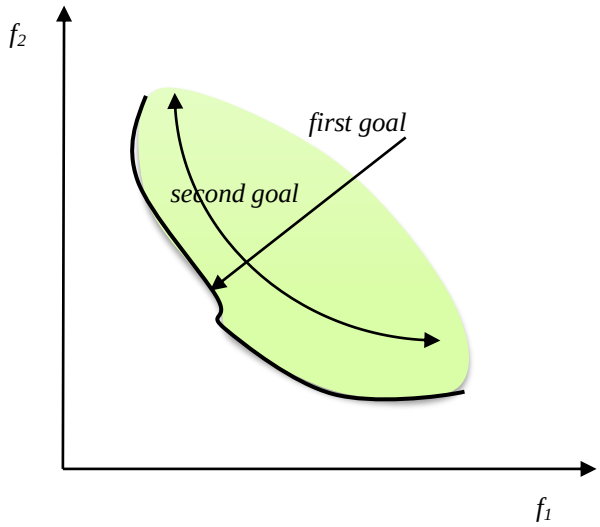

Fig. 3. Schematic of two goals in bi-objective optimization.

Many studies in this field have been done to develop the performance metrics. Knowles and Corne (2000) have compared most of these metrics based on their compatibility without performance relations between two sets of solutions and their monotonicity properties. Hansen and Jaszkiewicz (1998) described three performance metrics. These metrics evaluate two non-dominated sets and determine the expected total number of cases that a set of solutions dominates other solutions set.

According to Zitzler et al. (2002), a minimum *n* performance metrics are required to compare two or more sets of solutions when optimizing an *n*-objective problem. Use of any number less than *n* would result in inaccurate judgment because of dimensionality reduction. Deb and Jain (2002) showed that using a functionally independent set of variables can overcome this dimensionality problem. This would make the problem inaccurate theoretically but feasible practically. They also proposed two new running performance metrics, one for measuring the population members' diversity in each generation of a Multi-Objective Evolutionary Algorithm (MOEA) run and another for determining the convergence to the reference set. As a conclusion, performance metrics must be selected in a way to consider two fundamental aspects: convergence and diversity. In this research, the most frequently applied performance indicators are introduced.

5.1 Convergence-based metrics

Convergence metrics estimate the solutions' efficiencies with regards to the closeness to the optimal Pareto front. Various convergence-based measures such as error ratio, contribution, distance from reference, etc. have been suggested in the literature. Among the proposed metrics, generational distance is the most applied metric.

- *Generational distance*

Generational Distance (GD) metric measures the average distance between a reference set *R* and an approximated set *A* (Van Veldhuizen and Lamont, 2000). The exact Pareto front *PFtrue* represents the reference set *R*. In a given iteration *t*, the distance between the two sets, at a given iteration *t*, is obtained based on average distance over the pairwise minimum distances. The distance value is calculated using the Euclidean distance in the objective space. The value for generational distance is equal to 0, while the approximated front *A* is involved in the reference set *A*.

$$GD \cong \frac{\left(\sum_{i=1}^{n} d_i^{\,p}\right)^{1/p}}{n} \quad (1)$$

Where *di* is the Euclidean distance between the closest member of *PFknown* and the closest member of *PFtrue*, *p=2* and *n* denotes the number of vectors in *PFknown*. If *PFtrue* = *PFknown*, then GD is equal to 0, any other value of GD specifies *PFknown* deviates from *PFtrue*.

JSCDS
E-ISSN: 2289-8603

5.2 *Diversity-based metrics*

Diversity-based metrics measures the uniformity distribution of the solutions with regards to extension and dispersion. The diversity is generally explored in the objective space. In this kind of indicators, one can find two set coverage, the entropy, spacing and spread. Among them, two set coverage and spacing are the most applied metrics in the literature.

- *Spacing*

The spread of the vectors in *PFknown* is numerically described by the spacing metric. More precisely, the distance variance of neighboring vectors in *PFknown* is measured by this Pareto noncompliant metric. This metric is presented in Eq. (2).

$$S \cong \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(\overline{d} - d_i)^2} \tag{2}$$

Where n is the number of vectors in *PFknown*, and $d_i = \min_j\left(\left|f_1(x^i) - f_1(x^j)\right| + \left|f_2(x^i) - f_2(x^j)\right|\right)$, $i, j = 1, 2, \ldots, n$. $\overline{d}$ is the mean of all d_i and n is the number of vectors in *PFknown*. The result of $S = 0$ specifies all *PFknown* members are equidistantly spaced, however; the true situation in *PFtrue* maybe not so (Coello *et al*., 2007).

- *Two set coverage*

Two set coverage is a comparative measure which can also be named as relative coverage comparison of two sets (Zitzler et al., 2000). Suppose that X', $X'' \subseteq X'$ are two sets of phenotype decision vectors. Two Set Coverage is defined as the mapping of the order pair (X', X'') to the interval [0, 1] per Eq. (3).

$$CS(X', X'') \cong \frac{\left|\{a'' \in X''; \exists a' \in X' : a' \geq a''\}\right|}{\left|X''\right|} \tag{3}$$

If all points in X' dominant or equal to all points in X'', then by definition $CS = 1$. $CS = 0$ implies the opposite. Basically, *CS (X', X'')* and *CS (X'', X')* both have to be taken into account in order to avoid empty sets of the intersection. This metric can be applied for X = *Pknown* or *PFtrue* as well. The main advantages of this kind of Pareto compliant metric are the ease of calculation and providing relative comparison based on dominance numbers among MOEAs or generations.

5.3 *Statistical-based metrics*

Statistical-based metrics also can be used to measure the performance of metaheuristic algorithms. For instance, relative deviation index (RDI) and relative percentage deviation (RPD) are among the popular statistical metrics. RDI and the RPD are defined as (Dean, 1999):

$$RDI = \left(\frac{Alg_{sol} - Min_{sol}}{Max_{sol} - Min_{sol}}\right) \times 100 \tag{4}$$

$$RPD = \left(\frac{Alg_{sol} - Min_{sol}}{Min_{sol}}\right) \times 100 \tag{5}$$

Where Alg_{sol} is the value of the fitness function and Min_{sol} and Max_{sol} denote the best and the worst obtained solutions by a methaheuritics algorithm, respectively. Note that lower values of both RDI and RPD are indicatives of better efficiency of the solution algorithms.

6. Guidelines for solving an optimization problem using metaheuristics

Once a problem was formulated, the roadmap below may constitute a guideline in solving a problem (see Fig. 4).

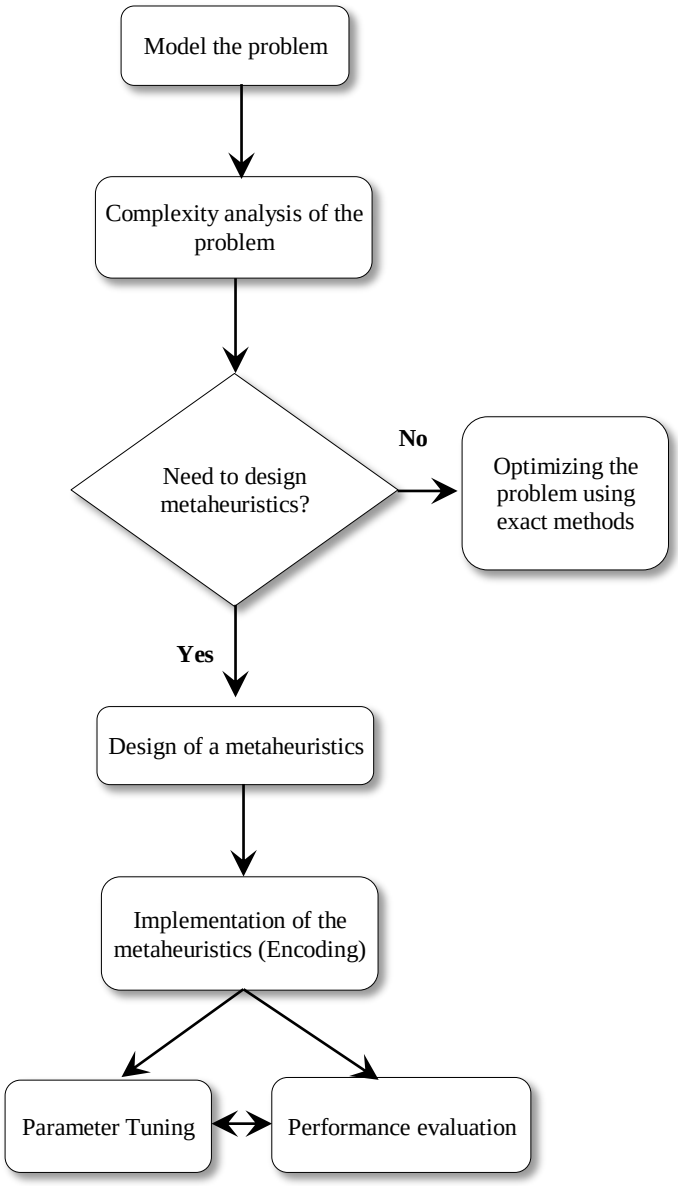

Fig. 4. Guidelines for solving a given optimization problem.

E-ISSN: 2289-8603

First, the problem difficulty and complexity must be taken into account. There are several reasons for choosing a metaheuristic algorithm to solve a problem, and one of the most important benefits of using metaheuristic algorithms is to avoid being trapped in local optima when exploring to find a feasible solution. Moreover, in optimization problems, metaheuristic algorithms seek fine solutions where the problem complexity or the search time does not allow applying exact optimization methods. When the need to design a metaheuristic algorithm is recognized, care should be taken to two common issues (Talbi, 2009):

i. The parameters that may have a substantial effect on the effectiveness and efficiency of the search should be calibrated for every metaheuristic algorithm. Tuning of Parameter may allow a robustness and larger flexibility; however, needs a careful initialization. It should be noted, there is no obvious prior exist to be defined which parameters' setting should be applied, but the optimal values of the parameters highly depend on two issues: (i) the problem or the instance at hand, (ii) the search time in solving the problem which the user desires to spend. Note that, a generally optimal parameter values set for metaheuristics does not exist.

ii. The last step of this roadmap is to assess the performance of the developed metaheuristic. It is an essential task to achieve and must be completed on a reasonable basis. Three steps must be considered to analyze the performance of a metaheuristic: (i) experimental design (e.g. selected instances and factors, goals of the experiments), (ii) measurement (such as measuring the quality of solutions) and (iii) reporting (such as box plots).

7. Conclusion

The purpose of the current study was to investigate the essential steps in implementing of metaheuristic algorithms. Although an extensive literature is available describing the implementation of metaheuristics; however, a brief review focusing on their immediate implication is almost sparse. It must be noted that when metaheuristics were applied to solve an optimization problem (particularly in engineering fields), in most cases; the complexity analysis of the problem, performance assessment and parameters tuning of the metaheuristics were totally neglected. The guidelines presented here can be used as a tool to assist the researchers in better implementation of metaheuristics.

Acknowledgments

The authors would like to thank Universiti Teknologi Malaysia (UTM) and Ministry of Higher Education (MOHE) Malaysia under Fundamental Research Grant Scheme (FRGS) Vot 4F850 for financial support provided throughout this research. The first author is a researcher at UTM under the Post-Doctoral Fellowship Scheme (PDRU Grant) for the project: "A Tuned NSGAII for Optimizing JIT Distribution Networks", Vot No.Q. J130000. 21A2. 03E46.

References

Blum, C., Blesa, M.J., Aguilera, A. Roli, M. (2008), Hybrid Metaheuristics-An Emerging Approach to Optimization, Studies in Computational Intelligence, volume 114, Springer-Verlag, Berlin, Germany.

Blum, C., Puchinger, J., Raidl, G. R. and Roli, A. (2011). Hybrid metaheuristics in combinatorial optimization: A survey. Applied Soft Computing. 11 (6): 4135-4151.

Boussaïd, I., Lepagnot, J. and Siarry, P. (2013). A survey on optimization metaheuristics. Information Sciences. 237: 82-117.

Chinneck, J. W. (2004). Practical optimization: a gentle introduction. Electronic document: http://www. sce. carleton. ca/faculty/chinneck/po. html.

Coello, C. C., Lamont, G. B. and Van Veldhuizen, D. A. (2007). Evolutionary algorithms for solving multi-objective problems. Springer Science & Business Media.

Deb, K. and Jain, S. (2002). Running performance metrics for evolutionary ourth Asia-Pacific Conference on Simulated Evolution and Learning (SEAL'02),(Singapore), Proceedings of the Fourth Asia-Pacific Conference on Simulated Evolution and Learning (SEAL'02), Singapore.

Glover, F. and Kochenberger, G. A. (2003). Handbook of metaheuristics. Springer.

Hansen, M. P. and Jaszkiewicz, A. (1998). Evaluating the quality of approximations to the non-dominated set. IMM, Department of Mathematical Modelling, Technical Universityof Denmark.

Knowles, J. D. and Corne, D. W. (2000). Approximating the nondominated front using the Pareto archived evolution strategy. Evolutionary computation. 8 (2): 149-172.

Lozano, M. and García-Martínez, C. (2010). Hybrid metaheuristics with evolutionary algorithms specializing in intensification and diversification: Overview and progress report. Computers & Operations Research. 37 (3): 481-497.

Talbi, E.-G. (2009). Metaheuristics: from design to implementation. John Wiley & Sons.

Van Veldhuizen, D. and Lamont, G. B. (2000). On measuring multiobjective evolutionary algorithm performance. Evolutionary Computation, 2000. Proceedings of the 2000 Congress on, IEEE.

Yang, X.-S. (2010). Engineering optimization: an introduction with metaheuristic applications. John Wiley & Sons.

Zitzler, E., Deb, K. and Thiele, L. (2000). Comparison of multiobjective evolutionary algorithms: Empirical results. Evolutionary computation. 8 (2): 173-195.

Zitzler, E., Laumanns, M., Thiele, L., Fonseca, C. M. and da Fonseca, V. G. (2002). Why Quality Assessment Of Multiobjective Optimizers Is Difficult. GECCO.

Author Biographies

Ashkan Memari is currently a postdoctoral fellow at UTM Razak School in Kuala Lumpur. He received his MSc and PhD in Industrial Engineering from UTM. His areas of interest include operations research, optimisation and evolutionary computation.

Robiah Ahmad is currently an Associate Professor at UTM Razak School in Kuala Lumpur. She graduated with a BSc in Electrical Engineering and BSc in Engineering Management from the University of Evansville USA in

E-ISSN: 2289-8603

1988. She obtained her MSc in Information Technology for Manufacture from Warwick University in 1992 and her PhD degree from the Universiti Teknologi Malaysia in 2004. Her areas of interest include system modelling and identification as well as optimisation using evolutionary computation.

Abd Rahman Abdul Rahim is currently an Associate Professor at UTM Razak School in Kuala Lumpur. He graduated with a BSc in Mechanical Engineering and BSc in Engineering Management from the University of Evansville USA in 1988. He obtained his MSc in Manufacturing Systems Engineering from Warwick University in 1991 and his PhD degree from Universiti Teknologi Malaysia in 2006. His areas of interest include supply chain and new product development. He is involved in research and consultancy with both local and multinational corporations.

E-ISSN: 2289-8603