Journal of Fundamental and Applied Sciences

ISSN 1112-9867

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

Research Article

Special Issue

SOFTWARE FRAMEWORK FOR OPTIMIZATION PROBLEMS AND META-HEURISTICS BASED ON SCRIPTING LANGUAGE

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

[1]Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA, Perak Branch, Tapah Campus, Perak, Malaysia

[2]40450 Shah Alam, Selangor, Malaysia

[3]Faculty of Electrical Engineering, Universiti Teknologi MARA, 23000Dungun, Terengganu, Malaysia

Published online: 17 October 2017

ABSTRACT

Since most researchers working in meta-heuristics fields are required to develop their own algorithm for their specific optimization problem, the task of selecting, modifying and extending codes from rapid software framework is very likely to occur. Software framework enables design and codes reuses to allow faster programming development of the meta-heuristics. This paper presents a review of the available rapid software framework for meta-heuristics and summarize some important features of the rapid software. A description about a new software framework based on scripting language is given. Furthermore, the scripting language that was used to develop a meta-heuristics algorithm of PSO-GA hybridization is evaluated according to the conciseness aspects. This results has clearly indicate that the program codes with scripting language has more conciseness that the main JAVA codes.

Keywords: software framework; scripting language;optimization;meta-heuristics.

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

doi: http://dx.doi.org/10.4314/jfas.v9i5s.4

1. INTRODUCTION

Optimization is presented in many aspects of real life activities, especially those that are related to scheduling, resource allocating and timetabling. To serve these activities,the researchers, users and organizations either private companies or public institutions have to confront with different decision alternatives for a huge number of planning and optimization problems. These tasks are really important to many professions. Generally, in practice, the steps for solving optimization problems are problem definition, problem model construction and problem optimization as illustrated in Fig. 1[1].

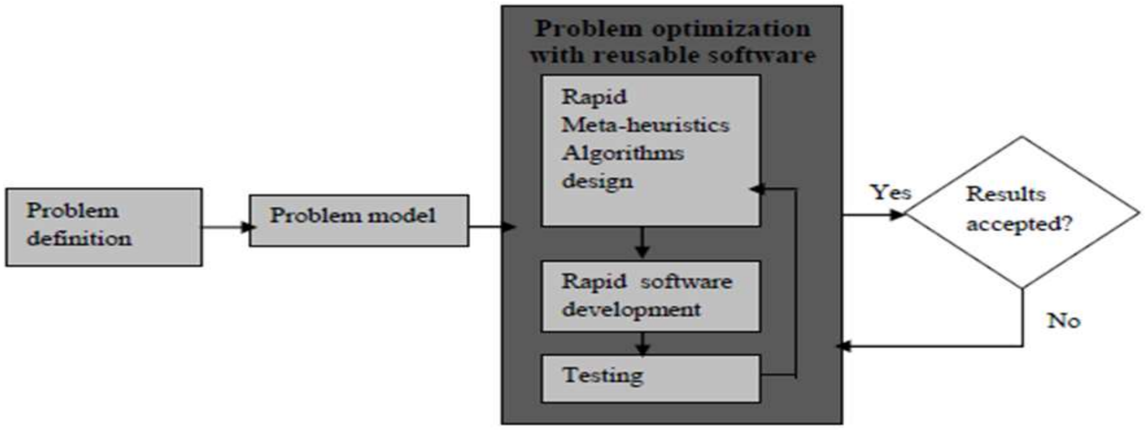
Problem optimization with reusable software
Rapid Meta-heuristics Algorithms design
Rapid software development
Testing
Problem definition
Problem model
Yes
Results accepted?
No

Fig.1. Solving optimization problem with meta-heuristics

Problem optimization with reusable software is the main interest of this paper. Therefore, this paper provides a brief discussion about the reusable software design and requirements including the programming approaches. Focusing on meta-heuristics hybridization[2], this paper provides reviews of existing software for meta-heuristics, the fundamental aspects of the software will be highlighted. Another contribution of this paper is the conciseness evaluation for the proposed scripting language.

2. BACKGROUND OF THE STUDY

2.1.Software Reuse for Meta-heuristics

Software reuse is defined as creating new software from the existing software rather than building software systems from scratch[3]. Nowadays, no one would seriously argue that software reuse has become a common in the rapid software engineering practice. Software reuse can be accomplished through several mechanisms and the following parts briefly explain some of the them namely software library, software framework, scripting language and domain-specific language.

2.2. Software Library

Software library is a collection of codes written with a set of well-defined interfaces that can interact with another independent program. The independent program can work with the software library without any details of the internal codes in the library. The advantages can be achieved with a minimum amount of codes development in a particular software, hence reduces the risk of software production for new projects. Nevertheless, the structure of different components in a software library is highly connected and dependent upon each other's resulting in the software operating similarly to that of a black-box reusable software. Black-box software library prohibits users to manipulate the invariant part of the software, which in turn requires major changes of components and functions for any works involving software library extension[4]. Modifying software library would require almost the same amount of effort as building the product itself from scratch. Hence, the advantages of software reuse to reduce software development risks could not be achieved in maximum.

2.3.Software Framework

Reusability only on the codes but not on the design is the main drawback of a software library. However, the conventional software library can be designed to enable both reusable source code and design, which is called as (object-oriented) class libraries or object-oriented software frameworks.

Although no generally accepted definition has been established for software frameworks, it is generally agreed (and used in this paper) that a software framework is a reusable part of software architecture comprising of both design and code. In [5]define a software framework as a set of classes that represents an abstract design and implementation process for an application in a given problem domain. A key distinguishing feature that separates a software framework from normal libraries is the software has better abstraction toenablegeneral functionality that can be changed with additional user-written codes for domain-specific functions. Thus, extending a software framework with user codes is easier than using a software library without knowing the details of the whole design, codes and internal working structure of the software.

2.4.Scripting Language

Other than software library and software framework, scripting languages have achieved remarkable acceptance [20] from the software developer. Scripting languages are easier to use than conventional programming languages such as C, C++ and JAVA. Two major enhancing features of scripting languages that support efficiency includes wordless [6]and easiness [7]. The advantage of wordless is reduces the developmental errors in programming. The scripting languages are normally less expressive, but concise.

A scripting language can be used to glue together existing smaller applications into a new application[6]. More than that, scripting language can be used to interact with the program from the software library or software framework that are developed with complex programming languages such as C or JAVA[7-8]. In this case, scripting language is operated as a high-level language for the complex programming language through the software reuse paradigm. Scripting programming at the front-end promises better productivity for application or algorithm development, whilst the other supports high efficiency performance in running algorithms[9].

2.5.Domain Specific Language (DSL)

Different with general-purpose programming language (GPL), domain-specific language (DSL) is a programming language that is personalized to a specific application domain. To enhance this further, the following describes more details of the four key elements of DSL as suggested by [7]:

2.5.1.Computer Programming Language

The use of DSL is another kind of programming language used by humans to interact with a computeras to achieve something. While the structure of DSL is designed with humans in mind such as readable, it must also be executable by a computer. Pseudo-code is a code that can be read easily by humans, but it is not a computer programming language. Currently, there exists modelling type programming languages in meta-heuristics domain for instances MDF[10] and ParadisEO[11]. The modelling language is considered as a computer programming language if it has an automatic translator that converts the modelling codes into an executable computer codes.

2.5.2.Language Nature

While DSL is a programming language, it should has a sense of fluency similar to human language. In simple chronology, it is always true to say that better fluency fosters better comprehension among peoples. With fluency, they can speak and convey information easily, smoothly and effectively. Similarly, DSL fluency is a good principle that can improve programmer productivity. Fluency is highly achievable if the language has small amounts of elements, keywords characters and expression.

2.5.3.Limited Expressiveness

Unlike GPL, a DSL supports a very minimum of features that good enough to support its domain. For DSL, expressiveness comes not just from individual expressions but also from the way they can be intergrated together. Instead of expressiveness, conciseness is more relevant to DSL.

2.5.4.Domain Focus

A DSL can feature a smaller scope of domain and can be designed with lesser program elements. The tendency of programmers to be more understandable and competent in the programming language would be greatly attained with domain focus DSL. Due to domain focus, DSL tends to have more concern on knowledge abstraction. One of the abstract features in a DSL is the keywords utilization. Most GPL has plural keywords for a particular operation thus a larger number of words are required to convey the similarinstructions. For example, while, do-while and for are representing control keywords for repetitive expressions with each one having a different syntactical presentation. In JAVA, at least three keywords are used just for displaying a message to screen such as *system:out:print(Goodbye).*

Scripting language can be GPL or DSL, but modern scripting language usually tends to be more DSL. Older scripting language such as Perl, TCL, Phyton and CGI is developed to support the more common types of traditional scripting including system administration, controlling remote applications, command line interface and server-side programming on the web[12]. At the beginning, these scripting languages are developed with DSL in mind. The Perl language for example was developed to support traditional activities such as navigating large file systems and manipulating large amounts of text, but due to the common requirement to networking, the language is supported with network and socket programming. While this language is usable for traditional scripting, major advancements have been done with the language so that now the language is capable of developing many kinds of applications with different technologies for examples databasegraphical user interfaces (GUI), networking and distribution processing.

To date, the number of modern scripting languages has been rapidly increasing. As this paper introducing scripting language for meta-heuristics, DSL features such as domain focus, concise and keywords utilization have been fundamental aspects in the programming language design.

2.6.Programming Approaches in Meta-heuristics Reusable Software

Since most researchers working in the meta-heuristics field are required to develop their own algorithm for their specific algorithm, the task of selecting, modifying and extending codes from reusable software is very likely to occur. They can make use of different programming approaches supported by reusable software. Generally, the two common approaches are GUI or programming language.

GUI is a programming approach operated at the front-end of reusable software, which performs operations in a drag-drop programming environment. Usually, GUI is used as a medium interface for algorithm and experimental configurations, or to display optimization results with visuals such as graphs and charts. GUI is very easy and convenient, but it is clumsy and has a strict rigidity to pre-defined functions of a software library or framework[13]. Literature has founded that GUI is not being widely adopted in the reusable software for meta-heuristics[14]. Moreover, since meta-heuristics is adaptable for different problems which demand more flexibility from the software to create new programs for a user-defined problem, deployment of a programming language, appears to be more ideal than the GUI approach. In addition, programming languages which are often in a textual form, have extra advantages as the following.

2.6.1.Free Editor and Platform Independency

Text based programming language can be developed using free editors such as Notepad, Notepad++ and Eclipse that are freely downloaded through the Internet. In addition, this software are platform independency.More interesting, Eclipse provides special environment support for syntax highlighting and excellent navigation capability.

2.6.2. Fast Processing

Most often, writing text programs is more efficient than drawing graphical models. While this is very helpful for the inexperienced programmer, working with the graphical tools uses a huge amount of computer memory and tends to take a longer processing time. Sometimes, it still takes a lot of effort to understand the graphical elements in GUI.

2.6.3. More Flexible

Text based programs have more flexibility than GUI for software extension. Once a developer is familiar with the programming language used in reusable software, they can be very productive for a variety of software creations. In meta-heuristics mainly the hybridization, flexibility is highly essential. As mention in the previous section, GPLs are known as a highly efficient language and usually being used for developing back-end software libraries or software frameworks. Thus, in this research, GPL is used as the back-end software from the overall software architecture. Programmers have to use GPL if the reusable software does not have GUI or DSL. With GPL, programmers must have experience with the language being used which is definitely more difficult than GUI and DSL. Fig. 2 illustrates the advantages and disadvantages of the front-end GUI and back-end GPL as a programming approach.

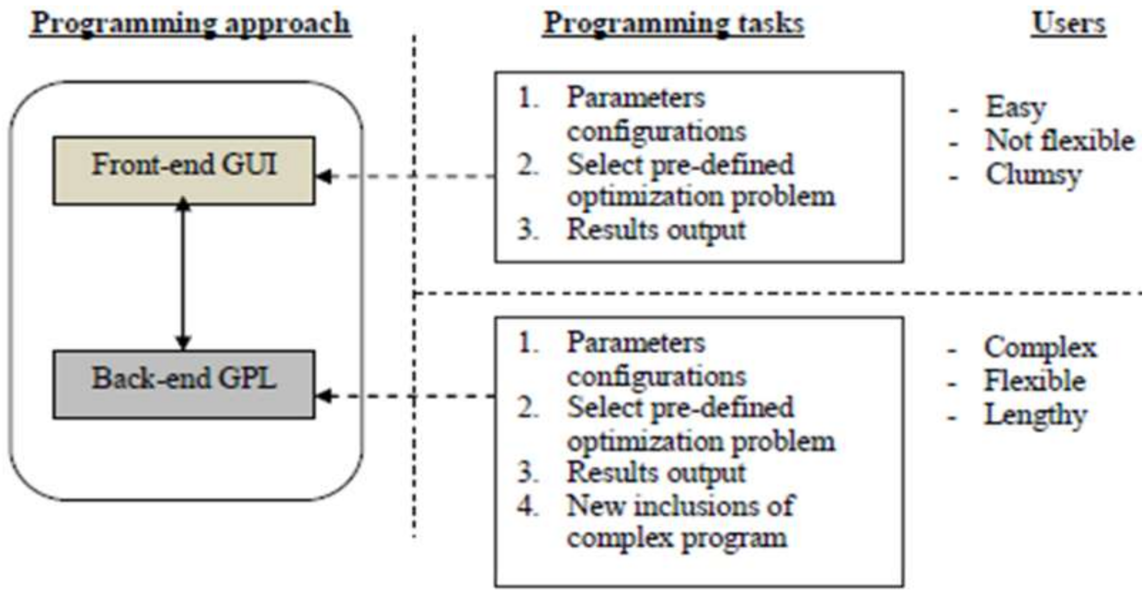
Programming approach
Programming tasks
Users
Front-end GUI
Back-end GPL
1. Parameters configurations
2. Select pre-defined optimization problem
3. Results output
- Easy
- Not flexible
- Clumsy
1. Parameters configurations
2. Select pre-defined optimization problem
3. Results output
4. New inclusions of complex program
- Complex
- Flexible
- Lengthy

Fig.2. Programming approaches with GUI and GPL

Fig. 2 also illustrates that GPL programs can be very lengthy especially in the written form, which disadvantage users to present the programs for publication in a limited number of pages.

Alongside this, GUI is limited in providing the precise textual codes. As discussed previously, to convey information with GUI sometimes demands more pages than textual codes. When considering to support both an easy and concise programming, DSL should be considered as the best programming platform at the frontend of software compared to GUI and GPL. Fig. 3 illustrates the role of DSL in a reusable software.

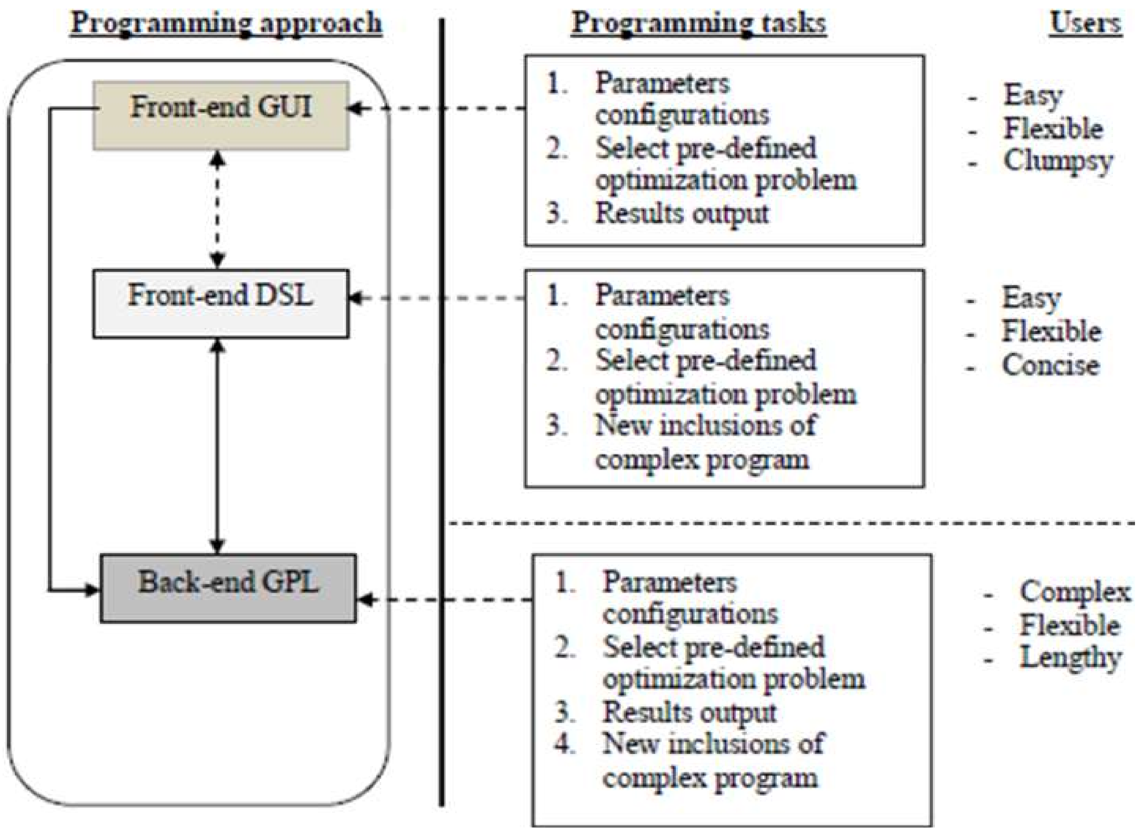
Programming approach
Programming tasks
Users
Front-end GUI
Front-end DSL
Back-end GPL
1. Parameters configurations
2. Select pre-defined optimization problem
3. Results output
- Easy
- Flexible
- Clumpsy
1. Parameters configurations
2. Select pre-defined optimization problem
3. New inclusions of complex program
- Easy
- Flexible
- Concise
1. Parameters configurations
2. Select pre-defined optimization problem
3. Results output
4. New inclusions of complex program
- Complex
- Flexible
- Lengthy

Fig. 3. Programming approaches with GUI, DSL and GPL

It can be depicted in the Fig. 3 that a DSL can be integrated with GUI or operated as an independent mechanism. Although without GUI, a DSL still can enable easy implementation, but more flexible than GUI [24] and more concise than GPL. Since this research attempts to produce a DSL for PSO-GA [22] hybrids, to review the existing DSL for meta-heuristics is a worthwhile. In this paper, the review focuses on the type of back-end software and the loop abstraction.

2.7.Back-End Software for DSL

As previously discussed, the back-end software for a DSL or scripting language can be designed as a software library or software framework. The benefits that have been discussed suggest that it can be useful to know which one is more widely used by the existing DSLs of meta-heuristics and what types of paradigms the software largely provides. Table 1 illustrates the types of back-end software of the exiting DSLs.

Table 1. Back-end software of existing DSLs for meta-heuristics

Software	Software Library	Software Framework	Meta-Heuristics Paradigm
EASEA[8]	✓	X	Single, Hybridization
PPCEA[15]	✓	X	Single
ESDL[13]	X	✓	Single
EAML[16]	✓	X	Single
TEA[17]	✓	X	Single
MDF[10]	X	✓	Single, Hybridization
ParadisEO[11]	X	✓	Single, Hybridization
ECJ[18]	X	✓	Single, Hybridization

There are four out of eight from the existing DSLs used a software framework as a base platform for executing meta-heuristic applications. The preference for software frameworks occurs because of the high flexibility in the software to support software extendibility. With the greater flexibility, Table 1 also shows that the DSLs with a software framework is more applicable to support meta-heuristics hybridization rather than a single implementation. Therefore, this research will use a software framework as the back-end software for the proposed scripting language.

2.8.Loop Abstraction

In the generic meta-heuristics framework, repetitive search is the primary instruction for finding the near-optimal solutions. Some DSLs use common iterative controller blocks such as while and for. Another approach hides the iterative block in a form of variable assignment or parameter configurations. The hiding approach encourages better abstraction of the algorithm and is able to decrease the complexity of the DSL[16]. Table 2 lists the DSLs in relation to the loop controller abstraction.

Table 2.loop abstraction in the existing DSLs for meta-heuristics

Software	Loop Abstraction
EASEA	X
PPCEA	X
ESDL	✓
EAML	X
TEA	✓
MDF	X
ParadisEO	✓
ECJ	X

While loop abstraction benefits in reducing code complexity, only three of the DSLs were designed with this feature. Without the loop abstraction being justified, the reason for long programs remains and appears in some of the DSLs. Regarding the functionality, this is not the reason for neglecting this feature. It has been proven by the ParadisEO that all meta-heuristic paradigms can be supported with loop abstraction. Besides, all algorithms from the class of evolutionary algorithms can also be properly designed in ESDL as well as in TEA with the loop abstraction. The table also highlights that the loop abstraction will be employed for designing the proposed scripting language.

3.THE PROPOSED SCRIPTING LANGUAGE

This section briefly describes the software and compiler architecture of the proposed scripting language constructions.

3.1. Overall Software Architecture

The proposed software consists of three-tier architecture, namely front-end scripting language, intermediate compiler and back-end software framework as presented in the following Fig. 4.

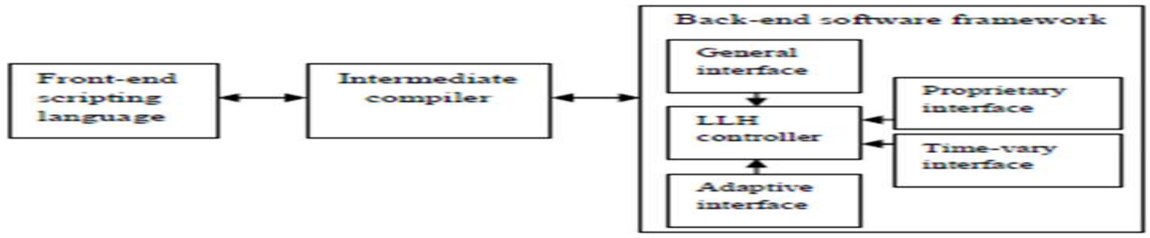

Fig.4. General software architecture

The back-end component is an object oriented software framework that operated as the underlying software to run and execute the JAVA codes for the optimization problem and algorithms. The front-end software is a text editor that can be used to write programs for defining and developing the PSO-GA [23] hybrids with the scripting language. It contains a series of commands that looks like JAVA and common English words.The intermediate compiler translates the scripting language into the relevant JAVA codes for execution.

4. THE EVALUATION

Majority of evaluation on scripting language focused on easiness and flexible aspects. Our previous evaluation has conducted the easiness test on the proposed scripting language[14]. In this paper, the approach for evaluating scripting language is introduced. Conciseness of a programming language is defined in this paper as less of codes but able to produce the desired output. In general, the conciseness can also be presented from the language simplicity[13]. However, to present the quantitative measurement would be more valuable in indicating the conciseness of programming language. In this work, conciseness is calculated by dividing the total ratio of character used in relation to each group of desired task or functionality. The list of the relevant codes written by the proposed scripting language and the main JAVA to achieve the program tasks that employed meta-heuristics [21] hybridization of PSO-GA is given as the following:

4.1.Program Specifications

- The proposed scripting language-*JACIE*is the only one keyword usedto begin and define a program.
- main JAVA-The relevant codes for defining a program in the JAVA software framework are:

  import net.source forge: jswarm_pso.;import java.io.*;*

  public static void main(String[] args)

4.2.Experiment Specifications

- The proposed scripting language can be written as:

  SGCrossoverMutation(Name CMR;ENum 50; Iter 3000;PSize 40);.
- The main JAVA codes can be written in the JAVA software framework as:

  int numberofexperim = 50;double[] chibestf = new double[numofexperim];

  double bestf = 0.0; double stddeviation = 0.0;double ch = 0.0; int converg = 0; int *avgconverg = 0; int totalconverg = 0;*

for(int i = 0; i< numberofexperim; i++)

double[] convergearray = new double [numberofiterations];

system.out.println(”The best fitness is : ”+ c +””+chibest f[c]);

bestf+ = chibest f [c];

system.out.println(”The totalbest fitness : ”+bestf);

system:out:println(”The average best fitness : ” + bestf =numberofexperim);

stddev+ = math.pow(chibestf [K],(bestf =numberofexperim),2);

4.3. General Specifications

- The few codes of the scripting language are:

  SEARCHSPACE(particle,40);PROBLEM(Ackley,min);

- The main JAVA codes that can be written in the JAVA software framework are:

  package net.sourceforge. jswarmpso.Ackley;Particle particles[];

  ParticleUpdate particleUpdate;

  Swarm swarm=new Swarm(40,newMyParticle(),newMyFitnessFunction());

  public MyFitnessFunction().super(false);swarm:initialization();

4.4.Update Specifications

- The only codes to update solutions in the scripting language are:

  Update(inertia[const 0.3];c1[const 1.5];c2[const 1.5];MxP 10.0;MnP 5.0;MxV 10.0;MnV 5.0);

- Some of the relevant JAVA codes in the JAVA software framework are:

  Swarm.setInertia(0.5);swarm.setGlobalIncrement(1.5);

  Swarm.setParticleIncrement(2:0);swarm.setMaxPosition(10);

  Swarm.setMinPosition(5);swarm.setMaxMinVelocity(10);

  Swarm.evaluate();for(int i = 0; i< numberofparticle; i++)swarm.update();

4.5. Crossover Specifications

- The proposed scripting language can be written as the following statement for crossover specifications:

  Crossover(Crossoverrate[const 0.8];

  Crossoperation[pbest];Selectionoperation[rouletewheel]);

- The most relevant JAVA codes in the JAVA software framework are:

  Crossover crossover;Particle particle1,particle2,Offstring;

  selection = newSelection(swarm);particle1 = selection:rouletewheel();

  particle1 = selection.rouletewheel();crossover = new pbestcrossover(particle1,particle2);

  doublecprob=crossover:probability(0.2);

  Offstring=crossover:crossoverallposition(crossoverprob);

4.6.Mutation Specifications

- The scripting language codes for mutation specifications can be written as:

  Mutation(Mutationrate[const 0.25];Mutationoperation[Gaussian]);

- The main JAVA codes in the JAVA software framework can be written as:

  Mutator mutator;mutator = newGaussianmutator(swarm);

  double y = mutator.probability(swarm,0,25);

  mutator.mutatorall position(swarm,y);

Then, the following Equation (1) defined the calculation of conciseness.

$$C = \sum_{i=1}^{n} \left(\frac{1}{h_i} \right) \quad (1)$$

The total conciseness from tasks *i={1..n}*is denoted as *C,* where *n*is the maximum number of specification tasks. In the programs that hybrid PSO-GA [19], *n*is equal to *5.* The first specification is Program and the fifth specification is mutation. Then,*h* is the number of non-space characters exist in the codes. The following Table 3 presentsthe measurement results.

Table 3. The conciseness measurement

Specification Task	The Proposed Scripting Language	JAVA
Program	0.2000	0.0100
Experiment	0.0200	0.0021
General	0.0204	0.0046
Update	0.0120	0.0044
Crossover	0.0143	0.0034
Mutation	0.0208	0.0080

The table shows that the total conciseness of scripting language is 0.29 and and main JAVA is 0.03. This measurement can give an indication that the program with scripting language has more conciseness that the program with JAVA codes.

5. CONCLUSION

Based on several evaluations that have been conducted in our previous research and this paper, it can be concluded that the proposed scripting language provides benefits to support easy programming environment. The scripting language is simple and straightforward to be used and comprehended. As to encourages algorithm designers to use abstractions for algorithmic variation, no conditional and repetition statements are included within the scripting code. This is to allow high level of abstraction with the loop abstraction. Besides ease of use, conciseness allows the whole program tasks and structure to be precisely presented just in a small number of statements or codes.

6. ACKNOWLEDGEMENTS

A lot of thanks to the Ministry of Education Malaysia and Universiti Teknologi MARA for their financial support to this project under FRGS Grant No. FRGS/1/2015/ICT01/UITM/02/1.

7. REFERENCES

[1] Talbi E G.A taxonomy of hybrid metaheuristics.Jounal of Heuristics, 2002, 8(5):541-564

[2] Dhanalakshmy DM, Jeyakumar G.A survey on adaptation strategies for mutation and crossover rates of differential evolution algorithm.International Journal on Advanced Science, Engineering and Information Technology, 2016, 6(5):613-623

[3] Boehm B.A view of 20th and 21st century software engineering.In 28th ACMInternational Conference on Software Engineering, 2006, pp. 12-29

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

[5] Johnson RE, Foote E. Designing reusable classes.Journal of Object-Oriented Programming, 1998, 1(2):22-35

[6] Ousterhout J K.Scripting: Higher level programming for the 21st century. Computer, 1998,31(3):23-30

[7] Fowler M.Domain specific languages. London: Pearson Education, 2010

[8] Collet P, Lutton E, Schoenauer M, Louchet J.Take it EASEA.In International Conference on Parallel Problem Solving from Nature, 2000, pp. 891-901

[9] Mernik M, Heering J, Sloane A M.When and how to develop domain-specific languages.ACM Computing Surveys, 2005, 37(4):316-344

[10] Lau HC, Wan WC, Halim S, Toh K.A software framework for fast prototyping of meta-heuristics hybridization.International Transactions in Operational Research, 2007, 14(2):123-141

[11] Cahon S, Melab N, Talbi E G.ParadisEO: A framework for the reusable design of parallel and distributed metaheuristics. Journal of Heuristics, 2004, 10(3):357-380

[12]Barron D.The world of scripting language. New Jersey: John Wiley and Sons, 2000

[13]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

[14]Masrom S, Abidin SZZ, Omar O.Scripting language constructs for dynamic parameterizations of PSO-GA hybrids. In IEEE International Symposium on Mathematical Sciences and Computing Research, 2015, pp. 18-23

[15]Liu SH, Mernik M, Bryant B R. Parameter control in evolutionary algorithms by domain-specific scripting language PPCEA.In 1st International Conference on Bioinspired Optimization Methods and their Applications, 2004, pp. 41-50

[16] Veenhuis C, Köppen M.XML based modelling of soft computing methods.In J. Benötez, O. Cordón, F. Hoffmann&R. Roy (Eds.),Advances in soft computing. London: Springer, 2003, pp. 149-158

[17]Emmerich M., Hosenberg R.TEA: A C++ library for the design of evolutionary algorithms. North Rhine-Westphalia: University Dortmund, 2001

[18] White D.Software review: The ECJ toolkit.Genetic Programming and Evolvable Machines, 2012, 13(1):65-67

[19] Masrom S, Abidin SZ, Omar N, Rahman AS, Rizman ZI. 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

[20] Ibrahim R, Masrom S, Yusoff RC, Zainuddin NM, Rizman ZI. Student acceptance of educational games in higher education. Journal of Fundamental and Applied Sciences, 2017, 9(3S):809-829

[21] Dahalan WM, Othman AG, Zoolfakar MR, Khalid PZ, Rizman ZI. Optimum DNR and DG sizing for power loss reduction using improved meta-heuristic methods. ARPN Journal of Engineering and Applied Sciences, 2016, 11(20):11925-11929

[22] Indera N I, Yassin IM , 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

[23] Zabidi A, Yassin IM, Tahir NM, Rizman ZI, 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

[24] Mohamad T M, Zairi I R, Wan A K W C, Fadhli D H M F.Fitness cycling device with graphical user interface based on IEEE 802.15.4 transceiver for real time monitoring.Journal of Applied Environmental and Biological Sciences, 2014, 4(12):108-114

How to cite this article:
Masrom S., Abidin S.Z.Z.,Omar N. Ismail Z., Software framework for optimization problems and meta-heuristics based on scripting language.Article title. J. Fundam. Appl. Sci., 2017, *9(5S), 33-48.*