Journal of Theoretical and Applied Information Technology
29th February 2012. Vol. 36 No.2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

AN AUTOMATED SUPPORT FOR EVALUATING ALTERNATIVE DESIGN DECISIONS

[1]ADIL A.A. SAED, [1] WAN M.N. WAN KADIR, [2]HENTABLI HAMZA, [2]ADIL YOUSIF
[1] Department of Software Engineering
[2] Department of Computer Systems and Communication
Faculty of Computer Science & Information Systems, Universiti Teknologi Malaysia
81310 Johor, Malaysia
E-mail: adil_sudan@hotmaill.com, wnasir@utm.my, hentabli_hamza@yahoo.fr, adiluok@gmail.com

ABSTRACT

In recent years, there have been increasing interests in using Component-Based System Development (CBSD) approach to develop large and complex applications. It's believed that satisfying quality attributes is an essential factor for the success of such systems. This paper introduces a novel approach to support the right decisions on selecting design alternatives decision at the early stage of component-based system development. In this approach, we implement Particle Swarm Optimization (PSO) algorithm, as the selected Intelligent Swarm (IS) optimization technique, to explore the design space for optimal design that satisfies the quality requirements. A case study from the Embedded System (ES) domain is used to demonstrate the applicability of the approach. The objectives of this paper are, to aid architects to decide on the level of redundancy within each subsystem based on reliability, and to make trades-off between reliability and performance based on the technical and cost preferences. The results of some preliminary experiments indicate the potentiality of the approach.

Keywords: *Optimization algorithm, metaheuristic, Intelligent Swarm (IS), Particle Swarm Optimization (PSO), Reliability, Redundancy, Design Decision, Embedded System (ES).*

1. INTRODUCTION

Component-based system development (CBSD) enables software architects to reason on the composed structure. This is not only essential for fulfilment of the functional properties but also for satisfying the non-functional properties and software quality as well. In the context of embedded system, the architect needs to know; what is the appropriate composite for optimal design, and what is the appropriate level of redundancy for each subsystem. This problem is classified as NP-complete and many researches have used metaheuristic techniques to tackle it [1-3]. Majority of studies in this area were focused on reliability. Few papers have studied the trade-off with other non-functional properties. In this paper we introduce an approach that uses intelligent swarm technique to assist architects to decide about the design alternatives on the early stage of software development. We are interested in addressing the challenges and complexity in design of the anti-lock breaking (ABS) which is used in the automotive industry. ABS is one of the embedded systems that have replaced the legacy mechanical, electrical and manual systems. The early assessment of reliability is essential especially in embedded system. Prediction of reliability aids architect to identify the inconsistency in the composition and to decide on the number and level of redundant components on the system. This leads to save the cost and time to develop system as well as the quality satisfaction.

Tackling the problem of redundancy using our approach aids decision makers to decide about the number of redundant components and their reliability and cost. The optimal solution can be selected according to the technical and cost preferences. The main focus of this study is to

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

implement single objective optimization algorithm with constraints to search design space. The work under progress is focusing on increasing the efficiency of search design space and addressing conflicting quality attributes using an enhanced intelligent swarm multi-objective algorithm.

The paper is been divided into six sections. The outlines of the related work are presented in Section 2. Section 3 demonstrates optimization problems formulation, intelligent swarm optimization and particle swarm optimization. The detailed information about the case study, system model, evaluation architecture and parameters setting are been illustrated in Section 4. Section 5 presents the results, analysis and discussions. Finally, the conclusion and future work stated in Section 6.

2. RELATED WORK

An interesting survey focuses on optimization of the software architecture has been introduced by Meedeniya et al [4] . The survey divides the optimization problem into three domains, Embedded System (ES), Information System (IS) and general. Some of taxonomies indentified in their survey are used here to evaluate the related work and to put our work in context. The problem domain discussed in the paper concern the application of architecture optimization at design time using Metaheuristics search in the domain of Embedded System (ES). The issues that have been studied on the related work are; quality attribute that should be met, transformation operators which identify the specific architecture problem to deal with such deployment, redundancy and component selection, dimensionality, and finally the class of the metaheuristic algorithm.

Variant quality attributes optimization techniques have been addressed in the recent researches. Reliability was the main focused when the transformation operator is about redundancy problem [5], [2], [6] and [7, 8]. In [9] energy has been studied beside the reliability. Cost and reliability together have studied by Wadekar and Gokhale in [1] using multi-objective algorithm to address the problem of architecture alternative.

Most of current approaches use evolutionary algorithm as a strategy for searching the design space. A single objective Intelligent swarm optimization technique has been used in [7] to address reliability in general domain.

None of the above work has studied the reliability, performance and cost to address the problem of redundancy and alternatives design decision for ES using single objective optimization technique. Our proposed approach uses one of the intelligent swarm optimization algorithms as a promised search technique. Besides, our approach considers performance and cost attributes to provide good opportunity for taking right design decision.

3. OPTIMIZATION ALGORITHMS

This section demonstrates the general formulation of optimization problems. Then we present the swarm intelligent and state its relationship to the family of evolutionary algorithms. Finally particle swarm optimization is described as it will be employed in our approach.

3.1 PROBLEM FORMULATION

This section describes the general optimization problem. Let us assume an optimization problem with a single objective function Y denoted as $f(x)$. f is an objective function to be minimized or maximized, the minimization problem is taken as general, where

$Y = \min f(x), x \in X$

Where Y is an objective space that contains an objective vector y. X is parameter space that contains x, which is a decision vector. n is a number of decision vectors.

The function has domain constraint. In addition, strategy of how to deal with the velocity is needed, (example of strategy: delete, edit, modify, and penalty). The domain constraint is a set of constraints, which is denoted as:

$Dc = \{dc_1, \ldots, dc_n\}$

Thus, the minimization problem expressed as follows:

$Y = f(x), x \in \{x_1, x_2, \ldots., x_n\}$

The above minimization problem is subjected to the following constraints

$dc_i(x) > 0$

$dc_i(x) = 0$

Example:

$\min [(x_1-3)^2 + (x_2-1)^2]$ (1)

Subject to

$x_1^2 - x_2^2 < 0$ (2)

$x_1 + x_2 = 1$ (3)

For any optimization, modeling the problem is essential and good mathematical model of the optimization problem is needed. Modeling means the process of identifying objective function, variables and constraints.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

3.2 INTELLIGENT SWARM OPTIMIZATION

Meta-heuristics are originated and inspired by natural process and creature's behaviors to solve complex real world problems. Optimization is at the heart of many natural processes such as; Darwinian evolution, social group behavior and foraging strategies. The last two decades have witnessed notable increasing in the domain of nature-inspired search and optimization algorithms. Recently, these techniques are applied to variant problems. Evolutionary computing methods and the swarm intelligence algorithms are the main common groups of methods represent the field[10].

Meta-heuristics evolutionary techniques such as Genetic Algorithms (GAs) have proven their usefulness to solve the problem of spanned design space. In recent researches Swarm Intelligent (SI) techniques such as Firefly optimization algorithm [11] Particle Swarm Optimization (PSO) [12], [13], [14] and alternative search technique, often performed better than many evolutionary techniques such as GA when applied to various problems [15, 16]. Evolutionary techniques need to handle the population movement; therefore, they are less fast in discovering optimal solutions. Furthermore, evolutionary algorithms may have a memory to store previous status; this may help in minimizing the number of individuals close to positions in candidate solutions that have been visited before, but it may also slow to converge since successive generations may die out. In this paper we introduce the use of particle swarm optimization to search design space. Our objectives are; to search the design space, automatically generate and evaluate the design alternatives. Then architects can reason on the provided options and choose the optimal solution that satisfies the specified quality requirements. Next section presents some details about PSO.

3.3 PARTICLE SWARM OPTIMIZATION (PSO)

Particle Swarm Optimization (PSO) is one of the swarm intelligent (SI) optimization methods, it's an adaptive optimization method introduced in 1995 by Kennedy and Eberhart [17, 18]. The method inspired by social behavior of swarms such as bird flocking or fish schooling. In PSO, particles never die, and particles are considered as simple agents that move and interact through the search space and record the best solution that they have visited. Each particle represents a candidate solution in the solution search space and each particle has a position vector and velocity. The behavior of the particles is subjected to their capability to train from their past personal experience and from the success of their neighbor to adapt the flying speed and direction to the target. However, particles manage the current position, velocity and personal best position. Beside the personal best solution, the swarm is targeting the global best solution

Let us assume a swarm with N particles flying in design space consists of D-dimensional. Each particle i, re-position itself xi in the direction of the global optimum base on the following two factors;

The best position achieved by itself (pBest I) expressed by pi = (p_i1, p_i2, . . . , p_iD). And the best position achieved by the whole swarm (gBest), for a given subset of the swarm which expressed by pg = (pg1, pg2. . . pgD). The difference between the current position of the particle i and the best position of its neighborhood expressed by (pg − xi).

For simplification the two equations are:

```
v[i] = w*v[i] + c1 * rand() * (pbest[i] -
present[i]
) + c2 * rand() * (gbest[i] - present[i])        (4)

present[i] = persent[i] + v[i]                   (5)
```

Where w is inertia weight, c1 and c2 are learning factors (weights)

The parameters used in algorithm are:

c_1: acceleration factor related to gbest =3

c_2: acceleration factor related to lbest =1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

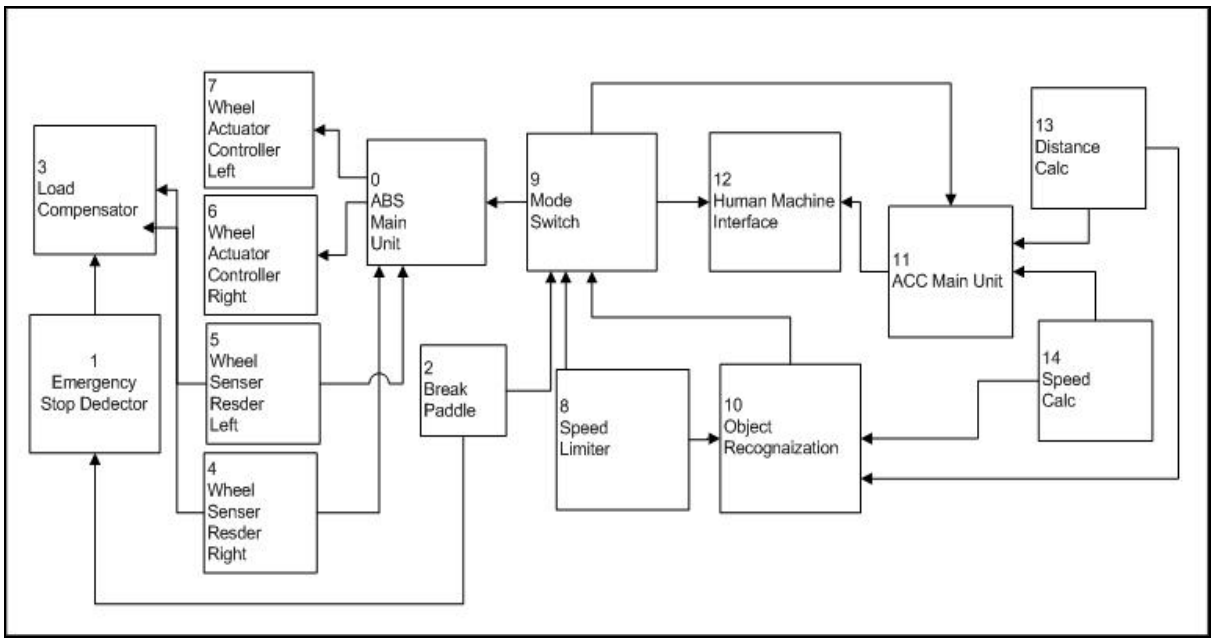

Figure 1: System Structure of Anti-lock Brake System (ABS)

rand1 ( ): random number between 0 and 1
rand2 ( ): random number between 0 and 1

The following is the algorithm for PSO optimization.

For each particle
Initialize particle with feasible random number
END
Do
For each particle
Calculate the fitness value
If the fitness value is better than the best fitness value (pBest) in history
Set current value as the new pBest
End
Choose the particle with the best fitness value of all the particles as the gBest
For each particle
Calculate particle velocity according to velocity update equation
Update particle position according to position update equation
End
Until (maximum number of iterations) or (condition be satisfied).

4. CASE STUDY

In this paper, PSO algorithm has been applied as search technique to the case study of Anti-lock Brake System (ABS). An analytic model has been used to evaluate each design alternative. In this section, we demonstrate the applicability of the method and the phases of application. First we give general idea about the ABS and its components, then the system model for the case study are demonstrated, after that the paper describes the models and functions used to evaluate the quality for each candidate. Then the parameters settings are outlined. Finally the results from the experiment are analyzed and discussed.

4.1 ABS SYSTEM OVERVIEW

ABS system currently used in most of modern cars to decrease the risk generated from the skidding and loss of control which mostly caused by locked wheels during breaking. It is well known that wheels will slip and lockup during severe braking or when braking on a slippery road surface (wet, icy, etc.). This usually causes a long stopping distance and sometimes the vehicle will lose steering stability. The objective of ABS is to prevent wheels from lockup and achieve minimum stopping distance. To achieve that, proper wheels

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

rotations during break operations is required. Therefore, ABS manipulates the wheels slip so that the maximum friction force is obtained and the steering stability is maintained.

. Adaptive Cruise Control (ACC); is introduced to prevent crashes by slow down the speed of wheels whenever another car is sensed in the front. Figure1 shows the main components of the composite system and the interactions as well. The following paragraph states some component's tasks and their interaction behaviors.

ABS Main Unit is the major decision making unit regarding the breaking levels for individual wheels. Load Compensator unit assists with computing adjustment factors from the wheel load sensor inputs. Components 4 and 5 are components that communicate with wheel sensors. Components 7 and 8 are used to control the break actuators. Break Paddle reads from the paddle sensor and sends the data to the Emergency Stop Detection unit. Wheel Sensors, Speed Limit, Object Recognition, Mode Switch and Human Machine Interface are contributed to the triggering of the service. Break Paddle reads from the paddle sensor and sends the data to the Emergency Stop Detection unit.

Next section describes the system model. The Serial-Parallel system model is discussed to be used to model the ABS system.

4.2 SYSTEM MODEL

The domain of Embedded System (ES) as an example of CBS is the focus of this paper. Automotive field is one of ES sub-domain in which each component represented by Electronic Control Unit (ECUs) and all of them are communicated to each other during the execution by bus. The ECUs manipulate one or more electrical system or subsystem, and the embedded software inside each is committed to satisfy certain function, such as receiving the speed of individual wheel sensors, and then send a signal to controller if one wheel loses traction in order to limit the break-force.

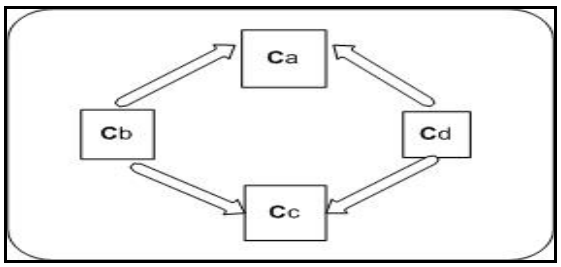

Figure 2: Parallel view point for the Subsystem-interaction model

Therefore, such systems can be represented as parallel system model. Since each component might have number of redundant components, then system have serial dimension. Thus it could be modeled as Serial-Parallel system. We can look at the system from two different views, inner view and outer view. Outer view represents subsystem; Figure2 shows interaction between sub systems C_a, C_b, C_c, and C_d. The second view, inner view, shows the details inside each subsystem, on other word, it describes the level of redundant within subsystems as depicted in Figure 3.

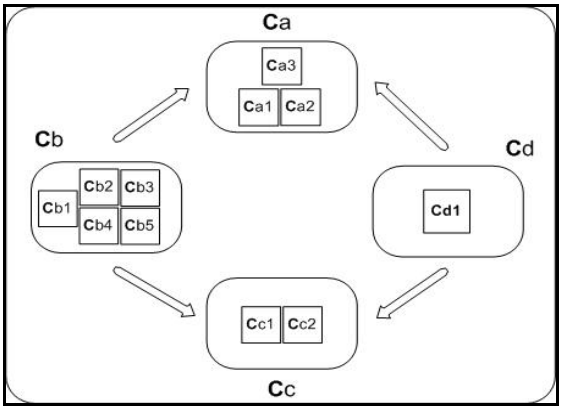

Figure 3: Serial-Parallel view of the system model

In such systems, each candidate has different level of redundancy components. For example in the following candidate, the level of redundancy for each subsystem respectively is; 3, 4, 1, and 0 components.

C_a (3), C_b (4), C_c (1), C_d (0)

The search technique is responsible to generate candidates from the design space. Then the quality attributes for each candidate could be evaluated. ABS system can be modeled as Serial-Parallel system. Reliability, cost, and response time as

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

quality attributes are studied based on the interactions between subsystems. In this paper the interactions between the ABS' components are formulated using Discrete Time Markov Chain (DTMC). The indicators of quality attributes are calculated based on formal notations as stated in the next section.

4.3 EVALUATING DESIGN ALTERNATIVES

Throughout the searching process, many candidates architecture are generated and thus must be evaluated. This section defines the indicators by which the quality attributes of each candidate are measured. Then the candidate evaluated according to these results. The relation between the quality attributes, candidates from design space, and the quantities of these attributes denoted as: [3, 9]

$$Q: A \rightarrow IR^3$$

Where Q denotes quality attributes, A is the area of design space from which all candidates are coming from, IR^3 is the real set to express the quantities of the quality attributes.

$$Q(cand) = (R_{cand}, C_{cand}, RT_{cand}).$$

Q(cand) is quality attribute for a candidate cand. R_{cand}, C_{cand}, and RT_{cand} are the reliability, cost and response-time for this specific candidate (cand) respectively. For each candidate there is subsystem denoted as sub, each sub has C(i) component, therefore the level of redundant components in each subsystem is;

$$a(i) + 1 \text{ (for } i=0,1,3, \ldots h) \quad (3)$$

h is the maximum number of redundancy.

Reliability

Reliability is calculated for each generated candidate during the searching design space by estimating the reliability of each single component.

There are three different techniques to calculate reliability; path-based, state-based and mixed of both. In the path-based the reliability is calculated with execution path. The execution path has start point and an end point. The state-based technique estimates the reliability using analytic model. The architecture is represented using control flow graph. The transformation of control graph can be characterized using Markov model. Composite model and hierarchical model are branched from the state-based approach. Variant architecture representation with variant failure model used to represent the applications has been proposed. Discrete Time Markov Chain (DTMC) with probability of reliability model, Continuous Time Markov Chain (CMTC) with Failure rate, and Simi-Markov Time Chain (SMTC) with Failure intensity model. The reliability model is used to calculate reliability of the system from its components. In State-based technique, which is commonly used, components are treated as black box, and existence of loop does not prevent the calculation of reliability. There for this work uses state-based model, composite model, to obtain estimation of reliability for the system.

Two steps are needed to calculate estimation of reliability for the system; the initial step is to calculate the reliability of sub-systems (sub) as follow.

The reliability for sub-system

$$R_{sub}(ci) = (1 - (1 - R(c_i))a(i)+1) \quad (6)$$

Where

$$R_{sub}(c_i) = e^{-\lambda \cdot st(c_i)}. \quad (7)$$

λ is the failure rate.

The second step is to calculate the estimation of overall system with the considering of number of visits v(ci) to each sub-system or component:

$$R_{cand} \approx \prod (R_{sub}(ci))v(ci) \quad (8)$$

Number of visits can be calculated from the equation below

$$v(c_i) = q_0(c_i) + \sum (v(c_j) \cdot p(c_j, c_i)) \quad (9)$$

q_0 is the probability that the execution is initiated from that component. $P(c_i,c_j)$ is the transform probability which indicates to the probability of executing cj after transmitting from ci. The above equation can be solved by using recursive function.

Cost

The cost for candidate cand is donated as C_{cand}. For the sake of simplicity, a simple equation is used to calculate the cost of each candidate base on the cost of individual component and the number of redundant components.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

$$C_{cand} = \sum Cst(c_i) \cdot (a(c_i) + 1) \quad (10)$$

Response-time

Three factors impact on response-time, sojourn time, estimated time per visit $et(c_i)$, and redundancy over head. The estimated time taken for a single visit of component which is known as Sojourn Time per visit $st(c_i)$. And there is an additional execution time caused because of increasing in the redundancy level, this factor known as redundancy overhead $\delta(c_i,c_j)$. The sojourn time which is denoted the estimated time for single component per visit is calculated as follows;

$$st(c_i) = et(c_i) + \sum \delta(c_i,c_j) \cdot a(c_i) \quad (1)$$

So, response-time RT for the candidate cand is calculated as follows. [3, 9]

$$RT(_{cand}) = \sum st(c_i) \cdot v(c_i) \quad (12)$$

4.4 PARAMETERS SETTING

The parameters and settings for the algorithm are presented in this section, as well as parameters to apply the case study.

Delphi 5 has been used as software programming language for implementing the experiment. The program was run in computer with 3.7 GHz processor and 2 GB RAM. The parameters for the objective functions, which have been identified in previous section, are obtained from literature. The values for PSO's parameters are stated below:

Inertia value: w = 0.4

Upper and lower limits of velocity are [4, -4].

c1 and c2 are 2 and 4 respectively.

Number of iterations and population size are adjusted via try and error. Next section discusses their impact on to the efficiency of the algorithm.

In Table 1 values for Cst(ci), q0, λ(ci), and P(xi,xj) are presented respectively. Table 2 shows parameters values for $et(c_i)$ and are $\delta(c_i,c_j)$. Descriptions of these parameters are shown below;

- Execution initiation probability - $q_0(c_i)$
- Cost of component - $Cst(c_i)$
- Failure Rate - $\lambda(c_i)$.
- Execution transfer probability Matrix - $p(c_i, c_j)$
- Redundancy overhead $\delta(c_i,c_j)$.
- Estimated time per visit $et(c_i)$.

5 RESULTS AND DISCUSIONS

Different population size has been studied to analyze the impact of population size on the accuracy of the results, Figure 4 illustrates a case where (10) population size compared against (20) population size with the same number of iterations (8).

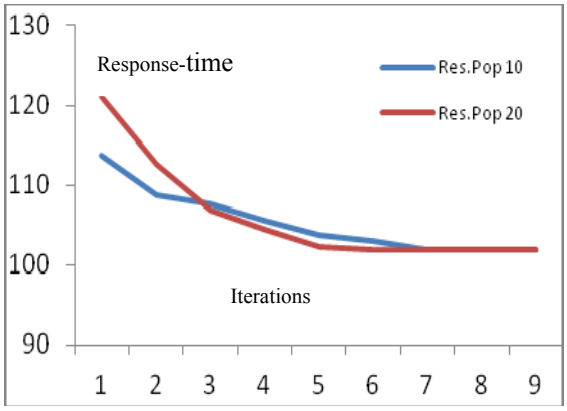

Figure 4: Comparison of different population size

It can be seen that, the bigger the population size, the wider the design space is covered and the quality of solution increased.

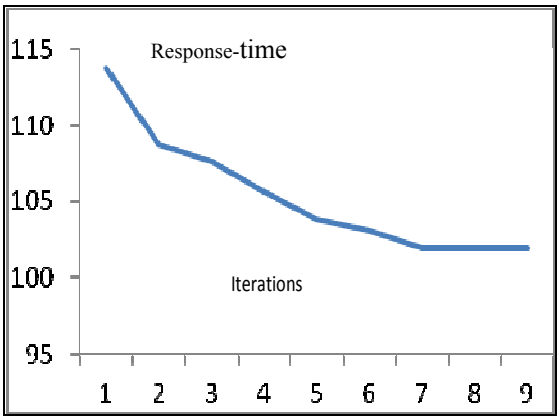

Figure 5: Effect of population size - 8 iterations

As plotted in Figure 5 and Figure 6, we observed that decreasing number of iteration from 8 to 5 affected the quality of solution. We used a similar number of populations for both 5, and 8 iterations to study the output for the minimum response-time.

We found that, the population size as well as number of iterations in our proposed algorithm are critical parameters and have obvious impact on the quality of solutions. The same setting has been

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

applied to reliability to validate the assumption, see Figure7. In this figure it can be seen that, the best solution has appeared starting from the third iteration.

Table 1: Parameters-1

CompID	Cst ($)	q0	λ	p(C i ,Cj)														
				C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}
C_0	15	0	4 · 10−6	0	0	0	0	0	0	0.5	0.5	0	0	0	0	0	0	0
C_1	8	0	6 · 10−6	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
C_2	10	0.3	5 · 10−6	0	0.75	0	0	0	0	0	0	0	0.25	0	0	0	0	0
C_3	10	0	8 · 10−6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
C_4	8	0.1	8 · 10−6	0.7	0	0	0.3	0	0	0	0	0	0	0	0	0	0	0
C_5	8	0.1	8 · 10−6	0.7	0	0	0.3	0	0	0	0	0	0	0	0	0	0	0
C_6	8	0.1	8 · 10−6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
C_7	8	0.1	8 · 10−6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
C_8	10	0.1	5 · 10−6	0	0	0	0	0	0	0	0	0	0.6	0.4	0	0	0	0
C_9	8	0	5 · 10−6	0.2	0	0	0	0	0	0	0	0	0	0	0.4	0.6	0	0
C_{10}	12	0	5 · 10−6	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
C_{11}	14	0	4 · 10−6	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
C_{12}	15	0	7 · 10−6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
C_{13}	15	0	3 · 10−6	0	0	0	0	0	0	0	0	0	0	0.5	0.5	0	0	0
C_{14}	15	0	3 · 10−6	0	0	0	0	0	0	0	0	0	0	0.5	0.5	0	0	0

Table 2: Parameters- 2

CompID	et (ms)	δ(ci,cj) (ms)														
		C_0	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}	C_{11}	C_{12}	C_{13}	C_{14}
C_0	50	0	0	0	1	1	1	3	2	0	0	0	0	0	0	0
C_1	30	0	0	1	3	0	0	0	0	0	0	0	0	0	0	0
C_2	10	0	1	0	0	0	0	0	0	0	3	0	0	0	0	0
C_3	40	2	2	0	0	2	2	0	0	0	0	0	0	0	0	0
C_4	10	2	0	0	3	0	0	0	0	0	0	0	0	0	0	0
C_5	10	2	0	0	3	0	0	0	0	0	0	0	0	0	0	0
C_6	10	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
C_7	10	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
C_8	20	0	0	0	0	0	0	0	0	0	4	3	0	0	0	0
C_9	20	0	0	2	0	0	0	0	0	0	0	0	2	2	0	0
C_{10}	50	0	0	0	0	0	0	0	0	2	1	0	0	0	2	1
C_{11}	40	0	0	0	0	0	0	0	0	0	0	1	0	3	1	1
C_{12}	40	0	0	0	0	0	0	0	0	0	1	0	2	0	0	0

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

C_{13}	50	0	0	0	0	0	0	0	0	0	0	3	1	0	0	0
C_{14}	50	0	0	0	0	0	0	0	0	0	0	2	3	0	0	0

Table 3: Result for two optimal solutions

Solutions	All Components														
Sol 1	**1**	**3**	**0**	**4**	**4**	**4**	**1**	**4**	**4**	**2**	**4**	**3**	**1**	**3**	**0**
Sol 2	**2**	**3**	**1**	**2**	**3**	**1**	**2**	**4**	**3**	**2**	**2**	**1**	**1**	**0**	**0**

Therefore we have to consider that parameters in order to obtain more efficient results.

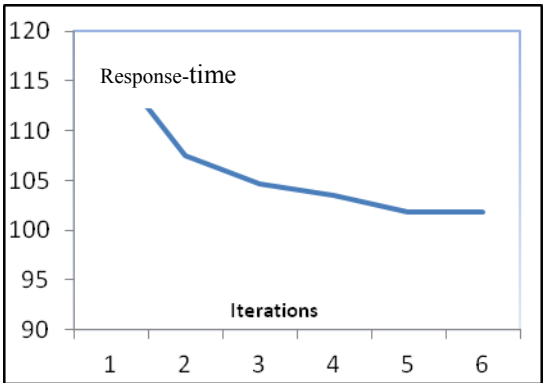

Figure 6: Effect of population size - 5 iterations

The parameters mentioned above (number of iteration and size of population) were used to support design decision and made trade-off between reliability and cost. Also trade-off between reliability and response time has been studied. For the first case, reliability and cost, we took cost as an objective function and Reliability as constraint (Reliability>=0.99995). Figure 8 illustrates this case.

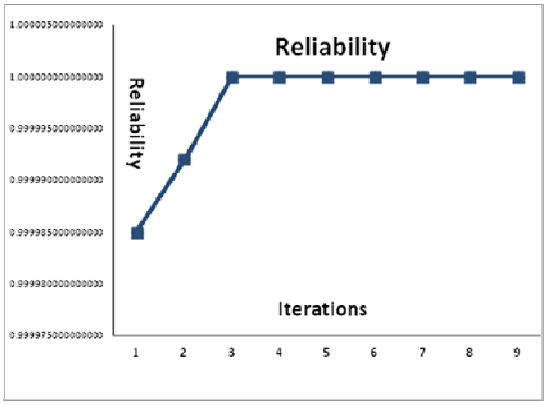

Figure 7: Reliability, 10 populations, 8 iterations

Table 3 illustrated the level of redundant for sub systems in two solutions. Table 4 states number of components, cost, and reliability for solution 1 and solution 2 respectively.

Table 4 shows that, high reliability can be obtained with less number of redundant components and lower cost. The first row states that 27 total number of components has given 0.999969956 of reliability with 427 $ cost, against solution 1 which provide lower reliability and higher cost.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

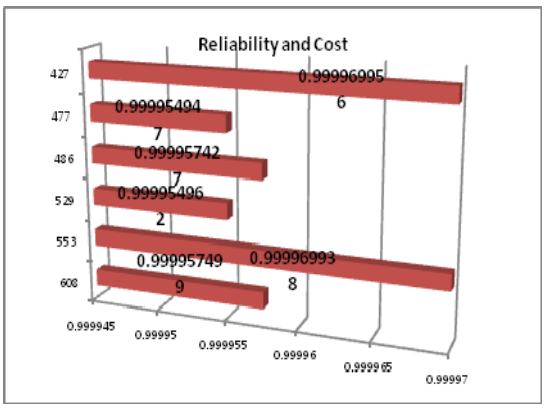

Figure 8: Optimizing Reliability function using cost as constraint

From Figure 8 we observe that, decision maker can easily decide on the components and level of redundancy for each component. The result indicated that high reliability with low cost could be achieved with minimum number of redundant component.

Table 4: Resulted Reliability, Number of component and Cost for Two Closed Alternatives

	N. Comp	Cost	Reliability
Sol.1	38	553	0.999969938
Sol.2	27	427	0.999969956

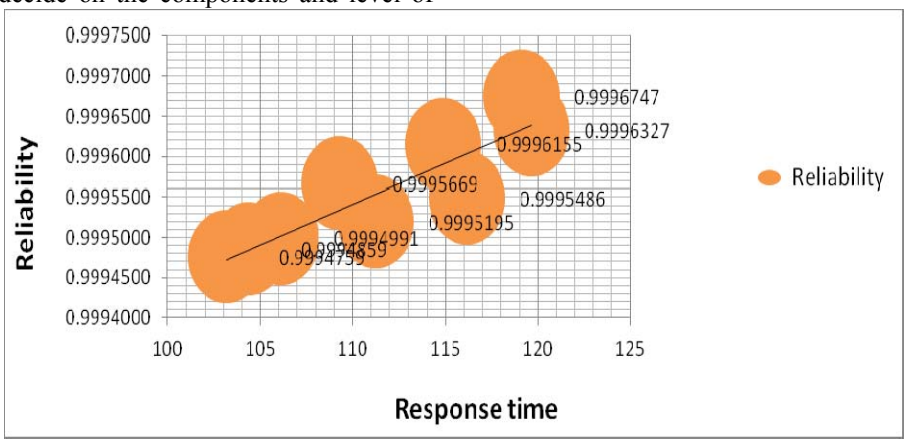

Figure 9: Relation between Response time and Reliability

Table 5: Reliability against Response Time

Response Time	Reliability
119.11	0.999674741907323
119.64	0.999632725452085
111.23	0.999519515469154
116.16	0.999548601909360
114.89	0.999615464113025
109.29	0.999566867998705
106.11	0.999499124039538
104.35	0.999485930355737
103.15	0.999475934647665

The graph plotted in Figure 9, depicts the relationship between two quality attributes, reliability and response-time. From the figure we can see how the conflict could be managed. It can be seen that, best response-time available with low reliability and vice versa. Assume that there is low limit for reliability (ex. Reliability must be not less than 0.99960). Two closest options to the optimal design are highlighted in Table 5. The options' reliability are0.99967and 0.99961with response time 119.11 and 114.89 respectively. The architect can decide to choose the one has the better response-time or the one with the better reliability.

6 CONCLUSION AND FUTRE WORK

This paper introduced an automatic support on evaluating design alternatives. The approach is based on particle swarm optimization (PSO) to aid architect take right design decision. Anti-lock Brake System (ABS) was used as a case study to demonstrate the applicability and the usefulness of the approach. The system for the ABS has been modeled, the functions for evaluating the design

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

alternatives of each candidate were developed, and the single objective optimization algorithm was used to generate and evaluate each design candidate. The results indicate that, the approach can be used to achieve the architect's objectives since he/she can decide on design option according to the technical and financial preferences. Fewer parameters without complexity in configuration and identifications of constraint have been applied. We have observed that, it's easy to symbolize the architecture alternatives as an optimization problem and thereby it's proved that the proposed approach is applicable and suitable to solve such problems.

PSO has proved to be an efficient optimization method for single objective optimization, and more recently it has shown promising results for solving multi-objective optimization problems in different areas. Our ongoing research aims to propose multi-objective Particle Swarm Optimization (MOPSO) approach for information system and embedded system to support the right design decisions in developing software systems that have conflicting quality attributes. In this paper reliability, cost, and response time have been considered, more quality attributes planned to be added such as availability. Furthermore, expected discount in the cost as the number of requested component increased will be considered as well.

The specifications of architecture could be implemented using AADL (Architecture Analysis and Design Language). AADL Model parser can interpret and extract system specification from designed on AADL specifications [3]. Therefore, AADL could be used to develop a tool in order to facilitate the implementation of the approach.

ACKNOWLEDGEMENT

The authors would like to thank Malaysian Ministry of Higher Education (MOHE) for their financial support under the Fundamental Research Grant Scheme (FRGS) Vot No. 78601.

REFRENCES:

[1] Wadekar, S.A. and S.S. Gokhale, Exploring Cost and Reliability Tradeoffs in Architectural Alternatives Using a Genetic Algorithm, in Proceedings of the 10th International Symposium on Software Reliability Engineering. 1999, IEEE Computer Society. p. 104.

[2] Ouzineb, M., M. Nourelfath, and M. Gendreau, Tabu search for the redundancy allocation problem of homogenous series–parallel multi-state systems. Reliability Engineering & System Safety, 2008. 93(8): p. 1257-1272.

[3] Meedeniya, I., et al., Reliability-driven deployment optimization for embedded systems. Journal of Systems and Software, 2011. 84(5): p. 835-846.

[4] Meedeniya, I. OptimizationSurvey. 2011 6 DECEMBER 2009, AT 17:34. [cited 2011 20-11-2011]; Available from: https://sdqweb.ipd.kit.edu/wiki/OptimizationSurvey

[5] Coit, D.W. and A.E. Smith, Solving the redundancy allocation problem using a combined neural network/genetic algorithm approach. Computers & Operations Research, 1996. 23(6): p. 515-526.

[6] Tian, Z., G. Levitin, and M.J. Zuo, A joint reliability–redundancy optimization approach for multi-state series–parallel systems. Reliability Engineering & System Safety, 2009. 94(10): p. 1568-1576.

[7] Chen, T.-C., Penalty Guided PSO for Reliability Design Problems, PRICAI 2006: Trends in Artificial Intelligence, Q. Yang and G. Webb, Editors. 2006, Springer Berlin / Heidelberg. p. 777-786.

[8] Amari, S.V. and V. Hegde. New allocation methods for repairable systems. In Reliability and Maintainability Symposium, 2006, RAMS Annual

[9] Meedeniya, et al., Architecture-Driven Reliability and Energy Optimization for Complex Embedded Systems Research into Practice – Reality and Gaps, G. Heineman, J. Kofron, and F. Plasil, Editors. 2010, Springer Berlin / Heidelberg. p. 52-67.

[10] Website, Swarm and Evolutionary computation. 2011 [cited 2011 13-11-2011]; Available from: http://web.mysites.ntu.edu.sg/epnsugan/PublicSite/Shared%20Documents/Swarm%20and%20Evolutionary%20Computation_2.pdf

[11] Adil yousif, a.h.a., sulaiman mohd nor, adil ali abdelaziz Scheduling jobs on grid computing using firefly algorithm. 2011, Journal of Theoretical and Applied Information Technology, 14570 -JATIT, p 155 - 164 Vol 33. No.2.

[12] Gudise, V.G. and G.K. Venayagamoorthy. Comparison of particle swarm optimization and backpropagation as training algorithms for

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

neural networks. Proceedings of the IEEE in Swarm Intelligence Symposium, 2003.
[13] Boeringer, D.W. and D.H. Werner, Particle swarm optimization versus genetic algorithms for phased array synthesis. IEEE Transactions on Antennas and Propagation, 2004. 52(3): p. 771-779.
[14] Eberhart, R. and Y. Shi, Comparison between genetic algorithms and particle swarm optimization, in Evolutionary Programming VII, V. Porto, et al., Editors. 1998, Springer Berlin / Heidelberg. p. 611-616.
[15] Liang, J.J., et al., Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. IEEE Transactions on Evolutionary Computation, 2006. 10(3): p. 281-295.
[16] Qasem, S.N. and S.M. Shamsuddin, Radial basis function network based on time variant multi-objective particle swarm optimization for medical diseases diagnosis. Applied Soft Computing, 2011. 11(1): p. 1427-1438.
[17] Kennedy, J. and R. Eberhart. Particle swarm optimization. in Neural Networks, 1995. Proceedings., IEEE International Conference .
[18]Kennedy, J. and R.C. Eberhart. A discrete binary version of the particle swarm algorithm. in Systems, Man, and Cybernetics,. Computational Cybernetics and Simulation., 1997 IEEE International Conference.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

AUTHOR PROFILES:

Mr. Adil A.A Saed graduated from University of Khartoum. Currently he is PhD student in Software Engineering Department – Universtiti technologi Malaysia. His research interests are component-Base system, model driven development, quality attributes of CBS, Optimization technique and in specific, Particle Swarm optimization - PSO. He is a member of Sudanese Students Research Group (SSRG) and the Ssoftware Engineering Research Group-UTM

Mr. Wan M.N. Wan Kadir is an Associate Professor in Software Engineering Department – Universtiti technologi Malaysia. He has a PhD. in the field of Software Engineering from the University of Manchester. His research interest covers various Software Engineering knowledge areas based on the motivation to reduce the cost of development and maintenance as well as to improve the quality of large and complex software systems.

Mr. Adil Yousif Received the B.Sc. and M.Sc. from University of Khartoum, Sudan. He works as a lecturer at Faculty of Computer Science and Information Technology, University of Kassala, Sudan. Currently he is a PhD candidate at the Faculty of Computer Science & Information Systems, Universiti Teknologi Malaysia, UTM. He is also a member of Pervasive Computing Research Group PCRG, in UTM. His research interests include computer networks, distributed systems, grid computing and optimization mechanisms.