December 2006

A Design Framework to Information System Functionality Defense through Diversity

Jingguo Wang
State University of New York at Buffalo

Raj Sharman
State University of New York at Buffalo

Stanley Zionts
State University of New York at Buffalo

A Design Framework to Information System Functionality Defense through Diversity

Jingguo Wang
Management Science and Systems
School of Management
State University of New York at Buffalo
Buffalo, NY 14260-4000
wang7@buffalo.edu

Raj Sharman
Management Science and Systems
School of Management
State University of New York at Buffalo
Buffalo, NY 14260-4000
rsharman@buffalo.edu

Stanley Zionts
Management Science and Systems
School of Management
State University of New York at Buffalo
Buffalo, NY 14260-4000
szionts@buffalo.edu

ABSTRACT

Diversification is one of the most effective approaches to defend information systems against worms, virus, and malicious behavior. However, how to design an integrated information system to achieve effective diversity is still a challenge due to combinatorially exploded solution space and multiple conflicting design objectives. In the paper, we present a systematic framework employing a combination of configuration evaluation through controlled system simulations and a neural network based feedback learning mechanism to explore the solution space, thus achieving an effective design solution for the integrated system. A simulation model is employed to evaluate design solutions, and an artificial neural network is trained to approximate the behavior of the system using system feedback. Guided by the trained neural network, a multiple-objective evolutionary algorithm (MOEA) is proposed to search the solution space and identify potential good solutions. The MOEA incorporates the concept of Herbert Simon's satisficing. It integrates the decision maker's preference and uses his/her aspiration level for the performance as its search direction. Potentially good solutions are then evaluated through simulation. The newly obtained simulation results can refine the neural network. The exploration process stops until the result convergences or a satisfied solution is found. We demonstrate and validate our framework through a case study.

Keywords

Information Assurance; System Functionality Defense; Diversity/Diversification; Artificial Neural Network; Multiple-Objective Evolutionary Algorithm (MOEA); Satisficing.

INTRODUCTION

Engineered attacks (such as viruses, hackers, and other threats) on the Internet's software infrastructure happen all the time. Viruses (like 'I love you') and warms (like 'Code Red' and 'Nimda') caused several billions dollars in damage.

According to the Ernst & Young Global Information Security Survey (Ernst&Young, 2004), one of the realities is that there is now a greater reliance on common software platforms, which extends the exposure of the entire network to common vulnerabilities. Most engineered attacks are oriented to a specific operation system, application or database server. Very few of them cut across platforms or systems. A study by the mi2g Intelligence Unit (mi2g, 2004) reveals that overall for January 2004, the most attacked Operating System for online servers was Linux (80%) followed by Windows (12%) and then by BSD and Mac OS X (3%). When confronting a certain type of attacks, the applications on the specific operation system are more vulnerable than the applications on other operation systems(Bain et al., 2001).

Replication and redundancy are used to enhance reliability and improve performance. However, the replicas are "homogenous". When confronting attacks, they are more likely to fail at the same time. A monoculture, whether it is in biological terms or in computing terms, has been shown to be inherently dangerous to the systems (Geer et al., 2003, Stamp, 2004). Diversity is currently the best defense against attacks on the flaws of any particular software component (Bain et al., 2001, Benjamin et al., 1998, Geer et al., 2003, Stamp, 2004). Diversity brings "heterogeneity" into the system (Zhang et al., 2002). It improves not only the performance of the system, but also the survivability of the system[1] when it confronts an attack (Bain et al., 2001).

In the paper we present a systematic framework to achieve effective functionality defense through diversity. Our framework employs a combination of configuration evaluation through controlled system simulations and a neural network based feedback learning mechanism in the exploration of the combinatorially exploded design space. A simulation model is employed to evaluate solutions, and an artificial neural network is trained to approximate the behavior of the system with system feedback. The neural network mechanism also demonstrates the tradeoffs among the performance metrics according to different design parametric settings. Guided by the trained neural network, a multiple-objective evolutionary algorithm (MOEA) is proposed to explore the solution space and identify potential good solutions. The proposed MOEA integrates the decision maker's preference, and incorporates Herbert Simon's concept of satisficing (Simon, 1996). We parameterize the decision maker's aspiration level into the proposed MOEA as its search direction. Potential good solutions are then evaluated through simulation. The simulation results can be used to refine the neural network. The exploration process stops until the result convergences, or a satisfied solution is found.

The framework can be an effective decision support tool for a system designer in systematically exploring design options and selecting an appropriate design configuration that best meets the design objectives. The framework incorporates an integration of several techniques such as MOEA, simulations and neural network based feedback learning. This approach follows a recent trend in research on using simulation modeling for large-scale system optimization by integrating with a meta model to approximate system behavior (April et al., 2003, Bhattacharjee et al., Forthcoming, Fu, 2002, Fu et al., 2000, Glover et al., 1999, Shantikumar and Sargent, 1983, Shao and Rao, 2001, Shim et al., 2002, Wang et al., 2005). We use a numeric example to demonstrate and validate the proposed framework, and the result is promising.

RELATED LITERATURE

The notion of using diversity to limit correlated risks is a widely accepted strategy in many fields. Redundancy without diversity is often argued to be useless against systematic attack and diversity to be of dubious value (Littlewood and Strigini, 2004). Bain et al. (2001) examined several widespread computer attacks to understand the effect of diversity on maintaining the integrity, and hence survivability, of information systems. Methods to achieve diversity are widely studied Chen et al. (2005), Zhang et al (2002), Littlewood et al. (2001), Forrest et al. (1997), O'Donnell and Sethu (2004), Deswarte et al. (1998), Hawthorne and Perry (2004). Page restriction prevents us from providing a more detailed literature survey.

Though security by diversity is introduced as a key defense against internet hackers, little research has been carried on formalizing and solving the problem on how to design an integrated information system to achieve effective diversity.

PROBLEM FORMULATION

In this section, we will describe our formulation to the system design problem.

System Architecture

A web application can usually be described in three layers: *presentation layer, application (or business logic) layer,* and *database layer* (see, e.g., Umar, 2003) (Figure 1).

[1] System survivability is the ability of a system to continue to fulfill its mission in the presence of attacks, accidents, or failures (Avizienis, A. 2000).

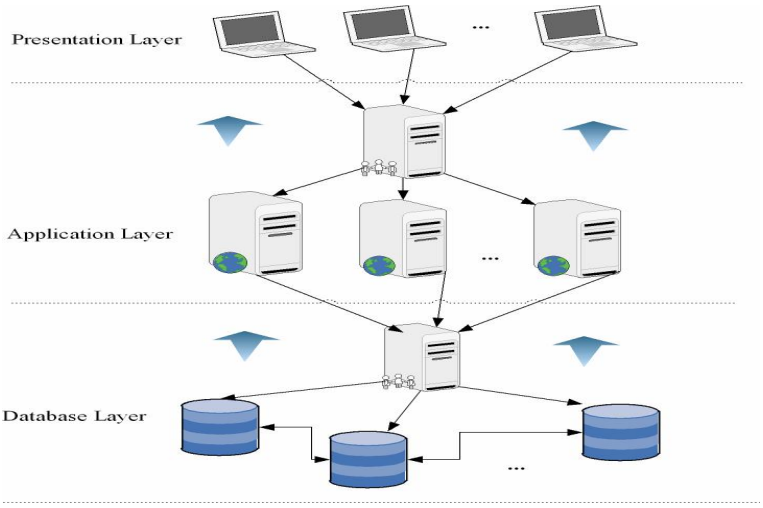

Figure 1 System Architecture

A challenge for distributed databases is how to keep the distributed data consistent. In the following section we will discuss some possible data consistency protocols. A consistency protocol provides an implementation of a consistency model. There are many consistency models that have been developed. In Figure 2 we show the read-one, write-all (ROWA) consistency model. In ROWA, a process reads a replica by finding any copy and using it, while write updates need to acquire all copies. We demonstrate our framework using the ROWA consistency model. A more detailed discussion of consistency models can be found in the works of (Tanenbaum and van Steen, 2002).

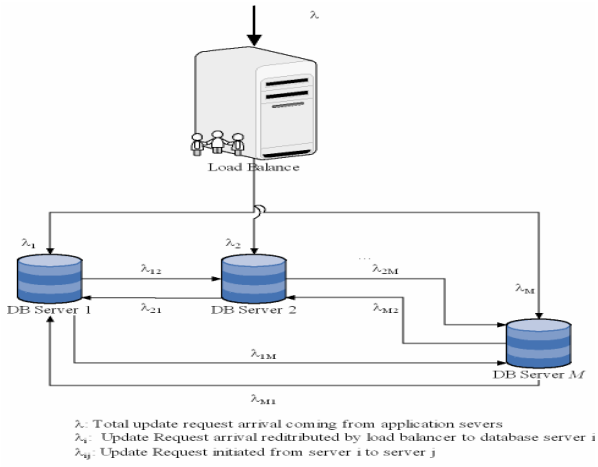

Figure 2 Read-One, Write-All

System Metrics

We consider three types of metrics to measure an integrated system, which include:

- System Survivability. Diversity is employed to introduce heterogeneity and limit correlated risks, thus increasing the survivability of the system. The system survivability can be measured by the proportion of time that system fails due to attacks.
- Average Response Time (ART). Research also showed that increases in delay clearly relate to decreases in performance, attitudes, and behavioral intentions of the clients (Galletta et al., 2004).
- System Total Cost. We want to minimize our expense to achieve the effective diversity.

These design objectives are conflicting with each other, which makes the design problem even hard.

Decision Problem

Assuming that there are a set of individual diversified application/database solutions, we consider a design problem, in which a decision maker has to choose a combination of these application/database solutions for an integrated system. The design objectives are to maximize system survivability, and minimize ART and total system cost. Because of the diversification, these application/database solutions differ from each other in many aspects. We evaluate these individual solutions using the set of criteria shown in Table 1. The evaluation for these criteria can be obtained based on historical data or from experts.

Criterion	Meaning
Maintenance cost per hour	The cost of recruiting service staff to repair the compromised server
Development cost	The cost of developing the individual solution
Average request processing time	The average time for the server to fulfil a request
Survival probability	The probability that the solution will survive when it confronts an attack

Table 1 Evaluation Criteria for Individual Solutions

SOLVING FRAMWORK

Given a number of diversified individual solutions, it is infeasible to evaluate all these solutions one by one because of combinatorially explored decision space. The design objectives are also conflicting, and it is difficult to specify accurate mathematical models for those objectives due to system complexity and request and attack randomness. Our design framework employs a combination of configuration evaluation through controlled system simulations and a neural network based feedback learning mechanism in the exploration of the design space. The simulation model is used to evaluate the effectiveness of the solutions, and a neural network is trained to approximate the behavior of the system with system feedback. Guided by the trained neural network, a multiple-objective evolutionary algorithm (MOEA) is proposed to explore the solution space and identify potential good solutions. The MOEA search incorporates the decision maker's preference as the search direction. Potential good solutions are then evaluated through simulation. The newly obtained simulation results can be used to refine the neural network. The exploration process stops until the result convergences or a satisfied solution is found. The detailed steps are described as follows.

Procedure random_solution_sampling
input:
n: Sample Size
|A|: the number of application solutions
|D|: the number of database solutions
output:
x(|A|, n): *x(a,j)*=1 indicates the application solutions a is chosen in sample j.
x(a,j)=0 indicates the application solutions a is not chosen in sample j.
y(|D|, n): *x(d,j)*=1 indicates the database solutions d is chosen in sample j.
x(d,j)=0 indicates the database solutions d is not chosen in sample j.
begin
for *j=1* to *n*
for each $a \in A$
randomly generate a value *r* between 0 and 1;
if r<0.5 then x(a,j)=1;
else x(a,j)=0 end;
end;
for each $d \in D$
randomly generate a value *r* between 0 and 1;
if r<0.5 then y(d,j)=1;
else y(d,j)=0 end;
end;
end;
end

Table 2 A Procedure for Random Sampling

Step 1: Solution Sampling

We first sample a set of random solutions from the design space. Table 2 shows an implementation of the random sampling. For a more detailed discussion on sampling see Keller, 2005..

Step 2 Prior Simulation

The simulation model is used to evaluate design solutions.

Step 3 Artificial Neural Network Training

In this step, a neural network is trained to learn the effect of different design configurations with the results from the prior simulation. We use a feed-forward fully-connected neural network. The input of the network is a binary vector which indicates a system solution. The output of the neural network includes the measures of system survivability, ART, and total cost. We use the cross-validation techniques (Russell and Norvig, 2003) to decide on the number of hidden layers and their sizes. The transfer function in the hidden layers is tan-sigmoid, and the output layer transfer function is linear. The Levenberg-Marquardt algorithm (Hagan and Menhaj, 1994) is used for network training. In general, on such problems, it has been shown that the Levenberg-Marquardt algorithm provides the fastest convergence (Demuth and Beale, 2002).

Step 4: Multiple Objective Evolutionary Algorithm (MOEA) Search

Evolutionary algorithm is particularly desirable for solving multiobjective optimization problems because they deal simultaneously with a set of possible solutions (population) which allows us to find several members of the Pareto optimal set in a single run of the algorithm, instead of having to perform a series of separate runs as in the case of the traditional mathematical programming techniques (Coello Coello and Mariano Romero, 2005;,Abraham and Jain, 2004). In our proposed evolutionary algorithm, we incorporate the decision maker's preferences in the proposed MOEA. The decision make specifies his aspiration level as the direction of MOEA search. Since we are only interested in obtaining a solution that is "closest" to the decision maker's aspiration level instead of exploring the whole set of Pareto frontier, we do not consider the selection to maximize the diversity of the obtained solution in our MOEA. The selection probability of the solutions is determined based on Rank Order Centroid method (Barron and Barret, 1996). We use two binary vectors X, Y (two chromosomes) to represent an integrated solution: X for the applications solution and Y for the database solution. The framework of the MOEA is shown in Table 3.

Fitness Function: The fitness function we use in the MOEA is a scalarizing function introduced by Wierzbicki (1980). It overcomes the weakness of linear functions which is that they cannot generate non-convex portions[2] of the Pareto frontier regardless of the weight combination used (Coello Coello, 2004). By the scalarizing function, the most highly ranked alternative(s) is always nondominated[3] The scalarizing function may be thought of as a variation of goal programming(Charnes and Cooper, 1961). Goal programming will often but not always find a 'nearest' nondominated solution if the aspiration level is not feasible, but it will find a solution exactly satisfying the aspiration level otherwise. The scalarizing function overcomes this deficiency and provides a nondominated solution corresponding to every aspiration level. We now describe the scalarizing function.

Given an aspiration level, the score γ_i for solution *i* using the scalarizing function is determined as

$$\gamma_i = \min_{j=1,2,3} w_j \delta_{ij} + \varepsilon \sum_{j=1}^{3} w_j \delta_{ij} \quad (1)$$

where ε is a sufficiently small positive scalar, and $\delta_{ij} = \frac{a_{ij} - \alpha_j}{I_j - N_j}$. a_{ij} is the performance of solution i on the objective *j*, *j=1,2,3*. I_j is the best performance on objective *j*, and N_j is the worst performance on objective *j*. $(I_j - N_j)$ represents the range of performance on objective j, which serve as a normalization term. α_j is the decision maker's aspiration level that

[2] A solution is convex dominated if it is dominated by a convex combination of other solutions.

[3] One alternative dominates another if the first alternative is at least as good as the second in every criterion and strictly better in at least one of them. Alternatives that are not dominated by any other solutions are called nondominated solutions

he want to achieve on objective *j*. w_j is the decision maker's weight on objective *j*. The solutions are then ranked based on their score γ_i in descending order.

```
Procedure MOEA_Guided_By_ANN
input:
      n: The Size of Population
      G: The number of generation evolved.
      P: initial population.
      T: performance of initial population
      |A|: the number of database solutions
      |D|: the number of database solutions
output:
      X(|A|, n): x(a,j)=1 indicates the application solutions a is chosen in sample j.
                 x(a,j)=0 indicates the application solutions a is not chosen in sample j.
      X(|D|, n): x(d,j)=1 indicates the database solutions d is chosen in sample j.
                 x(d,j)=0 indicates the database solutions d is not chosen in sample j.
begin
      train the neural network with the initial population P and its performance T.
      specify the decision maker's aspiration level for each objective.
      rank the alternatives using the scarlizing function.
      assign selection probability using rank order centroid.
      let μ =| P |.
      for g=1 to G
            for j=1 to λ
                  select two parent from P based on their selection probability.
                  % crossover
                  single point crossover for chromosome X.
                  single point crossover for chromosome Y.
                  % mutation
                  mutate on chromosome X.
                  mutate on chromosome Y.
            end;
            % Prediction
            predict the performance of the solutions using trained ANN.
            % Ranking
            rank the alternatives (parents and sons) using scaralizing function.
            % (μ + λ) -Selection
            select top μ solutions from (μ + λ) solutions. % Maintain A Elite List
            assign selection probabilities using rank order centroid.
      end;
end;
```

Table 3 The Frame of Multiobjective Evolutionary Algorithm Guided by ANN

Selection: We use a (μ + λ)-selection strategy, in which parents (with a number of μ) compete with their children (with a number of λ), and those which are at top μ of the ranking list are selected for the next generations.

Selection Probability Assignment: We use the rank order centroid method to determine the selection probability of the solutions. Barron and Barret(1996) suggest that rank order centroid method most accurately ranks solutions compared with other approximate weights such as rank sum weights and rank reciprocal weights. Through simulation, they evaluate the superiority of rank order centroid weights in terms of the degree of identification of the best solutions and value losses with respect to the various combinations of the number of solutions, the number of attributes, and four different distributions from which attribute values are generated. Our selection probability p_k for the solution with a rank *k* is determined as equation (2):

$$p_k = \frac{1}{\lambda}\sum_{j=k}^{\lambda}\frac{1}{j}\ ,\ k = 1, 2, ..., \lambda \tag{2}$$

Step 5: Posterior Simulation

The potential good solutions identified by the MOEA are then evaluated through the simulation model. The newly obtained simulation results can be used to refine the neural network. The re-trained neural network can more precisely predict the behavior of the system.

Step 6: Iterations/Stop Condition

The decision maker may stop the searching process when a satisfied solution is found, or when the result converges. We may apply different convergence criteria, one of which is the maximum absolute difference as defined by $\max_j |\frac{a_j^{'} - a_j}{I_j - N_j}| \le \varepsilon$, *j=1,2,3,* where $a_j^{'}$ is the performance in objective *j* of the best solution in current iteration, a_j is the performance in objective *j* of the best solution in last iteration, and ε is a sufficiently small positive value.

A CASE STUDY

In this section, we use a case to demonstrate and validate the proposed framework. We have 20 application solutions whose performance follows the schemes in Table 4, and 10 database solutions whose performance follows the schemes in Table 5. As we can see, there are 2^{30} (around a billion) different possible combinations as system solutions. It is infeasible to evaluate these solutions one by one.

Criterion	The Distribution That The Values Are Drawn From
Maintenance cost per hour	Normal Distribution (mean=$25, standard deviation=$5)
Development cost	Normal Distribution (mean=$8000, standard deviation=$100)
Average request processing time	Normal Distribution (mean=1s (second)., standard deviation=0.3s.)
Survival probability	Normal Distribution (mean=50%, standard deviation=20%)

Table 4 The schema to generate application server

Criterion	The Distribution That The Values Are Drawn From
Maintenance cost per hour	Normal Distribution (mean=$30, standard deviation=$5)
Development cost	Normal Distribution (mean=$10,000, standard deviation=$200)
Average request processing time	Normal Distribution (mean=0.5 s., standard deviation=0.1s.)
Survival probability	Normal Distribution (mean=60%., standard deviation=20%.)

Table 5 The schema to generate application server

The system architecture of our simulation model follows three-layer architecture as we have discussed. The dispatch rule of load balancers follows round robin. We used ROWA as the data consistency protocol. The data are fully replicated across the databases. All requests are assumed to be data operation requests. The server processing time for a request is exponentially distributed. The request arrival and the attack arrival are assumed to be Poison processes. The time between request arrivals is exponentially distributed with a mean of 60s (second), and the time between attack arrivals is exponentially distributed with a mean of 120s. When an attack arrives, it randomly picks a server. And then we determine whether the picked server fails or not based on its survival probability. If the server fails, the maintenance duration of the server is exponentially distributed with a mean of 2 hours. The maintenance cost is the product of the maintenance cost per hour and the maintenance duration. We use the steady state simulation (Law and Kelton, 1991). The first 50,000s served as a warm-up period, after which the system is stabilized. Each simulation has a period of 5,000 requests (around 300,000s). The average

response time (ART) of requests is used to measure Quality of Service.The proportion of the system failure time is recorded to measure the system survivability. The total cost includes the system development cost and the maintenance cost.

Initially, a set of 100 solutions are randomly sampled from the design space and evaluated through the simulation model. The simulate results are then used to train the neural network. Guided by the trained neural network we explore the decision space using the MOEA. The initial population for the MOEA is the set of random solutions. We have the population size μ equal to 100, and the offspring size λ equal to 120 for the MOEA. The MOEA evolves six generations. The decision maker's aspiration level for each objective is set at the best value that can be achieved, which is 1s for average response time, 0% for the proportion of system failure time, and $80,000 for the total cost. The worst acceptable value for each objective is 10sec for average response time, 10% for the proportion of system failure time, and $120,000 for the total cost. Each objective is equally weighted.

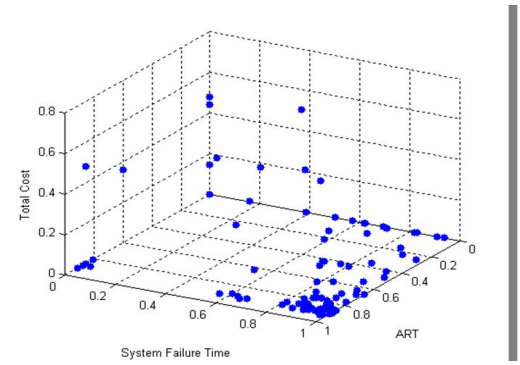

Figure 3 Performance Of Initial Samples

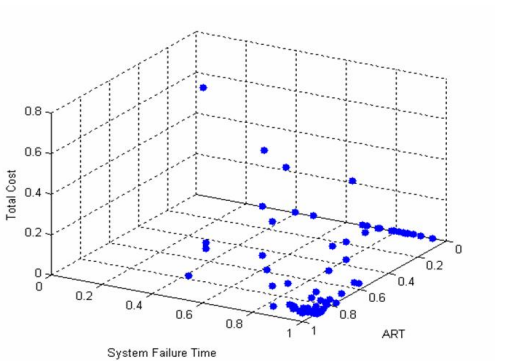

Figure 4 Performance Of Solutions Obtained After The 1st Iteration

Rank With γ_i In Descending Order	Solutions By Random Sampling		Solutions After 1st Iteration	
	Performance	γ_i	Performance	γ_i
The Best Solution	1.7sec, 8.2%, $98756	0.18	3.2sec, 2.6%, $106537	0.34
Average of Top 5% Solutions	1.7sec, 1.8%, $154094	0.04	4.7sec, 3.1%, $113862	0.15

Table 6 A Summary of Comparison

We normalize the performance of solutions with equation (3):

$$a'_{ij} = \frac{a_{ij} - N_j}{I_j - N_j} \tag{3}$$

where a_{ij} is the performance of solution *i* on the objective *j*, *j=1,2,3*. I_j is the best performance on objective *j*, and N_j is the worst performance on *j*. The normalized performances of the initial samples are scattered in Figure 3. The results are spread over the performance space. With the initial samples, the best score of γ_i (as defined in equation (1)) is 0.18. The performance of the best solution is (1.7sec, 8.2%, $98756). The average performance of top 5% design solutions based on γ_i is (1.7sec, 1.8%, $154094), and the average score γ_i of these 5% solutions is 0.04.

In the 1st iteration, we use the simulation results from random sampling to train the neural network. With the guide of the trained neural network, we obtained a new set of solutions using the MOEA. The normalized performance of these solutions is scattered in Figure 4. We can see that these solutions are moving toward decision maker's aspiration level, which corresponds to the point (1,1,1). The best score of γ_i we have now is 0.34. The performance of the best solution is (3.2sec, 2.6%, $106537). The average performance of top 5% design solutions based on γ_i is (4.7sec, 3.1%, $113862), and the average score γ_i of these 5% solutions is 0.15. These comparisons are summarized in Table 6.

CONCLUSIONS

In the study we introduce a systematic framework to help a decision maker to achieve the effective diversity design. Because of the diversification, these component solutions differ from each other in many aspects, such as maintenance and development cost, system survivability, as well as Quality of Service. We employed a combination of configuration evaluation through controlled system simulations and a neural network based feedback learning mechanism in the exploration of the design space. With the guidance of the trained neural network, a multiobjective evolutionary algorithm is developed to explore the decision space and identify the potential good solutions by incorporating the decision maker's preference. The design framework is validated through a case study, and the result is promising.

The approach developed in this paper can be an effective decision support tool for a system designer to systematically explore design options and select an appropriate design configuration that best meets the design objectives.

REFERENCES

1. Abraham, A., and Jain, L. (2004) Evolutionary Mulltipobjective Optimization, In A. Abraham, L. Jain and R. Goldberg (eds.) *Evolutionary Multiobjective Optimization*, Springer.
2. April, J., Glover, F., Kelly, J.P., and Laguna, M. (2003) Practical Introducation To Simulation Optimization, In *Proceedings of the The 2003 Winter Simulation Conference*.
3. Avizienis, A. (2000) Design Diversity and the Immune System Paradigm: Cornerstones for Information System Survivability, ISW-2000 Position Papers, available at http://www.cert.org/research/isw/isw2000/papers/17.pdf.
4. Bain, C., Faatz, D., Fayad, A., and Williams, D. (2001) Diversity as a Defense Strategy in Information Systems, The MITRE Corporation Techniqchal Paper, Available at http://www.mitre.org/work/tech_papers/tech_papers _01/ bain_diversity/bain_diversity.pdf.
5. Barron, F.H., and Barret, B.E.(1996) Decision Quality Using Ranked Attribute Weights, *Management Science*, 42, 1515-1523.
6. Benjamin, R., Gladman, B., And Randell, B. (1998) Protecting IT Systems from Cyber Crime, *The Computer Journal*, 41, 7.
7. Bhattacharjee, S., Zhang, H., Ramesh, R., and Andrews, D.H. (Forthcoming) A Decomposition and Guided Simulation Methodology for Large-Scale System Design: A Study in QoS-capable Intranets with Fixed and Mobile Components, *INFORMS Journal Of Computing*).
8. Budhijara, N., Marzulio, K., Schneider, F., and Toueg, S. (1993) The Primary-Backup Approach, In S. Mullender (ed.) *Distributed Systems,2nd ed.*, Addison-Wesley, Workingham, pp. 199-216.
9. Charnes, A., and Cooper, W.W. (1961) Management Models and Industrial Applications of Linear Programming, John Wiley & Sons, New York, NY.
10. Chen, P.-Y., Kataria, G., and Krishnan, R.(2005) Software Diversity for Information Security, In *Proceedings of the Fourth Workshop on the Economics of Information Security (WEIS05)*, Harvard University, Cambridge, MA.
11. Coello Coello, C.A. (2004) Recent Trends in Evolutionary Multiobjective Optimization, In A. Abraham, L. Jain and R. Goldberg (eds.) *Evolutionary Multiobjective Optimization*, , Springer.
12. Coello Coello, C.A., and Mariano Romero, C.E. (2005) Evolutionary Algorithms And Multiple Objective Optimization, In M. Ehrgott and X. Gandibleux (eds.) *Multiple Criteria Optimization: State of the Art Annotated Bibliographic Surveys*, Kluwer Academic Publishers, Boston/Dordrecht/London.
13. Demuth, H., and Beale, M. (2002) Neural Network Toolbox for Use with Matlab, User's Guide, The MathWorks, Inc.
14. Deswarte, Y., Kanoun, K., and Laprie, J.C. (1998) Diversity against accidental and deliberate faults, In *Proceedings of the Proc. of Computer Security, Dependability and Assurance: From Needs to Solutions.*
15. Ernst & Young (2004) Global Information Security Survey 2004, Ernst & Young LLP, Availability at http://www.ey.com/global/download.nsf/International/2004_Global_Information_Security_Survey/$file/2004_Global_Information_Security_Survey_2004.pdf .
16. Forrest, S., Somayaji, A., and Ackley, D. (1997) Building diverse computer systems, In *Proceedings of the Proceedings of the Sixth Workshop on Hot Topics in Operating Systems*, Los Alamitos, CA.
17. Fu, M.C. (2002) Optimization for Simulation: Theory vs. Practice, *INFORMS Journal on Computing* 14, 3, 192-215.
18. Fu, M.C., Andradottir, S., Carson, J.S., Glover, F., Harrell, C.R., Ho, Y.C., Kelly, J.P., and Robinson, S.M. (2000) Integrating Optimization and Simulation: Research and Practice, In *Proceedings of the The 2000 Winter Simulation Conference*, 610-660.

19. Galletta, D.F., Henry, R., McCoy, S., and Polak, P. (2004) Web Site Delays: How Tolerant are Users?, *Journal of the Association for Information Systems* 5, 1 1-28.
20. Geer, D., Pfleeger, C.P., Schneier, B., Quarterman, J.S., Metzger, P., Bace, B., and Gutmann, P.(2003) Cyberinsecurity: The Cost Of Monopoly, Computer & Communications Industry Association. Available at http://www.securityforest.com /wiki/index.php/Cyber_Insecurity:_The_Cost_of_Monopoly.
21. Gifford, D. (1979) Weighted Voting for Replicated Data, In *Proceedings of Seventh Symposium of Operation System Principles*, 150-162.
22. Glover, F., Kelly, J.P., and Laguna, M. (1999) New Advances for Wedding Optimization and Simulation, In *Proceedings of the The 1999 Winter Simulation Conference*, 255-260.
23. Hagan, M.T., and Menhaj, M.(1994) Training Feedforward Networks with the Marquardt Algorithm, *IEEE Transactions on Neural Networks*, 5, 6, pp. 989-993.
24. Hawthorne, M.J., and Perry, D.E. (2004) Applying Design Diversity to Aspects of System Architectures and Deployment Configurations to Enhance System Dependability,In *Proceedings of the International Conference on Dependable Systems and Networks*.
25. Jalote, P. (1994) *Fault Tolerance in Distributed Systems*, Prentice Hall, Englewood, NJ.
26. Keller, G.(2005) *Statistics for Management and Economics*, Thomson Brooks/Cole.
27. Law, A.M., and Kelton, W.D. (1991) *Simulation Modeling and Analysis*, McGraw-Hill Inc, New York.
28. Littlewood, B., Popov, P., and Strigini, L.(2001) Modelling software design diversity - a review, *ACM Computing Surveys*, 33, 2, 177-208.
29. Littlewood, B., and Strigini, L.(2004) Redundancy And Diversity In Security, In *Proceedings of the ESORICS 2004, 9th European Symposium on Research in Computer Security*, Sophia Antipolis, France.
30. mi2g, (2004) The World's safest Operating System, Available at *http://www.mi2g.net/cgi/mi2g/frameset.php?pageid= http%3A//www.mi2g.net/cgi/mi2g/press/190204_2.php*.
31. O'Donnell, A.J., and Sethu, H. (2004) On Achieving Software Diversity for Improved Network Security using Distributed Coloring Algorithms, In *Proceedings of the CCS'04*, Washington, DC, USA.
32. Russell, S., and Norvig, P. (2003) *Artificial Intelligence: A Modern Approach*, Prentice Hall, Upper Saddle River, N. J..
33. Shantikumar, J.G., and Sargent, R.G. (1983) A Unifying View of Hybrid Simulation/Analytic Models and Modeling, *Operations Research*, 31 1030-1052.
34. Shao, B.N., and Rao, H.R. (2001) A Comparative Analysis of Information Acquisition Mechanisms for Discrete Resource Allocation, *IEEE Transactions on Systems, Man and Cybernetics (Part A)* 31, 3, 199-209.
35. Shim, J.P., Warkentin, M., Courtney, J.F., Power, D.J., Sharda, R., and Carlsson, C. (2002) Past, present, and future of decision support technology, *Decision Support Systems*, 33, 2, 111-126.
36. Simon, H.A. (1996) *The Sciences of the Artificial*, MIT Press, Cambridge, Mass.
37. Stamp, M. (2004) Risks of Monoculture, *Communications Of The ACM*, 47, 3.
38. Tanenbaum, A.S., and van Steen, M.(2002) *Distributed Systems Principles And Paradigms*, Prentice Hall, Upper Saddle River, New Jersey.
39. Thomas, R.(1979) A Majority Consensus Approach to Concurrency Control for Multiple Copy Databases, *ACM transactions on Database Systems*, 4, 2, 180-209.
40. Umar, A. (2003) E-Business and Distributed Systems Handbook: Architecture Module, Nge Solutions.
41. Wang, J., Sharman, R., and Ramesh, R.(2005) "A Design Framework for Shared Content Management in Replicated Web Systems using Problem Decomposition, Controlled Simulation and Feedback Learning," School of Management Working Paper No. 810, State University of New York-Buffalo, 2005.
42. Wierzbicki, A.P. (1980) The use of reference objective in Multiobjective Optimization, In G. Fandel and T. Gal (eds.) *Multiple Criteria Decision Making, Theory and Application*, Springer, Berlin, 468-486.
43. Zhang, Y., Vin, H., and Alvisi, L.(2002) Heterogeneous Networking: A New Survivability Paradigm, In *Proceedings of the NSPW'01*, Cloudcroft, Ncw Mexico, USA.