Self-Healing Applications by Using Case Base Reasoning Algorithm

Nadir K. Salih, Doha Idries

Computer Program, Engineering College, University of Buraimi, Oman

DOI: https://doi.org/10.51584/IJRIAS.2024.908004

Received: 27 July 2024; Accepted: 01 August 2024; Published: 27 August 2024

ABSTRACT

There are many problems facing applications, such as Reliability, security and confidentiality problems, and problems facing the system's behavior from service interruptions or lack of availability at the time specified for the customer, is one of the most problems that affect the documentation of the system. This research provides a mechanism for automating system which is self-Healing using an inference algorithm based on previous cases (CBR). Which means that the algorithm will enable the application to detect and identify the problem and then self-treat it, and it will be able to examine and treat itself without external interference. This mechanism is applied to the mobile bank application as a case study of an application that contains sensitive data. This algorithm was implemented on Eclipse program using the Java language in programming and used the case database to save a set of cases like the occurrence of the problem, and the algorithm works on the conclusion of the best solution to solve the problem from among the set of available solutions.

Keywords: Autonomic computing, Self-Healing, Case Base Reasoning

INTRODUCTION

Smartphone with open-source operating systems is becoming increasingly popular now days. Android is one of the most popular open-source operating systems for mobile platforms. Android offers a set of powers to protect phone applications. With the basic functionality of System utilities, middleware in form of Virtual Machine (VM) [1]. The materials and applications of self-healing are no longer an illusion, and we are not far from the days when human-made materials can restore their structural integrity in case of failure. Self-healing is a part of the autonomic system and can be defined as the ability of a material to heal, recover or repair from the damages automatically and autonomously without any external intervention, many common terms such as self-healing, autonomic healing, and Self-repair is used to determine such property in materials [2]. Autonomic systems/applications exhibit eight defining characteristics [7]: Self Awareness: An autonomic application/system "knows itself" and is aware of its state and its behaviors. Self-Healing: An autonomic application or system should be able to detect and recover from potential problems and continue to function smoothly. Self Configuring: An autonomic application system should be able configure and reconfigure itself under varying and unpredictable conditions. Self-Optimizing: An autonomic application or system should be able to detect suboptimal behaviors and optimize itself to improve its execution. Self-Protecting: An autonomic application/system should be capable of detecting and protecting its resources from both internal and external attack and maintaining overall system security and integrity. Context Aware: An autonomic application system should be aware of it execution environment and be able to react to changes in the environment. Open: An autonomic application/system must function in an heterogeneous world and should be portable across multiple hardware and software architectures. Consequently, it must be built on standard and open protocols and interfaces. Anticipatory: An autonomic application/system should be able to anticipate to the extent possible, its needs and behaviors and those of its context, and be able to manage itself proactively.

Managed Element: This is the smallest functional unit of the application and contains the executable code (program, data structures) (e.g., numerical model of a physical process). It also exports its functional interfaces, its functional and beaver.

Functionality: Is a set of functions that satisfy specific application needs in terms of appropriateness, accuracy, interoperability, security and compliance with standards in the area.

Reliability: Is describes the application ability to work correctly in difficult or special conditions. High tolerance to errors and recoverability are two of the requirements that the application in question must have.

Usability: Is defined by the effort required to use the software by users, Operability is high because the time it produces the desired result is very short.

Efficiency: Software application is by the correlation optimum between the consumption of resources and complexity or difficulty of the problem to solve. The result of using the self-healing algorithm in an application that contributes to:-

1. Increased self-healing in applications.
2. a similarity which matching algorithm will improves the system performance.
3. As a result of using the CBR algorithm, the client has been provided with no service or process interruptions.
4. As a result of using CBR algorithm, it achieved reliability between the server and the client reducing the manual dispute process.
5. Solve the problems based on the conclusion of the solution stored in the case base and use the optimal solution

LITERATURE REVIEW

Ghosh, S.K. et al [2] Self-healing can be defined as the ability of a material to heal recover or repair damages automatically and autonomously, that is, without any external intervention Incorporation of self-healing properties in manmade materials very often cannot perform the self-healing action without an external trigger.

Table 1 Four aspects of self-management

Concept	**Current computing**	**Autonomic computing**
Self-configuration	Corporate data centers have multiple vendors and platforms. Installing, configuring, and integrating systems is time consuming and error prone.	Automated configuration of components and systems follows high-level policies. Rest of system adjusts automatically and seamlessly.
Self-healing	Problem determination in large, complex systems can take a team of programmer’s weeks.	System automatically detects, diagnoses, and repairs localized software and hardware problems.
Self-protection	Detection of and recovery from and cascading failures is manual.	System automatically defends against malicious attacks or cascading failures. It uses early warning to anticipate and prevent system wide failures.
Self-optimization	Systems have hundreds of manually set, nonlinear tuning parameters, and their number increases with each release.	Components and systems continually seek opportunities to improve their own performance and efficiency.

Van, I., Boja, C et al. [3] The reliability and security of software application is one of the most important characteristics of software quality because it describes the ability of the software to work without any failure and protect the user data. Mobile applications are compatible with desktop applications in terms of rich interfaces and functions and have become one of the most widely use applications. Mobile applications should provide special applications to avoid failure and maintain a high level of reliability and security because mobile operating systems provide limited services or access to administrative or regulatory tools that allow a user with an IT background to access temporary or permanent data. A high level of software reliability is directly affected by a low level of failure. Manish Parashar et al [4] Autonomic applications and systems are composed from autonomic elements and can manage their behaviors and their relationships with other systems/ applications in accordance with high-level policies. Azim, M.T. et al [5] Our first application uses dynamic analysis to detect when an application has entered an error and determines the finished part of the application; then executes the recovery process, either by eliminating transient errors and continuing to operate at full capacity or reversing to a safe state followed by closing the finished part, Limited mode while avoiding further breakdowns. They take approach Android platform, The Android platform consists of the Android OS (a Linux kernel customized for smart phones), the Dalvik VM (virtual machine) and apps, written mostly in Java, that are compiled to a byte code format named DEXand execute in the Dalvik VM. Apps can also embed native code (written in C/C++) that is executed directly by the OS. They take architecture Centered around detecting crashes and in response applying seal-off patches. A model is constructed rest, via static analysis on the app byte code); the model includes rollback (resume) points where the app will be driven when recovering from a fault. Next, the app is executed, either manually by users or automatically via systematic exploration tools and its execution logis monitored via dynamic analysis. When a failure is detected, we employ byte code rewriting (code generation and instrumentation). Jiang, M. et al [6] In order for the system to be self-healing, the system must be able to detect the errors that have occurred and how to correct them.Self-healing software systems. The model-based approach is used to classify software failures and determine their arrangements at the model level.

The basis of autonomic computing draws on biological analogies to describe an autonomic computer as a computer that is self-governing, in the same manner that a biological organism is self-governing. Puviani et al [7] is the process by which computer systems shall manage their own operation without human intervention. The growing complexity of modern networked computer systems is currently the biggest limiting factor in their expansion. The increasing heterogeneity of big corporate computer systems, the inclusion of mobile computing devices, and the combination of different networking technologies like WLAN, cellular phone networks, and mobile ad hoc networks make the conventional, manual management very difficult, time-consuming, self-management has been suggested as a solution to increasing complexity in cloud computing. Hellerstein et al [8] A method, computer program product, and data processing system for constructing a self-managing distributed computing system comprised of “autonomic elements” is disclosed. An autonomic element provides a set of services and may provide them to other autonomic elements. Relationships between autonomic elements include the providing and consuming of such services. These relationships are “late bound,” in the sense that they can be made during the operation of the system rather than when parts of the system are implemented or deployed. They are dynamic, in the sense that relationships can begin, end, and change over time. They are negotiated, in the sense that they are arrived at by a process of mutual communication between the elements that establish the relationship. Barel, M.A., et al [9] Analysis module analyzes the automation capabilities of the customer based on survey answers. Automation capabilities of an enterprise include, for example, the ability to be self-configuring, the ability to be self-healing, the ability to be self-optimizing, and the ability to be self-protecting. Across the four automation capabilities, there are several key operational areas where one can assess automation maturity. These operational areas are used as automation assessment categories in accordance with an exemplary embodiment of the present invention. The automation assessment categories may include, for example, problem management, availability management, security management, solution deployment, user administration, and performance and capacity management. Problem management is the act of identifying, isolating, and resolving issues that might negatively impact IT service delivery. Availability management is the act of ensuring that required IT services are available, as needed, to ensure business continuity. Security management is the act of securing critical business resources and data against attacks and authorized access from both external and internal threats. Solution deployment is the act of planning, testing, distributing, installing, and validating the deployment of new IT solutions, including the IT infrastructure elements, in a manner that is the least disruptive to operational services. The ability to roll back to a prior functioning environment if a change is unsuccessful is also necessary. KATHLEEN S. TOOHEY et al [12] Self-healing polymers composed of microencapsulated healing agents exhibit remarkable mechanical performance and regenerative ability1–3 but are limited to autonomic repair of a single damage event in a given location. Self-healing is triggered by crack-induced rupture of the embedded capsules thus, once a localized region is depleted of healing agent further repair is precluded. Re-mendable polymers4,5 can achieve multiple healing cycles, but require external intervention in the form of heat treatment and applied pressure. Christopher J. Hansen et al [13] Micro vascular Substrate Fabrication: Micro vascular interpenetrating networks are fabricated using a three-axis robotic deposition stage (ABL9000, Aerotech Inc.) whose motion is controlled by customized software (RoboCAD version 3.1). Wax and pluronic inks are housed in syringes that are fitted with 330 and 100mm nozzles (EFD Inc.) respectively

METHODOLOGY

Case-based reasoning can mean adapting old solutions to meet new demands; using old cases to explain new situations; using old cases to critique new solutions; or reasoning from precedents to interpret a new situation [15] The classic definition of CBR was coined by Riesbeck and Schank [5]: Case based reasoning solves problems by using or adapting solutions to old problems. Note that this definition tells us what a case-based reasoning does and not “how” it does what it does. Conceptually CBR is commonly described by the CBR cycle (Fig. 1). The case contains two components a description of the problem and the solution. Therefore, it is defined as an example of the problem [18]. The case base contains the experiences and conforms to one of the four sources of knowledge required in a CBR. They are the vocabulary, the case-base, the similarity measure and adaptation containers.

1. The first, the vocabulary, contains the terms which support the others. The case-base comprehends what is in a case and how cases are organized.

2. The similarity measure container contains knowledge to determine the similarity between two cases in the retrieval phase. The solution adaptation container contains knowledge to adapt past solutions to new problems in the reuse stage [19]. Solving a problem by CBR involves obtaining a problem description, measuring the similarity of the current problem to previous problems stored in a case base (or memory) with their known solutions, retrieving one or more similar cases, and attempting to reuse the solution of one of the retrieved cases, The solution proposed by the system is then evaluated [17]. In problem solving CBR, old solutions are used as inspiration for solving new problems. Since new situations rarely match old ones exactly, however, old solutions must be fixed to fit new situations. In this step, called adaptation, the ballpark solution is adapted to fit the new.

Implementation of Self-healing in Bank System

The implementation stage has been done in developing the self-healing in mobile application system and discussing the results. Development involves designing a self-healing of problems related to system behavior and improvement of the entire application. The method used in the development of the application and the database were discussed here. The self -healing in mobile application was developed using Eclipse. The application source code was created by using the Java programming language and the case-base was created using text files, The mobile bank application provides a set of services and processes to end users. In this research, we highlight the interruption of the process or service to the customer and we will use a mechanism that helps solve this problem related to the behavior of the system. The mechanism depends on storing a set of previous solutions and works to deduce the best solution from among the similar solutions. Below figure 1 shows the implemented framework of the self-healing on mobile application (mobile bank FIB) using CBR.

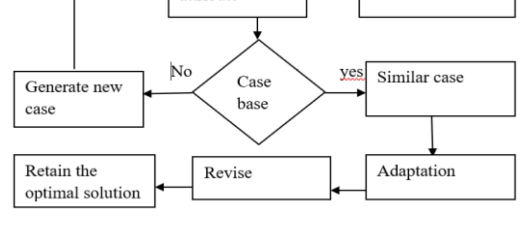


Figure 1 the sequence of interruption operation to buy electricity.

For this application, was developed in Platform Eclipse using the Java programming language. And Net. Benes program, Window 10 is used as on operating system with Intel(R) processor and 3 GB of RAM for develop the application environment. This application used self-healing on mobile application which has been collected from facial Islamic bank for evaluating the CBR algorithm. This dataset is used as the case base for the application. The interface allows the application to interact with the system and others. This system consists of three interfaces, the first interface to measure the weights of tests, the second interface is the new case for test cases, and the third interface is the results of the test. Below figure 2 shows the implemented framework of the self-healing system using CBR.

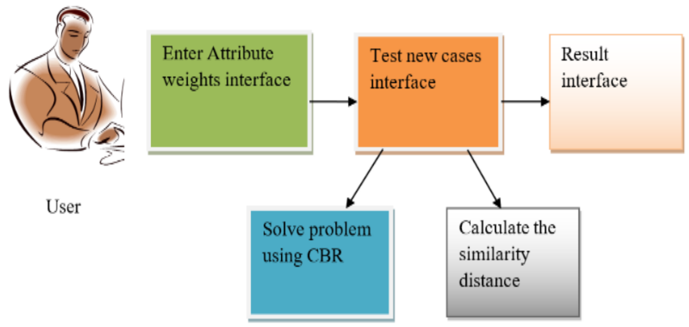
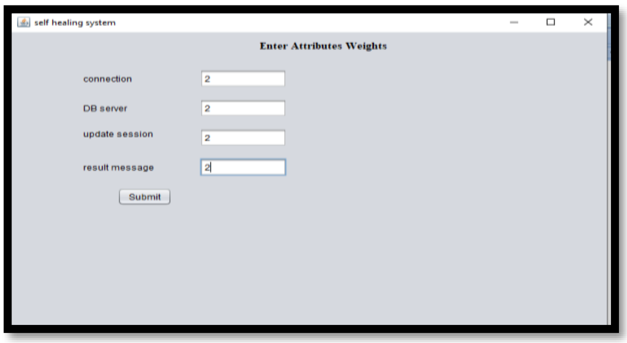


Figure 2 framework of the self-healing system using CBR.

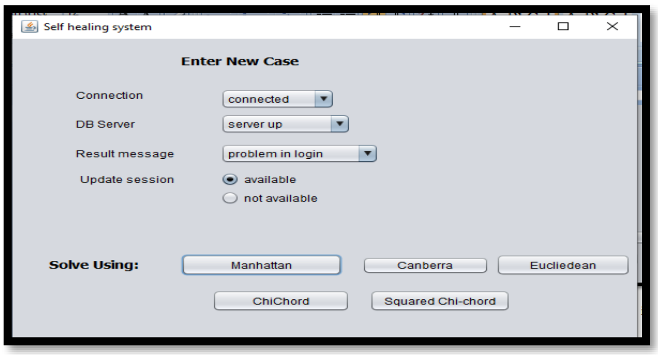
This is the first interface in the model. It consists of four text fields which are used as the input of weights for connection, Database server, update session and result message. And a button to handle this action. Weight is used in Manhattan and Euclidian functions to calculate the distance between new cases and stored cases following figure 3 is the test weight interface in our system.



connection	2
DB server	2
update session	2
result message	2

Figure 3 attribute weights of self-healing system interface

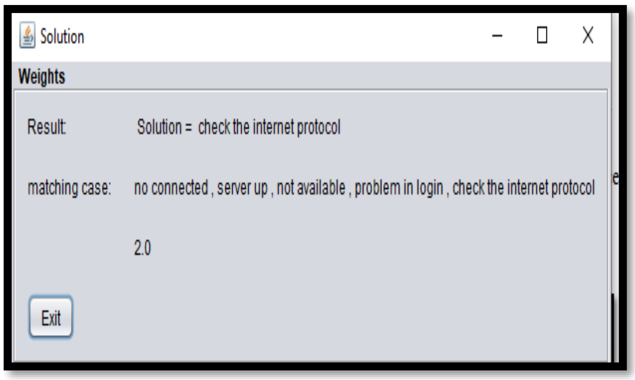
This is the page used to input the new cases of self-healing system information about their Attribute. From the diagnose interface, after clicking the diagnose button, the input will compare with the case-base, retrieve the data from the case-base to do the calculation of diagnosis 'similarity. After the calculation, will display the result and the similarity from the previous case.



Connection	connected
DB Server	server up
Result message	problem in login
Update session	[✓ available] not available

Figure 4 New cases of self-healing system and test interface

This is the result interface where the result of the diagnosis and the new cases will display. In the result interface, it will display the result of diagnosis, best matching case and similarity of the case to the previous case. User can choose either to retain the case or to the end page which is exit interface.



Result: Solution = check the internet protocol

matching case: no connected , server up , not available , problem in login , check the internet protocol

2.0

Figure 5 result of self-healing system interface using Manhattan function

The following tables describe the functions used and the codes that have been implemented, table (2) shows the code reading the value of the value of the new attribute from the user and storing it in the new attribute Value variable, and reading the value of the old Attribute from the case-base and store it in Case Attribute Value:

The first function to calculate the similarity distance between these cases is Manhattan function (1). The variable this distance stores the similarity distance which is calculated by subtract new Attribute Value form Case Attribute Value multiplied by weight of the new case attribute.

$$d= \sum_{i=1}^{n} | xi - yi | \qquad (1)$$

1. The second function is Euclidean :-

$$d= \sqrt{\sum_{i=1}^{n} (xi - yi)2} \qquad (2)$$

2. The Third function is Canberra**.**

$$dij= \Sigma\, k^{n}{}_{=1} \frac{| xik \_ xjk|}{| xik \_ xjk|} \qquad (3)$$

3. The fourth fonction is Squared –Chord

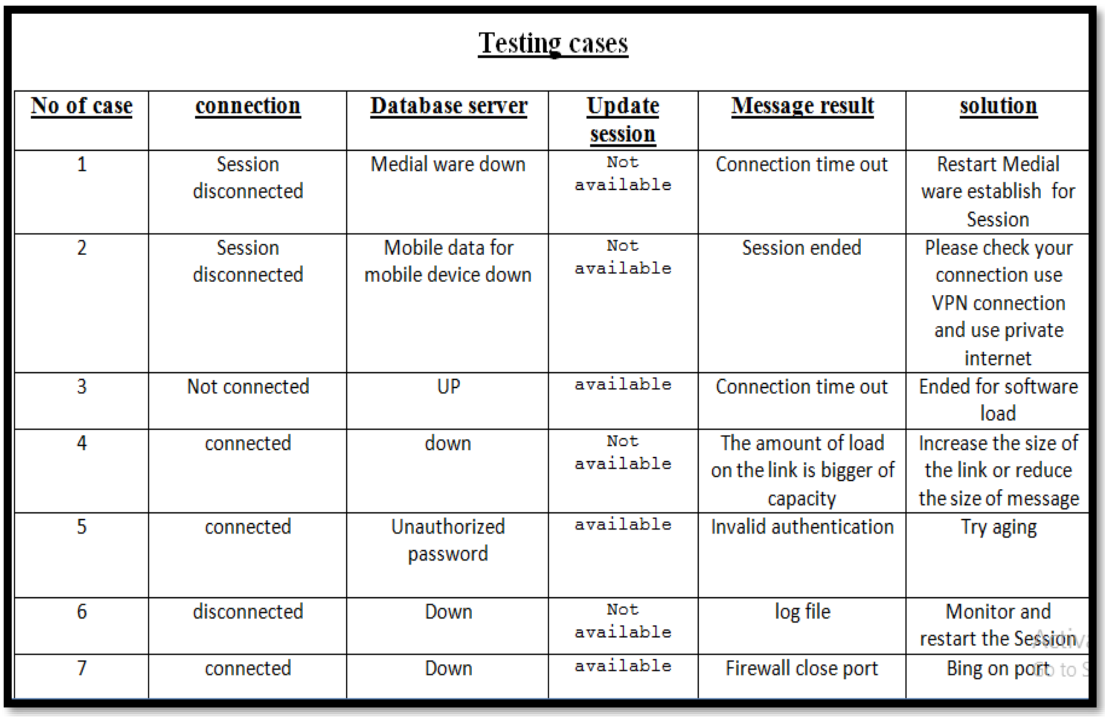
$$d= \sum_{i=1}^{n} (\sqrt{xi} - \sqrt{yi})\ 2 \qquad (4)$$

4. The fifth function is Squared Chi-Squared:-

$$d= \sum_{i=1}^{n} \frac{(xi - yi \ )2}{(\ xi \ + yi \ )} \qquad (5)$$

The Following table (2) show some cases using in calculate similarity distance

Table 2 the new testing cases



Testing cases

No of case	connection	Database server	Update session	Message result	solution
1	Session disconnected	Medial ware down	Not available	Connection time out	Restart Medial ware establish for Session
2	Session disconnected	Mobile data for mobile device down	Not available	Session ended	Please check your connection use VPN connection and use private internet
3	Not connected	UP	available	Connection time out	Ended for software load
4	connected	down	Not available	The amount of load on the link is bigger of capacity	Increase the size of the link or reduce the size of message
5	connected	Unauthorized password	available	Invalid authentication	Try aging
6	disconnected	Down	Not available	log file	Monitor and restart the Session
7	connected	Down	available	Firewall close port	Bing on port

In CBR, there are four components that are important during the prediction which are Retrieve, Reuse, Revise and Retain. In developing the diabetes diagnosis system, retrieve is referring to giving a target problem, retrieve cases from memory that are relevant to solve it. A case consists of a problem, its solution and how the solution is derived. In this application, case retrieval refers to the process of finding the nearest case, which includes the solution for the new case within the case-base. After the nearest case is retrieved, the solution from the previous case is reused to solve the new case.

RESULTS AND DISCUSSION

The four attributes or features used are Connection, Database server, Message result, and update session measurement as a system input that uses analogy in problem solving and inference to match a previous case of the system with the new problem to find a solution. After the similarity is found, the similarity will be calculated using these functions Manhattan, Euclidean, Canberra, Squared Chord and Squared chi-squared functions. The most important features (weight) are determined and it will be used in similarity computation.

Chi-squared test is the best for used to determine whether there is a statistically significant difference between the expected frequencies and the observed frequencies in one or more categories. To measure the accuracy by Equation (6):

$$\text{Accuracy} = \frac{\text{Correct case}}{\text{Total Testing Cases}} *100 \tag{6}$$

CBR algorithm in this app will bring accuracy over 40% to determine system problems, and 61% max. The accuracy of all functions used to calculate the similarity distance is shown in Table (4) below:

Table (4) Similarity Function Accuracy

Function	Accuracy
Manhattan	40%

Euclidian	40%
Canberra	60%
Squared Chord	40%
Squared chi-squared	60%

The following figure (6) is shows the accuracy rate using Euclidean functions and determines number of test cases which have been tested using the similarity and determines the correct result.

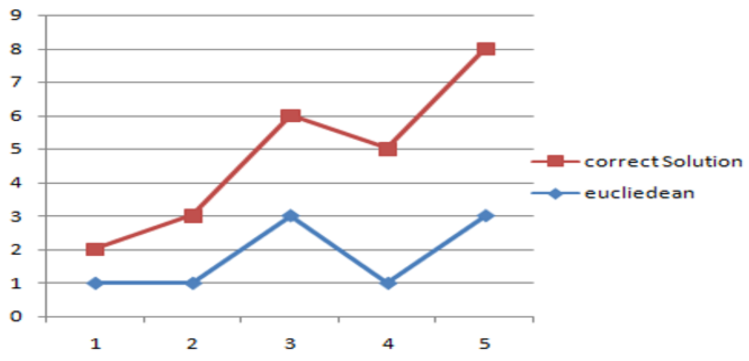


Figure 6 Euclidean function accuracy

The following figure (7) is a chart that shows the accuracy rate using Manhattan functions and determines number of test cases which have been tested using the similarity and determine correct result.

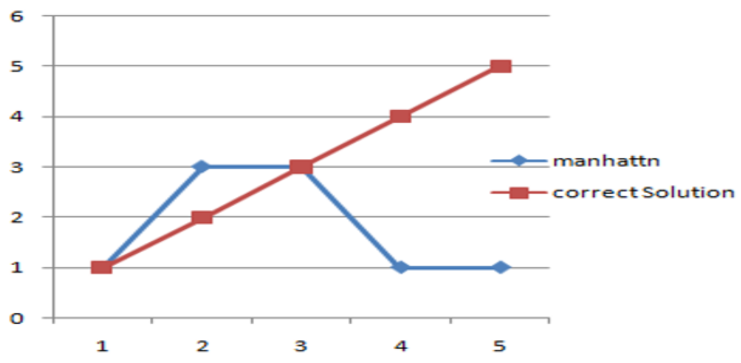


Figure 7 Manhattan function accuracy

The following figure (8) is a chart that shows the accuracy rate using Canberra functions and determines number of test cases which have been tested using the similarity and determines the correct result.

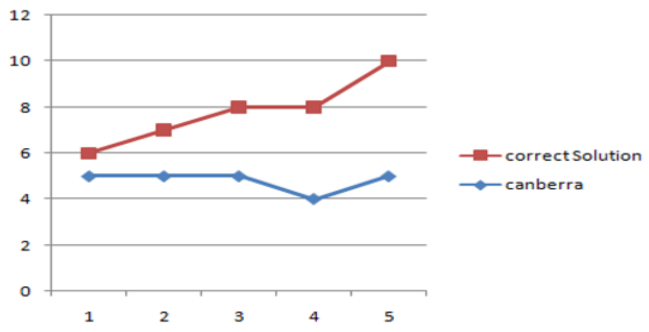


Figure 8 Canberra function accuracy

The following figure (9) is a chart that shows the accuracy rate using chi-chord Functions and determines number of test cases which have been tested using the similarity and determines the correct result.

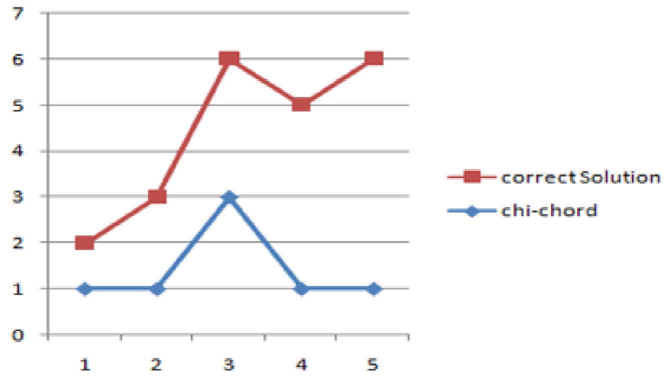


Figure 9 chi-chord function accuracy

The following figure (10) is a chart that shows the accuracy rate using sugared chi-chord Functions and determines number of test cases which have been tested using the similarity and determines the correct result.

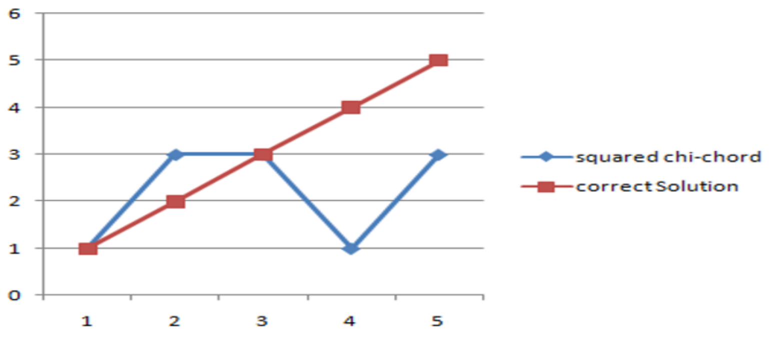


Figure 10 sugared chi-chord function accuracy

The following figure (11) is a chart that shows the accuracy rate using all Functions and determines number of test cases which have been tested using the similarity and determines the correct result.

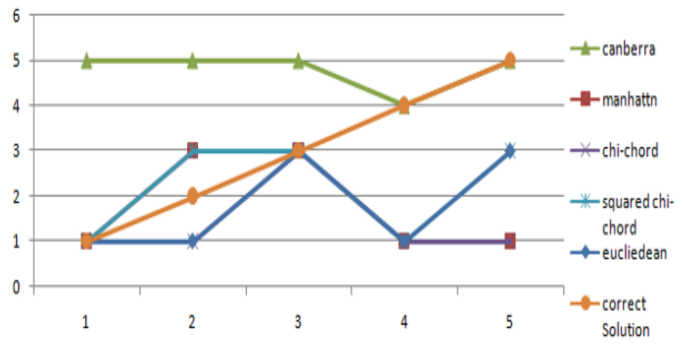


Figure 11 Similarity Functions Accuracy Rate

The application and success of this work will provide benefit from the development of CBR systems in mobile applications in general and the development of the mobile bank system in particular by reducing the dispute that usually occurs when the causes occur, and it becomes possible to correct all error of this through a set of sentences. Self-healing in mobile app system design is presented with the development environment, designing of interface is discussed. The application applies the CBR algorithm.

CONCLUSION AND RECOMMENDATION

The goal of self-healing systems is to provide applicable, highly accessible and reliable systems. It offers properties such as adaptation, transformation, self-preservation, etc. The mechanism used for self-healing and adaptation in this study is case base reasoning, but this study is still in its infancy. To solve the problem here are some suggestions and recommendations that should be made to improve the application. Development of the application by linking the algorithm to the Mobile Bank application. For the next version this application can be implemented inside the mobile.

REFERENCES

1. Powar, S. and Meshram, B.B., 2013. Survey on Android security framework. International Journal of Engineering Research and Applications, 3(2), pp.907-911.
2. Ghosh, S.K. ed., 2009. Self-healing materials: fundamentals, design strategies, and applications (pp. 138-217). Weinheim: Wiley-vch.
3. Kephart, J.O. and Chess, D.M., 2003. The vision of autonomic computing. Computer, 36(1), pp.41-50. 4. Begum, S., Ahmed, M.U. and Funk, P., 2009. Case-based systems in health sciences: a case
4. Parashar, M. and Hariri, S., 2004, September. Autonomic computing: An overview. In International workshop on unconventional programming paradigms (pp. 257-269). Springer, Berlin, Heidelberg.
5. Azim, M.T., Neamtiu, I. and Marvel, L.M., 2014, September. Towards self-healing smartphone software via automated patching. In Proceedings of the 29th ACM/IEEE international conference on Automated software engineering (pp. 623-628).
6. Combemale, B., Broto, L., Crégut, X., Daydé, M. and Hagimont, D., 2008, September. Autonomic management policy specification: From uml to dsml. In International Conference on Model Driven Engineering Languages and Systems (pp. 584-599). Springer, Berlin, Heidelberg.
7. Puviani and Friel (2013). "Self-Management for cloud computing". Proceedings of Science and InformationConference (SAI), 2013: 940–946 – via IEEE Xplore.
8. Hellerstein, J., Kephart, J., Lassettre, E., Pass, N., Safford, D., Tetzlaff, W. and White, S., International Business Machines Corp, 2004. Self-managing computing system. U.S. Patent Application 10/252,247.
9. Barel, M.A., Carter, S., Crosskey, J.P., Ernest, L.M., Evans, D.H., Ford, L.L., Lilies, R.C., Spence, D. and Swett, A.L., International Business Machines Corp, 2011. Method, apparatus, and program for implementing an automation computing evaluation scale to generate recommendations. U.S. Patent 8,019,640.
10. Saha, G.K., 2007. Software-implemented self-Healing system. CLEI Electronic Journal, 10(2).
11. Chan, K.M. and Bishop, J., 2009, May. The design of a self-healing composition cycle for Web services. In 2009 ICSE workshop on software engineering for adaptive and self-managing systems (pp. 20-27). IEEE.
12. Toohey, K.S., Sottos, N.R., Lewis, J.A., Moore, J.S. and White, S.R., 2007. Self-healing materials with microvascular networks. Nature materials, 6(8), pp.581-585.
13. Hansen, C.J., Wu, W., Toohey, K.S., Sottos, N.R., White, S.R. and Lewis, J.A., 2009. Self-healing materials with interpenetrating microvascular networks. Advanced Materials, 21(41), pp.4143-4147.
14. Karray, M.H., Ghedira, C. and Maamar, Z., 2011, March. Towards a self-healing approach to sustain web services reliability. In 2011 IEEE Workshops of International Conference on Advanced Information Networking and Applications (pp. 267-272). IEEE.
15. Watson, I., 1995, January. An introduction to case-based reasoning. In UK Workshop on Case-Based Reasoning (pp. 1-16). Springer, Berlin, Heidelberg.
16. Nadir K. Salih, Abdel-hafiz A. Khoudour, Mawahib S. Adam, Samar M. Hassen "Autonomic Computing Architecture by Self-defined URI" International Journal of Computer Trends and Technology 68.3 (2020):1-6.2020
17. Sheima S. El-hwaij, Nadir K.Salih. Autonomic management by self-optimization for WEINMANN. IEEE, International Conference on Communication, Control, Computing and Electronics Engineering (ICCCCEE), 2017.
18. Nadir K Salih, Tianyi Zang. Need of Autonomic Management SaaS Application. International Journal of Computer Science Issues , 2016.
19. Nadir K Salih, Tianyi Zang. Autonomic Management for Applicability and Performance in SaaS Model. International conference on parallel and distributed processing techniques and applications (PDPTA'14), held in July 21-24 Las Vegas, USA.- 2014.(SCI).
20. Nadir K Salih, Tianyi Zang. Self-management SaaS Application by CBR Algorithm . International conference on parallel and distributed processing techniques and applications (PDPTA'17), held in July

21-24 Las Vegas, USA.- 2017.

21. G.K.Viju, Nadir K.Salih. A secure multicast protocol for ownership rights. International Conference of Computing and Information Technology (ICCIT), 2012, pp 788-793.
22. Nadir Kamal Salih, Self-Diagnosis of Cancer Using Case Base Reasoning Algorithm. International Journal of Research in Engineering, Science and Management. Volume4, Issue6,P: 24-28,2021.
23. Blair, G.S., Coulson, G., Blair, L., Duran-Limon, H., Grace, P., Moreira, R. and Parlavantzas, N., 2002, November. Reflection, self-awareness and self-healing in OpenORB. In Proceedings of the first workshop on Self-healing systems (pp. 9-14).
24. Blair, G.S., Coulson, G., Blair, L., Duran-Limon, H., Grace, P., Moreira, R. and Parlavantzas, N., 2002, November. Reflection, self-awareness and self-healing in OpenORB. In Proceedings of the first workshop on Self-healing systems (pp. 9-14).