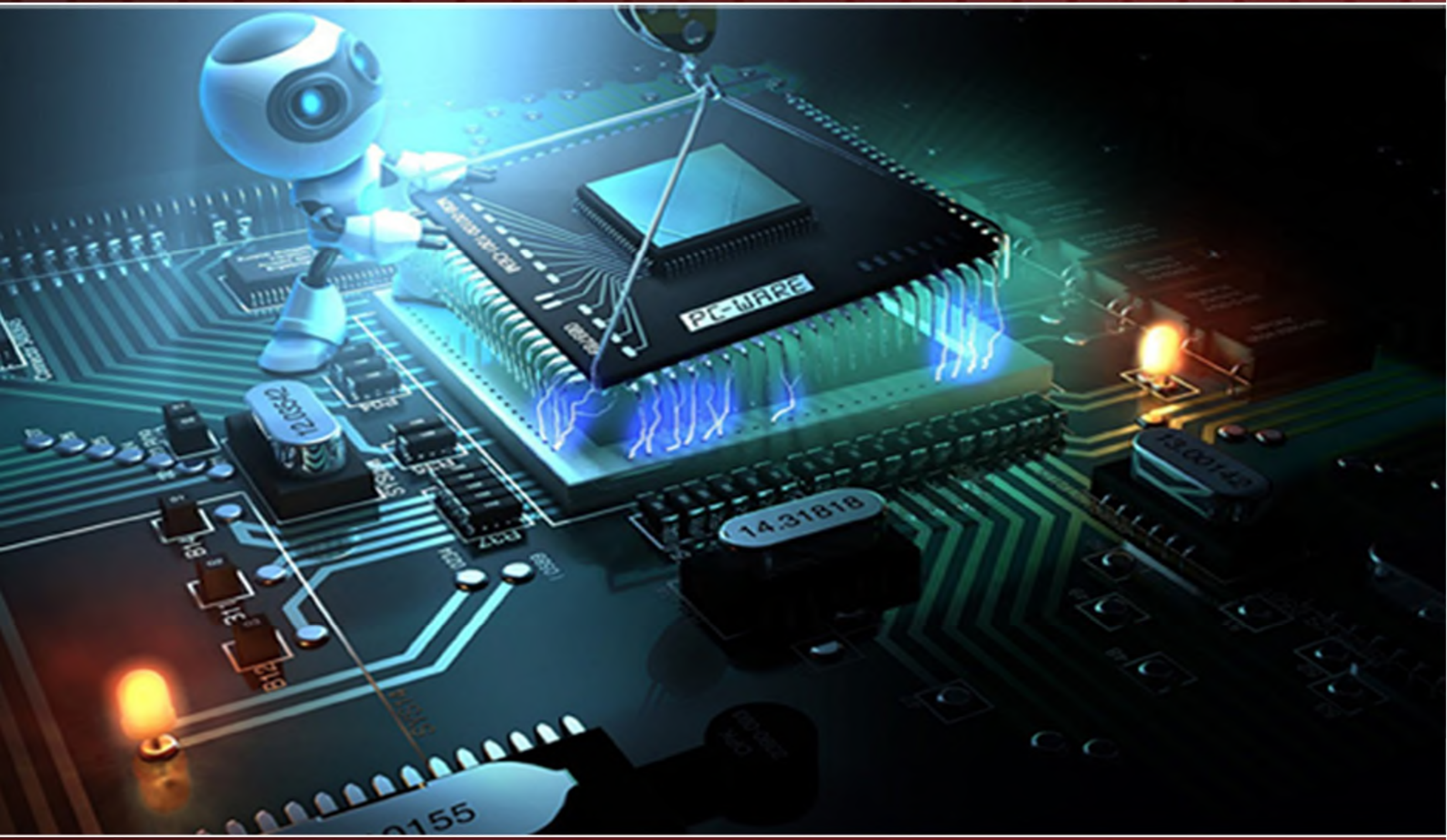
ISSN: 2054-7390

Volume 7 No 4, Aug 2019

Transactions on Machine Learning and Artificial Intelligence



Society for Science and Education
UNITED KINGDOM

TABLE OF CONTENTS

EDITORIAL ADVISORY BOARD

Editor-in-Chief

Professor Er Meng Joo
Nanyang Technological University
Singapore

Members

Professor Djamel Bouchaffra
Grambling State University, Louisiana
United States

Prof Bhavani Thuraisingham
The University of Texas at Dallas
United States

Professor Dong-Hee Shin,
Sungkyunkwan University, Seoul
Republic of Korea

Professor Filippo Neri,
Faculty of Information & Communication Technology
University of Malta
Malta

Prof Mohamed A Zohdy,
Department of Electrical and Computer Engineering
Oakland University
United States

Dr Kyriakos G Vamvoudakis,
Dept of Electrical and Computer Engineering
University of California Santa Barbara
United States

Dr Luis Rodolfo Garcia
College of Science and Engineering
Texas A&M University, Corpus Christi
United States

Dr Hafiz M. R. Khan
Department of Biostatistics
Florida International University
United States

Dr. Xuewen Lu
Dept. of Mathmatics and Statistics
University of Calgary
Canada

Dr. Marc Kachelriess
X-Ray Imaging and Computed Tomography
German Cancer Research Center
Germany

Dr. Nadia Pisanti
Department of Computer Science
University of Pisa
Italy

Dr. Frederik J. Beekman
Radiation Science & Technology
Delft University of Technology, *Netherlands*

Dr. Ian Mitchell
Department of Computer Science
Middlesex University London
United Kingdom

Professor Wee SER
Nanyang Technological University
Singapore

Dr Xiaocong Fan
The Pennsylvania State University
United States

Dr Julia Johnson
Dept. of Mathematics & Computer Science
Laurentian University, Ontario
Canada

Dr Chen Yanover
Machine Learning for Healthcare and Life Sciences
IBM Haifa Research Lab
Israel

Dr Vandana Janeja
University of Maryland, Baltimore
United States

Dr Nikolaos Georgantas
Senior Research Scientist at INRIA, Paris-Rocquencourt
France

Dr Zeyad Al-Zhour
College of Engineering, The University of Dammam
Saudi Arabia

Dr Zdenek Zdrahal
Knowledge Media Institute,
The Open University, Milton Keynes
United Kingdom

Dr Farouk Yalaoui
Institut Charles Dalaunay
University of Technology of Troyes
France

Dr Jai N Singh
Barry University, Miami Shores, Florida
United States

Dr. Laurence Devillers
Computer Science, Paris-Sorbonne University
France

Dr. Hans-Theo Meinholz
Systems analysis and middleware
Fulda University of Applied Sciences
Germany

Dr. Katsuhiro Honda
Department of Computer science and Intelligent Systems
Osaka Prefecture University
Japan

Dr. Uzay Kaymak
Department of Industrial Engineering & Innovation
Sciences, Technische Universiteit Eindhoven University
of Technology, *Netherlands*

Dr. Fernando Beltran
University of Auckland Business School
New Zealand

Dr. Weiru Liu
Department of Computer Science
University of Bristol
United Kingdom

Dr. Aladdin Ayesh
School of Computer Science and Informatics
De Montfort University, Leicester
United Kingdom

Dr. David Glass
School of Computing, Ulster University
United Kingdom

Dr. Sushmita Mukherjee
Department of Biochemistry
Weil Cornor Medical College, New York
United States

Dr. Rattikorn Hewett
Dept. of Computer Science
Texas Tech University
United States

Dr. Cathy Bodine
Department of Bioengineering
University of Colorado
United States

Dr. Daniel C. Moos
Education Department
Gustavus Adolphus College
United States

Dr. Anne Clough
Department of Mathematics,
Statistics and Computer Science, Marquette University
United States

Dr. Jay Rubinstein
University of Washington
United States

Dr. Frederic Maire
Department of Electrical Engineering and Computer Science
Queensland University of Technology
Australia

Dr. Bradley Alexander
School of Computer Science
University of Adelaide
Australia

Dr. Erich Peter Klement
Department of Knowledge-Based Mathematical Systems
Johannes Kepler University Linz
Austria

Dr. Ibrahim Ozkan
Department of Economics
Hacettepe University
Canada

Dr. Sattar B. Sadkhan
Department of Information Networks
University of Babylon
Iraq

Dr. Mikhail Bilenko
Machine Intelligence Research (MIR) Group, Yandex
Russia

Anne Hakansson
Department of Software and Computer systems
KTH Royal Institute of Technology
Sweden

Dr. Adnan K. Shaout
Department of Electrical and Computer Engineering
University of Michigan-Dearborn
United States

Dr. Tomasz G. Smolinski
Department of Computer and Information Sciences
Delaware State University
United States

Dr. Yi Ming Zou
Department of Mathematical Sciences
University of Wisconsin
United States

Mohamed A. Zohdy
Department of Electrical and Systems Engineering
Oakland University
United States

Dr. Krysta M. Svore
Microsoft Quantum – Redmond Microsoft
United States

Dr. John Platt
Machine learning, Google
United States

Dr. Wen-tau Yih
Natural language processing
Allen Institute for Artificial Intelligence
United States

Dr. Matthew Richardson
Natural Language Processing Group, Microsoft
United States

Amer Dawoud
Department of Computer Engineering
University of Southern Mississippi
United States

Dr. Jinsuk Baek
Department of Computer Science
Winston-Salem State University
United States

Dr. Harry Wechsler
Department of Computer Science
George Mason University
United States

Dr. Omer Weissbrod
Department of Computer Science
Israel Institute of Technology
Israel

Dr. Marina Papatriantafilou
Department of Computer Science and Engineering
Chalmers University of Technology
Sweden

Dr. Florin Manea
Dependable Systems Group, Dept. of Computer Science
Kiel University Christian-Albrechts
Germany

Prof. Dr. Hans Kellerer
Department of Statistics and Operations Research
University of Graz
Austria

Dr. Dimitris Fotakis
School of Electrical and Computer Engineering
National Technical University of Athens
Greece

Dr. Faisal N. Abu-Khzam
Department of Computer Science and Mathematics
Lebanese American University, Beirut
Lebanon

Dr. Tatsuya Akutsu
Bioinformatics Center, Institute for Chemical Research
Kyoto University, Gokasho
Japan

Dr. Francesco Bergadano
Dipartimento di Informatica,
Università degli Studi di Torino
Italy

Dr. Mauro Castelli
NOVA Information Management School (NOVA IMS),
Universidade Nova de Lisboa
Portugal

Dr. Stephan Chalup
School of Electrical Engineering and Computing,
The University of Newcastle
Australia

Dr. Yael Dubinsky
Department of Computer Science
Israel Institute of Technology
Israel

Dr. Francesco Bergadano
Department of Computer Science
University of Turin
Italy

Dr. Xiaowen Chu
Department of Computer Science, Hong Kong Baptist
University, Kowloon Tong
Hong Kong

Dr. Alicia Cordero
Instituto de Matemática Multidisciplinar, Universitat
Politècnica de València
Spain

Dr. Sergio Rajsbaum
Instituto de Matemáticas, Universidad Nacional
Autónoma de México
Mexico

Dr. Tadao Takaoka
College of Engineering
University of Canterbury, Christchurch
New Zealand

Dr. Bruce Watson
FASTAR Group, Information Science Department,
Stellenbosch University
South Africa

Dr. Tin-Chih Toly Chen
Department of Industrial Engineering and Management,
National Chiao Tung University, Hsinchu City
Taiwan

Dr. Louxin Zhang
Department of Mathematics, National University of
Singapore

DISCLAIMER

All the contributions are published in good faith and intentions to promote and encourage research activities around the globe. The contributions are property of their respective authors/owners and the journal is not responsible for any content that hurts someone’s views or feelings etc.

SOCIETY FOR SCIENCE AND EDUCATION
UNITED KINGDOM

TMLAI TRANSACTIONS ON
MACHINE LEARNING AND ARTIFICIAL INTELLIGENCE

Volume 7 No 4
ISSN 2054-7390

A Population-Based Multicriteria Algorithm for Alternative Generation

[1]Julian Scott Yeomans
[1] *OMIS Area, Schulich School of Business, York University, Toronto, ON, M3J 1P3 Canada;*
syeomans@schulich.yorku.ca

ABSTRACT

Complex problems are frequently overwhelmed by inconsistent performance requirements and incompatible specifications that can be difficult to identify at the time of problem formulation. Consequently, it is often beneficial to construct a set of different options that provide distinct approaches to the problem. These alternatives need to be close-to-optimal with respect to the specified objective(s), but be maximally different from each other in the solution domain. The approach for creating maximally different solution sets is referred to as modelling-to-generate-alternatives (MGA). This paper introduces a computationally efficient, population-based multicriteria MGA algorithm for generating sets of maximally different alternatives.

Keywords: Multicriteria Objectives, Population-based algorithms, Modelling-to-generate-alternatives.

1 Introduction

Complex problems frequently contain inconsistent and incompatible design specifications that can be difficult to incorporate into supporting mathematical formulations [1]-[5]. While "optimal" solutions can be calculated for the mathematical models, they may not provide a truly best solution to the "real" problem as there are usually unmodeled components not apparent when the initial mathematical models are formulated [1][2][6]. Generally, it is better to construct a small number of dissimilar alternatives that provide distinct viewpoints for the particular problem [3][7]. These distinct solutions should be close-to-optimal with respect to the specified objective(s), but be maximally different from each other within the solution domain. Numerous approaches collectively referred to as *modelling-to-generate-alternatives* (MGA) have been proposed to satisfy this multi-solution requirement [6]-[8]. The principal motivation behind MGA is to construct a set of options that are "good" with respect to all specified objective(s), but are fundamentally different from each other in the decision space. Decision-makers have to conduct a subsequent assessment of this set of alternatives to determine which specific alternative(s) most closely achieve their specific requirements. Consequently, MGA approaches are considered as decision support methods rather than as solution determination processes as assumed in explicit optimization.

The initial MGA algorithms used straightforward, iterative methods for constructing alternatives by incrementally re-solving their procedures whenever a new solution needed to be generated [6]-[10]. These iterative approaches imitated the seminal MGA method of Brill *et al.* [8] where, after the initial mathematical model had been optimized, all supplementary alternatives were produced one-at-a-time.

DOI: 10.14738/tmlai.74.6733
Publication Date: 14th July, 2019
URL: http://dx.doi.org/10.14738/tmlai.74.6733

Consequently, these incremental approaches all required *n*+1 iterations of their respective algorithms – initially to optimize the original problem, then to produce each of the subsequent *n* alternatives [7][11]-[18].

Subsequently, it was demonstrated how a set of maximally different alternatives could be generated using *any* population-based algorithm by permitting the generation of the overall optimal solution together with *n* distinct alternatives in a single computational run irrespective of the value of *n* [19]-[23]. In [24] a new data structure was created that permits alternatives to be *simultaneously* constructed by population-based solution methods and this was incorporated into a dual-criterion procedure in [25].

In this paper, a multicriteria, objective is introduced that combines the data structure into a simultaneous solution approach to create a new stochastic MGA algorithm. The max-sum components of the objective produce a maximum distance between alternatives by ensuring that the total deviation between all of the variables in all of the alternatives is collectively large. This does not, however, preclude the occurrence of relatively small (or zero) deviations between certain individual variables within certain solutions. In contrast, max-min objectives force a maximum distance between every variable over all solutions. By considering the multiple objectives simultaneously, the alternatives created can be forced as far apart as possible for all variables in general and the closest distance in the worst case between any solutions will never be less than the value obtained for the max-min objective. This stochastic MGA algorithmic approach proves to be extremely computationally efficient.

2 Modelling to Generate Alternatives

Mathematical optimization focuses almost exclusively on producing single optimal solutions to single-objective problems or, equivalently, constructing sets of noninferior solutions to multi-objective formulations [2][5][8]. While these conventions create solutions to the mathematical formulations as derived, whether the outputs provide "best" solutions to the original "real" problems remains less convincing [1][2][6][8]. Most "real world" decision situations possess numerous system conditions that can never be completely accounted for in mathematical constructions [1], [5]. In addition, it may not be possible to account for all of the subjective requirements as there are frequently numerous adversarial stakeholders and incompatible components to incorporate. Most subjective aspects remain unquantified and unmodelled in the mathematical formulations. This frequently happens when conclusions must be based not only on modelled objectives, but also upon more incongruent stakeholder predilections and socio-political-economic aspects [7]. Several "real life" instances of these idiosyncratic modelling features are described in [6][8]-[10].

When potentially unaccounted objectives and unmodelled components exist, non-traditional techniques are needed to scour the decision region for not only noninferior sets of alternatives, but also for solutions that are clearly *sub-optimal* to the modelled problem. Specifically, any search for alternatives to problems known or suspected to contain unmodelled components must concentrate not only on non-inferior sets of solutions, but also necessarily on explicit explorations of the problem's inferior solution domain.

To illustrate the consequences of unmodelled objectives on a decision search, assume that the optimal solution for a maximization problem is $\boldsymbol{X}^*$ with objective value $Z1^*$ [24]. Suppose a second, unquantified, maximization objective $Z2$ exists that represents some "politically acceptable" factor. Assume that the solution, $\boldsymbol{X}^a$, belonging to the 2-objective noninferior set, exists that corresponds to a best compromise solution if both objectives could have been simultaneously considered. Although $\boldsymbol{X}^a$ would be considered as the best solution to the real problem, in the actual mathematical model it would appear inferior to solution ***X****, since $Z1^a \leq Z1^*$. Therefore, when unquantified components are included in the decision-making process, inferior decisions to the mathematically modelled problem could be optimal to the underlying "real" problem. Thus, when unquantified issues and unmodelled objectives could exist, alternative solution procedures are required to not only explore the solution domain for noninferior solutions to the modelled problem, but also to concurrently search the solution domain for inferior solutions. Population-based algorithms prove to be proficient solution methods for concurrent searches throughout a decision space.

The prime directive for MGA is the construction of a practicable set of options that are quantifiably good when evaluated with respect to the modelled objectives, yet remain as different as possible from each other within the decision space. By achieving this task, the resultant set of alternatives can supply quite different perspectives with respect to the modelled objective(s) yet very differently with respect to the potentially unmodelled aspects. By creating these good-but-different options, the decision-makers can then identify specific desirable qualities within the alternatives that might satisfy the unmodelled objectives to varying degrees of stakeholder tolerability.

To motivate the MGA process, it is necessary to more formally characterize the mathematical definition of its goals [6][7]. Assume that the optimal solution to the original mathematical formulation is $\boldsymbol{X}^*$ producing a corresponding objective value of $\boldsymbol{Z}^* = F(\boldsymbol{X}^*)$. The resulting model can then be solved to produce an alternative solution, $\boldsymbol{X}$, that is maximally different from $\boldsymbol{X}^*$:

$$\text{Maximize} \quad \Delta(\boldsymbol{X}, \boldsymbol{X}^*) = \sum_i (X_i - X_i^*)^2 \tag{1}$$

$$\text{Subject to:} \quad \boldsymbol{X} \in D \tag{2}$$

$$| F(\boldsymbol{X}) - \boldsymbol{Z}^* | \leq T \tag{3}$$

where Δ represents an appropriate difference function (shown in (1) as an absolute difference) and T is a tolerance target relative to the original optimal objective value $\boldsymbol{Z}^*$. T is a user-specified limit that determines what proportion of the inferior region needs to be explored for acceptable alternatives. This difference function concept can be extended into a difference measure between any *set of alternatives* by replacing $\boldsymbol{X}^*$ in the objective of the maximal difference model and calculating the overall minimum absolute difference (or some other function) of the pairwise comparisons between corresponding variables in each pair of alternatives – subject to the condition that each alternative is feasible and falls within the specified tolerance constraint.

The population-based multicriteria MGA procedure to be introduced is designed to generate a pre-determined small number of close-to-optimal, but maximally different alternatives, by adjusting the value of T and solving the corresponding maximal difference problem instance by exploiting the population structure of the algorithm. The survival of solutions depends upon how well the solutions perform with respect to the problem's originally modelled objective(s) and simultaneously by how far away they are from all of the other alternatives generated in the decision space.

3 Population-based, Multicriteria MGA Algorithm

In this section, a data structure is employed that enables a multicriteria MGA solution approach via any population-based algorithm [24]. Suppose that it is desired to produce P alternatives that each possess n

decision variables and that the population algorithm is to possess K solutions in total. That is, each solution contains one set of P maximally different alternatives. Let $\boldsymbol{Y}_k$, $k = 1,\dots, K$, represent the k^{th} solution in the population which is comprised of one complete set of P different alternatives. Namely, if $\boldsymbol{X}_{kp}$ is the p^{th} alternative, $p = 1,\dots, P$, of solution k, $k = 1,\dots, K$, then $\boldsymbol{Y}_k$ can be represented as

$$\boldsymbol{Y}_k = [\boldsymbol{X}_{k1}, \boldsymbol{X}_{k2},\dots, \boldsymbol{X}_{kP}] \ . \tag{4}$$

If X_{kjq}, $q = 1,\dots, n$, is the q^{th} variable in the j^{th} alternative of solution k, then

$$\boldsymbol{X}_{kj} = (X_{kj1}, X_{kj2},\dots, X_{kjn}) \ . \tag{5}$$

Consequently, the entire population, $\boldsymbol{Y}$, consisting of K different sets of P alternatives can be expressed in vectorized form as,

$$\boldsymbol{Y'} = [\boldsymbol{Y}_1, \boldsymbol{Y}_2,\dots, \boldsymbol{Y}_K] \ . \tag{6}$$

The following population-based MGA method produces a pre-determined number of close-to-optimal, but maximally different alternatives, by modifying the value of the bound T in the maximal difference model and using any population-based method to solve the corresponding, maximal difference problem. The multicriteria MGA algorithm that follows constructs a pre-determined number of maximally different, close-to-optimal alternatives, by modifying the bound value T in the maximal difference model and using any population-based technique to solve the corresponding maximal difference problem. Each solution in the population comprises one set of p different alternatives. By exploiting the co-evolutionary aspects of the algorithm, the algorithm evolves each solution toward sets of dissimilar local optima within the solution domain. In this processing, each solution alternative mutually experiences the search steps of the algorithm. Solution survival depends upon both how well the solutions perform with respect to the modelled objective(s) and by how far apart they are from every other alternative in the decision space.

A straightforward process for generating alternatives solves the maximum difference model iteratively by incrementally updating the target T whenever a new alternative needs to be produced and then re-solving the resulting model [24]. This iterative approach parallels the original Hop, Skip, and Jump (HSJ) MGA algorithm of Brill *et al*. [8] in which the alternatives are created one-by-one through an incremental adjustment of the target constraint. While this approach is straightforward, it entails a repetitive execution of the optimization algorithm [7][12][13]. To improve upon the stepwise HSJ approach, a concurrent MGA technique was subsequently designed based upon co-evolution [13][15][17]. In a co-evolutionary approach, pre-specified stratified subpopulation ranges within an algorithm's overall population are established that collectively evolve the search toward the specified number of maximally different alternatives. Each desired solution alternative is represented by each respective subpopulation and each subpopulation undergoes the common processing operations of the procedure. The survival of solutions in each subpopulation depends simultaneously upon how well the solutions perform with respect to the modelled objective(s) and by how far away they are from all of the other alternatives. Consequently, the evolution of solutions in each subpopulation toward local optima is directly influenced by those solutions contained in all of the other subpopulations, which forces the concurrent co-evolution of each subpopulation towards good but maximally distant regions within the decision space according to the maximal difference model [7]. Co-evolution is also much more efficient than a sequential HSJ-style approach in that it exploits the inherent population-based searches to concurrently generate the entire set of maximally different solutions using only a single population [12][17].

While concurrent approaches can exploit population-based algorithms, co-evolution can only occur within each of the stratified subpopulations. Consequently, the maximal differences between solutions in different subpopulations can only be based upon aggregate subpopulation measures. Conversely, in the following simultaneous MGA algorithm, each solution in the population contains exactly one entire set of alternatives and the maximal difference is calculated only for that particular solution (i.e. the specific alternative set contained within that solution in the population). Hence, by the evolutionary nature of the population-based search procedure, in the subsequent approach, the maximal difference is simultaneously calculated for the specific set of alternatives considered within each specific solution – and the need for concurrent subpopulation aggregation measures is avoided.

Using the data structure terminology, the steps for the multicriteria MGA algorithm are as follows [14][19]-[24]. It should be readily apparent that the stratification approach employed by this method can be easily modified for any population-based algorithm.

Initialization Step. Solve the original optimization problem to find its optimal solution, $\boldsymbol{X^*}$. Based upon the objective value $F(\boldsymbol{X^*})$, establish P target values. P represents the desired number of maximally different alternatives to be generated within prescribed target deviations from the $\boldsymbol{X^*}$. Note: The value for P has to have been set *a priori* by the decision-maker.

Without loss of generality, it is possible to forego this step and to use the algorithm to find $\boldsymbol{X^*}$ as part of its solution processing in the subsequent steps. However, this significantly increases the number of iterations of the computational procedure and the initial stages of the processing become devoted to finding $\boldsymbol{X^*}$ while the other elements of each population solution are retained as essentially “computational overhead”.

Step 1. Create an initial population of size K where each solution contains P equally-sized partitions. The partition size corresponds to the number of decision variables in the original optimization problem. $\boldsymbol{X_{kp}}$ represents the p^{th} alternative, $p = 1,\ldots,P$, in solution $\boldsymbol{Y_k}$, $k = 1,\ldots,K$.

Step 2. In each of the K solutions, evaluate each $\boldsymbol{X_{kp}}$, $p = 1,\ldots,P$, with respect to the modelled objective. Alternatives meeting their target constraint and all other problem constraints are designated as *feasible*, while all other alternatives are designated as *infeasible*.

Note: A solution can be designated as feasible only if all of the alternatives contained within it are feasible.

Step 3. Apply an appropriate elitism operator to each solution to rank order the best individuals in the population. The best solution is the feasible solution containing the most distant set of alternatives in the decision space (the distance measures are defined in Step 5).

Note: Because the best-solution-to-date is always retained in the population throughout each iteration, at least one solution will always be feasible. Furthermore, a feasible solution based on the initialization step can be constructed using P repetitions of $\boldsymbol{X^*}$.

Step 4. Stop the algorithm if the termination criteria (such as maximum number of iterations or some measure of solution convergence) are met. Otherwise, proceed to Step 5.

Step 5. For each solution $\boldsymbol{Y_k}$, $k = 1,\ldots, K$, calculate R Max-Min and/or Max-Sum distance measures, D^r_{k}, $r = 1,\ldots, R$, between all of the alternatives contained within the solution.

As an illustrative example for calculating the multicriteria distance measures, compute:

$$D^1_k = \Delta^1 (\boldsymbol{X}_{ka}, \boldsymbol{X}_{kb}) = \min_{a,b,q} | X_{kaq} - X_{kbq} | , \qquad a = 1,\ldots,P,\ b = 1,\ldots,P,\ q = 1,\ldots,n, \tag{7}$$

$$D^2_k = \Delta^2 (\boldsymbol{X}_{ka}, \boldsymbol{X}_{kb}) = \sum_{a=1 to P} \sum_{b=1 to P} \sum_{q=1\ldots n} | X_{kaq} - X_{kbq} | \tag{8}$$

and

$$D^3_k = \Delta^3 (\boldsymbol{X}_{ka}, \boldsymbol{X}_{kb}) = \sum_{a=1 to P} \sum_{b=1 to P} \sum_{q=1\ldots n} (X_{kaq} - X_{kbq})^2. \tag{9}$$

D^1_k denotes the minimum absolute distance, D^2_k represents the overall absolute deviation, and D^3_k determines the overall quadratic deviation between all of the alternatives contained within solution *k*.

Alternatively, the distance functions could be calculated using some other appropriately defined measures.

Step 6. Let $D_k = G(D^1_k, D^2_k, D^3_k, \ldots, D^R_k)$ represent the multicriteria objective for solution *k*. Rank the solutions according to the distance measure D_k objective – appropriately adjusted to incorporate any constraint violation penalties for infeasible solutions. The goal of maximal difference is to force alternatives to be as far apart as possible in the decision space from the alternatives of each of the partitions within each solution This step orders the specific solutions by those solutions which contain the set of alternatives which are most distant from each other.

Step 7. Apply applicable algorithmic “change operations” to each solution within the population and return to Step 2.

4 Conclusion

Complex problem solving inherently involves incongruent features and indeterminate performance specifications. These situations commonly possess inconsistent structural components that are difficult to incorporate into supporting decision systems. There are always unmodelled features, not apparent during model formulation, that can significantly impact the adequacy of its solutions. These components force decision-makers to combine uncertainties into their solution process prior to any problem resolution. When faced with these inconsistencies, the likelihood that any single solution can concurrently satisfy all of the ambiguous system requirements to “optimum” is quite low. Therefore, any decision support approach must somehow address these complicating aspects in some way, while simultaneously being flexible enough to include the intrinsic planning uncertainties.

This paper has provided a new multicriteria approach and an updated MGA procedure. This new computationally efficient MGA method establishes how population-based algorithms can simultaneously construct entire sets of close-to-optimal, maximally different alternatives by exploiting the evolutionary characteristics of any population-based solution approach. In this MGA role, the multicriteria objective can efficiently generate the requisite set of dissimilar alternatives, with each generated solution providing an entirely different outlook to the problem. The max-sum criteria ensures that the distances between the alternatives created by this algorithm are good in general, while the max-min criteria ensures that the distances between the alternatives are good in the worst case. The value of an absolute-type function delivers a physical interpretation to its measure of distance. Since population-based procedures can be applied to a wide spectrum of problem types, the practicality of the multicriteria algorithm can be extended to many “real world” applications. These extensions will be considered in future studies.

REFERENCES

[1]. Brugnach, M., A. Tagg, F. Keil, and W.J. De Lange, *Uncertainty matters: computer models at the science-policy interface*. Water Resources Management, 2007. 21: p. 1075-1090.

[2]. Janssen, J.A.E.B., M.S. Krol, R.M.J. Schielen, and A.Y Hoekstra, *The effect of modelling quantified expert knowledge and uncertainty information on model based decision making*. Environmental Science and Policy, 2010. 13(3): p. 229-238.

[3]. Matthies, M., C. Giupponi, and B. Ostendorf, *Environmental decision support systems: Current issues, methods and tools*. Environmental Modelling and Software, 2007. *22(2)*: p. 123-127.

[4]. Mowrer, H.T., *Uncertainty in natural resource decision support systems: Sources, interpretation, and importance*. Computers and Electronics in Agriculture, 2000. *27(1-3)*: p. 139-154.

[5]. Walker, W.E., P. Harremoes, J. Rotmans, J.P. Van der Sluis, M.B.A. Van Asselt, P. Janssen, and M.P. Krayer von Krauss, *Defining uncertainty – a conceptual basis for uncertainty management in model-based decision support*. Integrated Assessment, 2003. 4(1): p. 5-17.

[6]. Loughlin, D.H., S.R. Ranjithan, E.D. Brill, and J.W. Baugh, *Genetic algorithm approaches for addressing unmodelled objectives in optimization problems*. Engineering Optimization, 2001. *33(5)*: p. 549-569.

[7]. Yeomans, J.S., and Y Gunalay, *Simulation-Optimization Techniques for Modelling to Generate Alternatives in Waste Management Planning*. Journal of Applied Operational Research, 2011. 3(1): p. 23-35.

[8]. Brill, E.D., S.Y. Chang, and L.D Hopkins, *Modelling to generate alternatives: the HSJ approach and an illustration using a problem in land use planning*. Management Science. 1982. 28(3): p. 221-235.

[9]. Baugh, J.W., S.C. Caldwell, and E.D Brill, *A Mathematical Programming Approach for Generating Alternatives in Discrete Structural Optimization*. Engineering Optimization. 1997, 28(1): p. 1-31.

[10]. Zechman, E.M., and S.R. Ranjithan, *An Evolutionary Algorithm to Generate Alternatives (EAGA) for Engineering Optimization Problems*. Engineering Optimization. 2004, 36(5): p. 539-553.

[11]. Gunalay, Y., J.S. Yeomans, and G.H. Huang, *Modelling to generate alternative policies in highly uncertain environments: An application to municipal solid waste management planning*. Journal of Environmental Informatics, 2012. *19(2)*: p. 58-69.

[12]. Imanirad, R., and J.S. Yeomans, *Modelling to Generate Alternatives Using Biologically Inspired Algorithms*. in *Swarm Intelligence and Bio-Inspired Computation: Theory and Applications*, X.S. Yang, Editor 2013. Amsterdam: Elsevier (Netherlands). p. 313-333.

[13]. Imanirad, R., X.S. Yang, and J.S. Yeomans, *A Computationally Efficient, Biologically-Inspired Modelling-to-Generate-Alternatives Method*. Journal on Computing. 2012, 2(2): p. 43-47.

[14]. Yeomans, J.S., *An Efficient Computational Procedure for Simultaneously Generating Alternatives to an Optimal Solution Using the Firefly Algorithm*, in *Nature-Inspired Algorithms and Applied Optimization*, Yang, X.S. Editor 2018. Heidelberg (Springer), Germany. p. 261-273.

[15]. Imanirad, R., X.S. Yang, and J.S. Yeomans, *A Co-evolutionary, Nature-Inspired Algorithm for the Concurrent Generation of Alternatives*. Journal on Computing. 2012, 2(3): p. 101-106.

[16]. Imanirad, R., X.S. Yang, and J.S. Yeomans, *Modelling-to-Generate-Alternatives Via the Firefly Algorithm*. Journal of Applied Operational Research. 2013. 5(1): p. 14-21.

[17]. Imanirad, R., X.S. Yang, and J.S. Yeomans, *A Concurrent Modelling to Generate Alternatives Approach Using the Firefly Algorithm*. International Journal of Decision Support System Technology. 2013, 5(2): p. 33-45.

[18]. Imanirad, R., X.S. Yang, and J.S. Yeomans, *A Biologically-Inspired Metaheuristic Procedure for Modelling-to-Generate-Alternatives*. International Journal of Engineering Research and Applications. 2013, 3(2): p. 1677-1686.

[19]. Yeomans, J.S., *Simultaneous Computing of Sets of Maximally Different Alternatives to Optimal Solutions*. International Journal of Engineering Research and Applications, 2017. *7(9)*: p. 21-28.

[20]. Yeomans, J.S., *An Optimization Algorithm that Simultaneously Calculates Maximally Different Alternatives*. International Journal of Computational Engineering Research, 2017. *7(10)*: p. 45-50.

[21]. Yeomans, J.S., *Computationally Testing the Efficacy of a Modelling-to-Generate-Alternatives Procedure for Simultaneously Creating Solutions*. Journal of Computer Engineering, 2018. *20(1)*: p. 38-45.

[22]. Yeomans, J.S., *A Computational Algorithm for Creating Alternatives to Optimal Solutions*. Transactions on Machine Learning and Artificial Intelligence, 2017. 5(5): p. 58-68.

[23]. Yeomans, J.S., *A Simultaneous Modelling-to-Generate-Alternatives Procedure Employing the Firefly Algorithm*, in *Technological Innovations in Knowledge Management and Decision Support*, Dey, N. Editor, 2019. Hershey, Pennsylvania (IGI Global), USA. p. 19-33.

[24]. Yeomans, J.S., *An Algorithm for Generating Sets of Maximally Different Alternatives Using Population-Based Metaheuristic Procedures*. Transactions on Machine Learning and Artificial Intelligence, 2018. 6(5): p. 1-9.

[25]. Yeomans, J.S., *A Bicriterion Approach for Generating Alternatives Using Population-Based Algorithms*. WSEAS Transactions on Systems, 2019. 18(4): p. 29-34.

SOCIETY FOR SCIENCE AND EDUCATION UNITED KINGDOM
TMLAI TRANSACTIONS ON MACHINE LEARNING AND ARTIFICIAL INTELLIGENCE
Volume 7 No 4
ISSN 2054-7390

Time Series Analysis on Nigeria Foreign Exchange Reserve

Ajao, I. O., Osunronbi, F.A., and Ibikunle, K. S. O.
Department of Mathematics and Statistics,
The Federal Polytechnic, Ado-Ekiti, Ado-Ekiti, Nigeria.
isaacoluwaseyiajao@gmail.com

ABSTRACT

Time series analysis was carried out on Nigeria External Reserves for the period of 1960 – 2018. An empirical investigation was conducted using time series data on Nigeria External Reserve for a period of 58 years. The techniques of estimation employed in the study include Phillips-Perron unit root test, Dickey-Fuller's test, the Autocorrelation function and the Partial Autocorrelation function for the model selection. The Box-Jenkins ARIMA methodology was used for forecasting the monthly data collected from 1960 to 2018. Result of the analysis revealed that the series became stationary at first difference. The diagnostic check showed that ARIMA (1, 1, 2) is appropriate or optimal model based on the Loglikelihood (LogLik), Akaike's Information Criterion (AIC), as well as the small standard error of the AR(1), MA(1) and MA(2) parameters. The performance of "forecast.Arima()" function in R gives the best model for Nigeria external reserve. Testing for other ARIMA models is necessary in order to establish the best. The downward movement in the forecasts of Nigeria external reserve would be helpful for policy makers in Nigeria.

Keyword: External reserve, ARIMA, stationarity, model selection, forecasts

1 Introduction

External reserves, also known as International Reserves, include international reserve assets of the monetary authority but exclude the foreign currency and the securities held by the public including the banks and corporate bodies. External reserves are needed to guard against possible financial crisis. Zubair and Olarenwaju (2014) tentatively identified ARIMA (1,2,2) model as a suitable model for modelling and forecasting Nigeria's external reserves using a monthly 50 years' data (January, 1960 – December, 2008). The Nigeria's external reserves was found to be on the increase and the paper further called on the Nigerian government to exercise fairness, justice, and equity in order to strengthen her economy. The model was fitted with over differenced log-transformed series which could affect the forecasting power of the ARIMA (1,2,2) model, therefore, the ARIMA (1,1,2) model would have been preferred since a single ordinary difference would have rendered the series stationary. Akpanta and Okorie (2015) identified ARI (5,1,0) model as a suitable model for modeling and forecasting the Nigeria's external reserves. Nigeria's 54 years' external reserve data from January, 1960 to December, 2013 was used to perform analysis in R. The ARI (5,1,0) model with the smallest AIC and BIC statistics was found to out-perform the ARIMA (5,1,1) model. One-year forecast was made with the best ARI (5,1,0) model and the Nigeria's external reserves was trending upwards. From the paper, the point forecast values are higher than the observed values. Interestingly the observed values are found to lie within the 90% confidence intervals. Iwueze et al, (2013)

DOI: 10.14738/tmlai.74.6566
Publication Date: 16th July, 2019
URL: http://dx.doi.org/10.14738/tmlai.74.6566

recommended the Auto Regressive integrated moving average (ARIMA) process of order (2,1,0) for forecasting the natural log-transformed Nigeria's external reserves, using II years data (from January, 1999 – December, 2008), where the Nigeria's foreign reserves were found to be on the increase. However, the point forecast from this model shows a large discrepancy from the observed and was attributed to the fall in income from petroleum products which is the main source of the Nigeria's external reserves. In the paper the ARIMA (2,1,0) though a candidate model could not have been the best, instead the ARIMA (2,1,2) would have been considered because the ACF plot of the first ordinary differenced log-transformed series showed a significant spike at lag 2 and cut-off thereafter. Ohakwe et al, (2013) modeled Nigerian External Reserves from the period of 1960 – 2010 using descriptive time series technique and Box-Jenkins (ARIMA) model. Etuk et al, (2013) identified and established the adequacy of the seasonal ARIMA (5,1,0)(0,1,1). For modeling and forecasting the amount of monthly rainfall in Port Harcourt, Nigeria. This paper therefore attempts to identify and construct a more reliable Box-Jenkins ARIMA (p,d,q) model that would best explain the underlying generating process and make forecast into the future of the Nigerian External Reserves. Sinha (2010) evaluate the state of the Indian economy throughout recession by analyzing different macro-economic factors such as External Reserves, inflation, exchange rate, fiscal deficit and capital markets. This study forecast some of the major economic variables by ARIMA modelling and presents a depiction of the Indian economy in the coming years. The results indicate that Indian economy is stimulating after a slowdown in the phase of global recession. It was forecasted that External reserves, fiscal deficit, capital markets and foreign investments will increase in 2010-2011. Giavazzi and Pagano (1990) studied the external reserves in Denmark and Ireland in the 1980s and showed that in these countries a drastic cut in public deficits led to a sharp increase in private consumption. The empirical studies that focus on the debt overhang effects of budget deficits try to analyse the nonlinear relationship between public spending and public debt with the growth rate of the economy. Borensztein (1990) provided a major and interesting attempt to test the debt overhang effect empirically. Using data for the Philippines, he found that the debt overhang hypothesis was largely valid. Specifically, he found that debt overhang had an adverse effect on private investment. The effect was strongest when private debt, rather than total debt, was used as a measure of debt overhang. Fairly similar results were obtained in another study carried out by Alfredo and Francisco (2004). They explored the relationship between external debt and growth for a number of Latin American and Caribbean economies. The result of their study showed that lower total external debt levels were associated with higher growth rates, and that this negative relationship was driven by the incidence of public external debt levels, and not by private external debt levels.

Ndung'u (1998) examined the dynamic impact of external debt accumulation on private investment and growth in Africa. He argued that the external debt problem in Africa has led to an investment pause and has reduced growth performance substantially. To strengthen his argument, he used results from recent empirical work by Elbadawi, et al. (1997) to show the dynamics of the problem and how a country moves from one side of the Laffer curve to the other and the effects on investment and growth. Once a country gets onto the wrong side of the Laffer curve and does not reverse the trend, the accumulated effects further affect growth performance. In another study, Iyoha (1999) adopted a simulation approach to investigate the impact of external debt on economic growth in sub-Saharan African countries. An important in this study was the significance of debt overhang variables in the investment equation, suggesting that mounting external debt depresses investment through both a "disincentive" effect and a "crowding out" effect. Policy simulation was undertaken to investigate the impact of alternative debt stock reduction scenarios (debt reduction packages of 5%, 10%, 20% and 50%) on investment and economic growth. It was found that debt stock reduction would have significantly increased investment and growth performance. Audu (2004) investigated the impact of external debt on economic growth and public investment in Nigeria. Usman and Ibrahim (2010) made a study of external reserves holding with implications for investment, inflation and exchange rate. Using Vector Error Correction (VEC) model they concluded that demand for external reserves in Nigeria "has been driven mainly by current account variability, real exchange rate and opportunity cost of holding reserves (measured by the difference between the real return on reserves and the real return on domestic investments)". They opined that their finding corroborates those of Adam and Leonce (2007) who stated that "demand for international resources in Africa is determined by Export, GDP growth and opportunity cost of holding reserves". Nzotta (2009) attempt to construct a time series model which was utilized to forecast the Nigeria External Reserves to get its future estimations up to fourth quarter of 2015. The study found on the figures collected through secondary sources from 1962 to 2008. ARIMA models were seek on the collected data and to conclude ARIMA (2, 1, 1) is create to be an appropriate model, which is then apply for forecasting purpose. The outcome of the future forecast explains the significant improvement in the fourth quarter of 2015. Usman and Ibrahim in their study say that "changes in external reserves show no significant relationship with inflation in Nigeria". They further added that although "external reserves position for Nigeria has no import on inflation rate but the domestic money supply should be a control variable to regular domestic inflation rate".

2 Materials and Methods

2.1 Autoregressive (AR) Process

The equation below is an example of an Autoregressive process

$$(Y_t - \delta\ ) = \alpha_1\ (Y_{t-1} - \delta) - +u_t$$

Where δ is the mean of *Y* and where u_t is an uncorrelated random error term with zero mean and constant variance σ^2 (i.e., it is *white noise*), then we say that Y_t follows a first-order autoregressive, or AR (1),stochastic process. Here the value of *Y* at time *t* depends on its value in the previous time period and a random term; the *Y* values are expressed as deviations from their mean value. In other words, this model says that the forecast value of *Y* at time *t* is simply some proportion (=α_1) of its value at time (*t* − 1) plus a random shock or disturbance at time *t*; again the *Y* values are expressed around their mean values. But if we consider this model,

$(y_t\ - \delta\ ) = \alpha_1\ (Y_{t-1} - \delta)\ +\ \alpha_2\ (Y_{t-2} - \delta) + u_t$ then we say that Y_t follows a second-order autoregressive, or AR (2)process. That is, the value of *Y* at time *t* depends on its value in the previous two time periods, the *Y* values being expressed around their mean value δ. In general, we can have

$(Y_t - \delta\ ) = \alpha_1\ (Y_{t-1} - \delta)\ +\ \alpha_2\ (Y_{t-2} - \delta)\ \ +\ldots+\ \alpha_P\ (Y_{t-P} - \delta)\ +\ \ u_t$ in which case Y_t is a pth-order autoregressive, or AR(p),process.

2.2 Moving Average (MA) process.

The AR process just discussed is not the only mechanism that may have generated *Y*. Suppose we model *Y* as follows:

$Y_t = \mu + \beta_0 u_t + \beta_1 u_{t-1}$

Where μ is a constant and u is the white noise stochastic error term. Here Y at time t is equal to a constant plus a moving average of the current and past error terms. Thus, we say that Y follows a first-order moving average,or an MA (1),process. But if Y follows the expression

$Y_t = \mu + \beta_0 u_t + \beta_1 u_{t-1} + \beta_2 u_{t-2}$, then it is an MA (2) process.

More generally,

$Y_t = \mu + \beta_0 u_t + \beta_1 u_{t-1} + \beta_2 u_{t-2} + \cdots + \beta_P u_{t-P}$

2.3 Autoregressive Integrated Moving Average (ARIMA) model

The ARMA models, described above can only be used for stationary time series data. However, in practice many time series such as those related to socio-economic and business show non-stationary behavior. Time series, which contain trend and seasonal patterns, are also non-stationary in nature. Thus from application view point ARMA models are inadequate to properly describe non-stationary time series, which are frequently encountered in practice. For this reason, the ARIMA model is proposed, which is a generalization of an ARMA model to include the case of non-stationarity as well. In ARIMA models a non-stationary time series is made stationary by applying finite differencing of the data points. Which is written as ARIMA (p, d, q).

Here, *p*, *d* and *q* are integers greater than or equal to zero and refer to the order of the Autoregressive, integrated, and moving average parts of the model respectively. The integer *d* controls the level of differencing. Generally, *d*=1 is enough in most cases. When *d*=0, then it reduces to an ARMA (*p, q*) model. It is widely used for non-stationary data, like economic and stock price series.

2.4 Box-Jenkins Methodology

After describing various time series models, the next issue to our concern is how to select an appropriate model that can produce accurate forecast based on a description of historical pattern in the data and how to determine the optimal model orders. Statisticians George Box and Gwilym Jenkins developed a practical approach to build ARIMA model, which best fit to a given time series and also satisfy the parsimony principle. Their concept has fundamental importance on the area of time series analysis and forecasting. The Box-Jenkins methodology does not assume any particular pattern in the historical data of the series to be forecasted. Rather, it uses a three step iterative approach of model identification, parameter estimation and diagnostic checking to determine the best parsimonious model from a general class of ARIMA models. This three-step process is repeated several times until a satisfactory model is finally selected. Then this model can be used for forecasting future values of the time series.

3 Data Analysis

The data used for this project is extracted from the bulletin of Central Bank of Nigeria (CBN) and the data is on Nigeria External Reserves (US million) from 1960 to 2017, a period of 58 years. All data analyses were done using R version 3.4.4

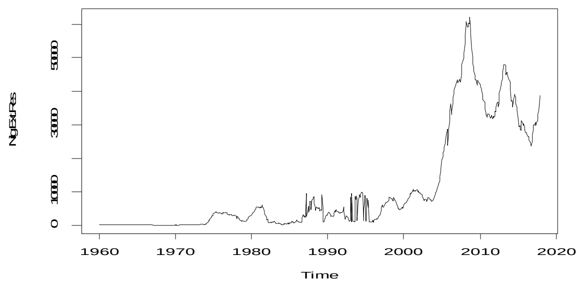


Fig.1: Time plot for Nigeria External Reserve data

From the time plot, it appears that the random fluctuations in the time series are not roughly constant in size over time, there is trend, the time series data does not appear to be stationary in mean and variance, as its level and variance appear not to be roughly constant over time. Therefore, there is need to difference this series in order to fit an ARIMA model. There is an upward movement in the Nigeria External Reserve series data so it is not stationary.

In order to establish if the time series data (Nigeria External Reserves) is stationary or not, the Phillips-Perron test for the null hypothesis that series data has a unit root against a stationary alternative is performed:

Table 1: Phillips-Perron Unit Root Test before differencing

Dickey-fuller statistic	P-value	Remark
-1.8498	0.6419	Not stationary

From the output, it can be seen that the p-value is 0.6419, which is greater than 0.05 and this indicates that there is no stationarity in the time series data.

Since the Nigeria External Reserve data is not stationary, there is a need to carry out differencing in order to make the time series data stationary.

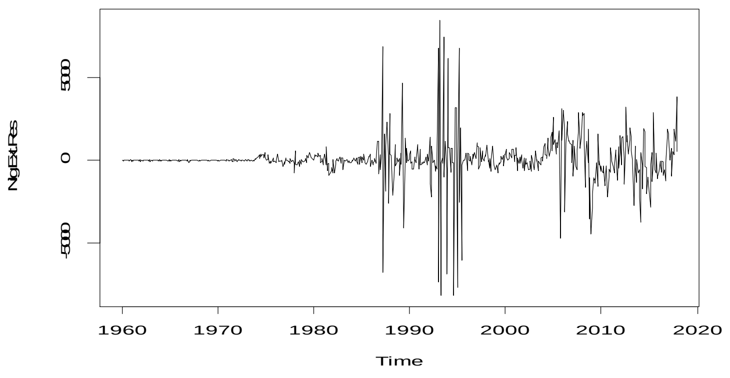


Fig.2: Time plot for Nigeria External Reserve data after first differencing

From the time plot, it appears that the random fluctuations in the time series are roughly constant in size over time, there is no trend, the time series data appears to be stationary in mean and variance, as its mean and variance appear to be roughly constant over time. Therefore, there is no need to difference this series in order to fit an ARIMA model. The Nigeria External Reserve series data is stationary.

Table 2: Phillips-Perron Unit Root Test after differencing

Dickey-fuller statistic	P-value	Remark
-30.6880	0.0100	Stationary

From the output, it can be seen that the p-value is 0.01, which is less than 0.05 and this indicates that the data is stationary.

4 Arima Model Selection

Since the series is stationary after first order differencing, the next step is to choose the ARIMA model that best fit the Nigeria External Reserve data by plotting a correlogram and partial correlogram for lags 1-20 and investigate what ARIMA model to use:

Table 3: Autocorrelations of the series

Lag	1	2	3	4	5	6	7	8	9	10
Autocorrelation	0.165	0.090	0.016	-0.003	0.215	-0.030	0.129	-0.048	0.050	0.118
Lag	11	12	13	14	15	16	17	18	19	20
Autocorrelation	-0.108	0.058	0.009	0.063	-0.041	0.028	-0.056	-0.039	0.141	-0.017

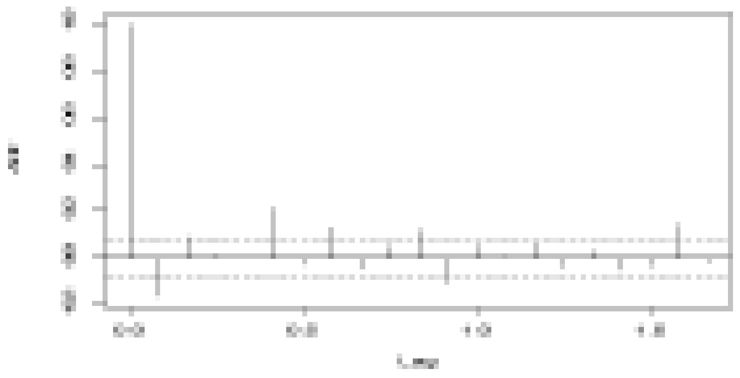


Fig. 3: Autocorrelation for the first differenced data

It can be seen from the correlogram that the autocorrelations from lags 1 and 2 exceed the significance bounds, and that the autocorrelations tail off to zero after lag 18. The autocorrelations for lags 2, 4, 6, 8 and 18 are positive, and decrease in magnitude with increasing lag. The autocorrelation function tails off to zero after lag 1.

Table 4: Partial autocorrelations of the series

Lag	1	2	3	4	5	6	7	8	9	10
Autocorrelation	-0.165	0.064	0.042	0.000	0.217	0.040	0.106	-0.023	0.020	0.091
Lag	11	12	13	14	15	16	17	18	19	20
Autocorrelation	-0.090	-0.036	0.033	0.046	-0.067	0.034	-0.069	-0.054	0.110	0.042

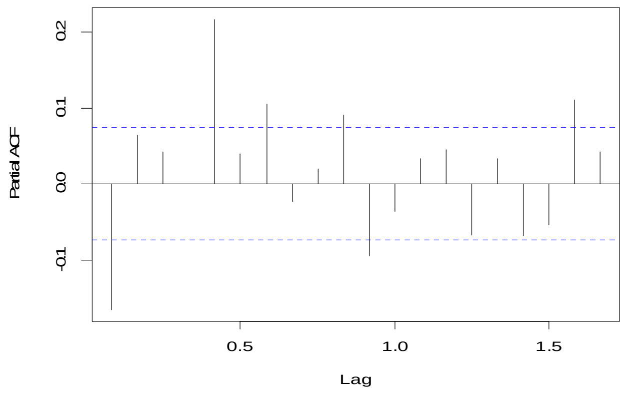


Fig. 4: Partial autocorrelation for the first differenced data

From the partial correlogram, it can be seen that the partial autocorrelation at lag 1 is negative and does not exceeds the significance bounds (-0.165), while the partial autocorrelation at lag 4 is also positive and also exceeds the significance bounds.

Since the correlogram tails off to zero after lag 1, and the partial correlogram is zero after lag 4, the ARIMA models possible for the time series data (Nigeria External Reserve) is $ARIMA(1,1,2)\ where\ p = 1, d = 1\ and\ q = 2$.

The ARIMA model can therefore be written as:

$$y_t = \alpha + \rho_1 y_{t-1} + \theta_1 \in_{t-1} + \theta_2 \in_{t-2} + \in_t$$

Where

5 Parameters Estimation

The ARIMA model chosen for the time series data (Nigeria External Reserve) is $ARIMA(1,1,2)$ and the model is given by $y_t = \alpha + \rho_1 y_{t-1} + \theta_1 \in_{t-1} + \theta_2 \in_{t-2} + \in_t$. The parameters ρ_1, θ_1 and θ_2 are estimated as follows:

Table 5: Parameter estimates from ARIMA (1,1,2) model

	Coef.	Std. error
Constant	65.8838	74.6583
AR		
lag1	0.8441	0.0378
MA		
lag1	-1.2655	0.0595
lag2	0.2819	0.0547

From the result above, the model is obtained as follows:

$$y_t = 65.8858 + 0.8441 y_{t-1} - 1.2655 \in_{t-1} + 0.2819 \in_{t-2} + \in_t$$

It can be seen that the $ARIMA(1,1,2)$ model fitted for the Nigeria External Reserve in US millions from 1960 to 2018 gives a good fit and it is the best selected model based on the small standard error of the AR(1), MA(1) and MA(2).

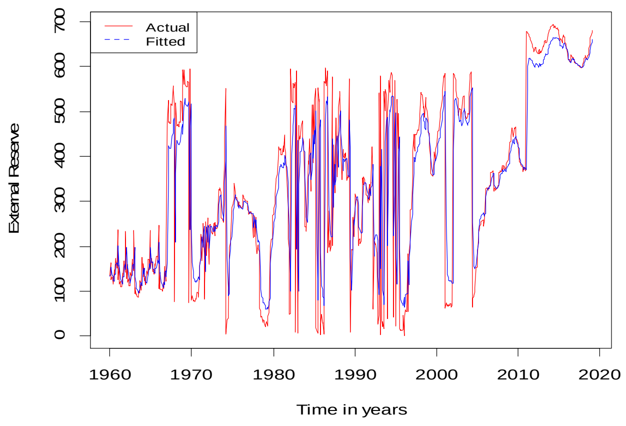


Fig. 5: Actual and the fitted series

Table 6: Parameter estimates from ARIMA (1,1,2) model

Model	Log-likelihood	AIC
ARIMA (2,1,0)	-4347.69	8701.38
ARIMA (1,1,0)	-4353.65	8711.30
ARIMA (0,1,1)	-4342.22	8688.44
ARIMA (2,1,1)	-4327.48	8662.97
ARIMA (1,1,2)	-4325.06	8660.21
ARIMA (0,1,0)	-4388.28	8778.57
ARIMA (1,1,3)	-4333.76	8673.53
ARIMA (1,1,1)	-4326.00	8662.00
ARIMA (0,0,1)	-4559.58	9125.07

6 Making Forecasts

The original time series for the Nigeria External Reserve indicates the reserve for 58 years (1960-2017). The forecast function gives a forecast of the Nigeria External Reserve for the next five years (NER January 2018-December 2022), as well as 80% and 90% prediction intervals for those predictions. The Nigeria External Reserve for December 2017 was 407 US million Dollars (the last observed value in the time series data), and the forecasted Nigeria External Reserve for the next five years is increasing with time as given by the ARIMA model.

The plot of the observed Nigeria External Reserve for the 58 years, as well as the Nigeria External Reserve that would be predicted for the next 5 years using $ARIMA(1,1,2)$ model is given below:

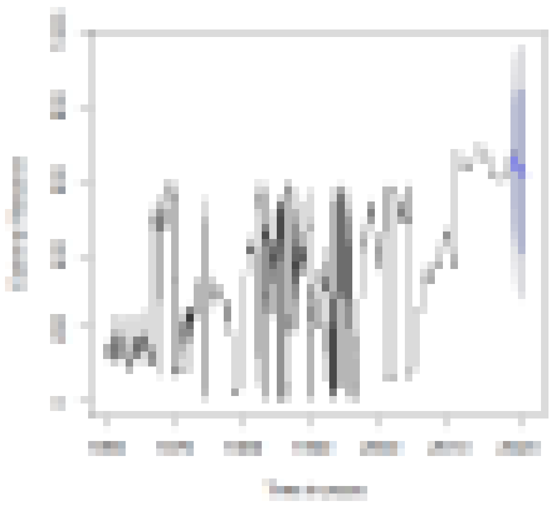


Fig. 6: The actual and the forecasts series

7 Summary and Conclusion

The major aim of this research work is to fit a robust time series model for Nigerian External Reserve. This is done by examining the stationarity in the Nigeria external reserve data. The study carries out an empirical analysis to determine the best ARIMA model for the Nigeria external reserve. This study investigated the Time series analysis of Nigeria External Reserves for the period of 1960 – 2017. The study set out to study the stationarity in the Nigeria External Reserve. An empirical investigation was conducted using time series data on Nigeria External Reserve from 1960 to 2017 a period of 57 years. The techniques of estimation employed in the study include Phillips-Perron unit root test, Dickey-Fuller's test, the Autocorrelation function and the Partial Autocorrelation function for the model selection. The Box-Jenkins ARIMA methodology was used for forecasting the monthly data collected from 1960 to 2018. Result of the analysis revealed that the series became stationary at first difference. The diagnostic checking has shown that ARIMA (1, 1, 2) is appropriate or optimal model based on the Loglikelihood (LogLik), Akaike's Information Criterion (AIC), as well as the small standard error of the AR(1), MA(1) and MA(2) parameters.

RECOMMENDATION

The Nigerian government should promote exportation of domestic products as a high exchange rate will make our goods more attractive in the foreign market and will increase foreign exchange earnings. Time series data analysts are encouraged to explore R package in order to discover better methods.

REFERENCES

[1] Adam E. and Léonce N. (2007). African Development Bank; United Nations. Economic Commission for Africa. (2007-11). Reserves accumulation in African countries: sources, motivations and effects.UN. ECA African Economic Conference (2007, Nov. 15-17: Addis Ababa, Ethiopia). Addis Ababa: UN.ECA,. http://hdl.handle.net/10855/21152"

[2] Ajao I.O; Obafemi O.S and Bolarinwa F. A. (2017). Modelling Dollar-Naira Exchange Rate in Nigeria. National statistical society vol. 1, 2017.Department of Mathematics and Statistics, The Federal Polytechnic, Ado-Ekiti, Ado-Ekiti, Nigeria.

[3] Akpanta, A.C and Okorie, I.E. (2015). ARI (p.d) Modelling and forecasting of Nigeria's External Reserves.World Journal of Probability and Statistics Research, Vol. 1, June, 2015. pp 1-9.

[4] Alfredo S. and Francisco R.B. (2004). External Debt and Economic Growth in Latin America. www.cbaeconomia.com/debt-latin.

[5] Alhaji, M. Nda (2006). Effective Reserves Management in Nigeria:Issues, Challenges and Prospects. Bullion (Central Bank of Nigeria) 30, No. 3 (July – September, 2006): 47 – 53.

[6] Audu, Isa (2004)."The Impact of External Debt on Economic Growth and Public Investment: The Case of Nigeria". African Institute for Economic Development and Planning (IDEP) Dakar Senegal. http://www.unidep.org.

[7] Borensztein E. (1990). Debt Overhang, Debt Reduction and Investment:The Case of the Philippines (September 1990). IMF Working Paper, Vol., pp. 1-27, 1990. Available at SSRN: https://ssrn.com/abstract=884986.

[8] Box, G.E.P and Cox, D.R (1964). An Analysis of Transformation. Journal of the Royal Statistical Society Series B. 26211 – 46.

[9] Box, G.E.P and Jenkins G.M (1976). Time series analysis; forecasting and control, rev. ed., Oakland, California: Holding-Day.

[10] Central Bank of Nigeria (1999). Amendment ACT No. 41.

[11] Davidson R. MacKinnon J. G. (2004).Econometric Theory and Methods. New York: Oxford University Press. p. 623. ISBN 0-19-512372-7.

[12] Doguwa, S.I and Alade S.O. (2015). On Time Series Modeling of Nigeria External Reserves. CBN Journal of Applied Statistics, vol. 6 No. 1 (a).

[13] Elbadawi I. (1997).Determinants of the real exchange rate in south Africa:Centre for the study of African economies institute of economics and statistics, university of oxford. www.researchgate.net/publication/5070621_Determinants_of_the_Real_Exchange_Rate_in_south_Africa.

[14] Etuk, E. H., Moffat, U.I and Chims, E.B (2013).Modeling monthly Rainfall Data of Port Harcourt, Nigeria by Seasonal Box-Jenkins Method:International Journal of Sciences, - 7:vol 2.

[15] Fischer, S. (2001). Opening Remarks, IMF Washington DC.

[16] Giavazzi F. and Pagano M. (1990).Can severe fiscal contractions be expansionary? Tales of two small European countries. www.nber.org/chapters/10973.

[17] IMF (2009).Balance of Payment and International Investment Position Manual,6th Edition (Washington: International Monetary Fund).

[18] Iwueze, I.S, Nwogu, E.C and Nlebedim, V.U (2013). Time Series modeling of Nigeria External Reserves.CBN Journal of Applied Statistics Vol. 4, No. 2:111-128.

[19] Kyland F. E. & Prescott E. C. (1977). Rules Rather than Direction:The Inconsistency of Optimal Plans. The Journal of Political Economy, 85(3), 473-492. Lanedell-Mills, Joshin N. (1989). The Demand for International Reserves and their Opportunity Cost. IMF Staff Papers, 36.

[20] Mei-Yin Li, & Jne-Shyan Wang (2008). Foreign Exchange Reserves and Inflation:An Empirical Study of five East Asia Economies. Aletheia University Taiwan and National Chengchi University, Taiwan.

[21] Ndungu'u N.S. (1998).The dynamic of external debt accumulation on private investment and growth in Africa. Department of Economics. University of Nairobi, Kenya. https://repository.uneca.org/bitstream/handle/10855/15520/bib-62116.

[22] Nzotta S. M. (2004). Money, Banking and Finance (Theory and Practice).Owerri: Hudson-Jude Nigeria Publishers.

[23] Ohakwe, J., Odo, I. Nwosu C. (2013). A Statistical Analysis of the Nigerian. External Reserves and the Impact of Military and Civilian Rule.Bulletin of Mathematical Sciences and Application, Vol. 2 No. 1 (2013), Pp. 63 – 84), ISSN: 2278 – 9634.

[24] Okeregwu B.A. and Etuk E.H. (2017). Time Series Analysis of Nigerian External Reserves. CARD International Journal of Educational Research and Management Technology (IJERMT)Department of Mathematics, Rivers State University, P.M.B 5080, Nkpolu-Oroworukwo, Port Harcourt. ISSN: 2545-5893 (Print) 2545-5877 (Online) Volume 2, Number 4, December 2017. http://www.casirmediapublishing.com.

[25] Phillips, P. C. B. and Perron, P. (1988)."Testing for a Unit Root in Time Series Regression" Biometrika . 75 (2): 335–346. doi: 10.1093/biomet/75.2.335.

[26] Sinha P. (2010). Modeling and forecasting of macro-economic variables of India:Before, during and after recession. Faculty of management studies, University of Delhi. https://mpra.ub.unimuenchen.de/.../modeling_and_forecasting_of_macro economic_variables_of_india.

[27] Usman A. and Ibrahim W. (2010). External reserves holdings in Nigeria:Implications for investment, inflation and exchange rate. Department of Economics, University of Ilorin, Ilorin, Nigeria. Department of Economics, Al-Hikmah University, Ilorin, Nigeria. Journal of Economics and International Finance Vol. 2(9), pp. 183-189, September 2010. Available online at http://www.academicjournals.org/JEIF. ISSN 2006-9812 ©2010 Academic Journals.

[28] Zubair, M.A and Olanrewaju, S.O. (2014). Time Series Model of Nigeria's External Reserves.The International Journal of Engineering and Sciences (IJES) vol. 3, ISSVE 1, ISSN(e): 2319 – 1813 ISSN (P): 22319-1805:1-10.

SOCIETY FOR SCIENCE AND EDUCATION UNITED KINGDOM
TMLAI TRANSACTIONS ON MACHINE LEARNING AND ARTIFICIAL INTELLIGENCE
Volume 7 No 4
ISSN 2054-7390

The Theory Graph Modeling and Programming Paradigms of Systems from Modules to the Application Areas

E. M. Lavrischeva
Doctor of phys.-мat. Sci., Professor of MIPT, General Sci. Specialist ISPRAS
Lavryscheva@gmail.com, lavr@ispras.ru

ANNOTATION

The mathematical basics of graph modeling and paradigm programming of applied systems (AS) are presented. The vertices of graph are been the functional elements of the systems and the arcs define the connections between them. The graph is represented by an adjacency and reach ability matrix. A number of graph of program structures and their representation by mathematical operations (unions, connections, differences, etc.) are shown. Given the characteristics of graph structures, complexes, units, and systems created from the modules of the graph. The method of modelling the system on the graph of modules, which describe in the programming languages (LP) and the advanced operations of association (link, assembling, make, building, config etc.). The standard of configuration (2012) Assembly of heterogeneous software elements in AS of different fields of knowledge is made. A brief description of modern and future programming paradigms for formal theoretical creation of systems from intelligent and service-components of the Internet is given. There are the new direction of modern paradigms programming in the near future.

Keywords: graph theory; adjacency matrix, reach ability; mathematical operations; configuration assembling; paradigm programming; future technologies.

1 Introduce. The Graph Theory and Paradigms of Programs

Programming theory is a mathematical science, the object of study of which is the mathematical abstraction of the functions of programs with a certain logical and information structure, focused on computer execution. With the advent of the LP began to develop new methods of analysis of algorithms of AS problems, the graph theory for the representation the structure AS by separate programs elements, displaying them in the vertices of the graph to create a complex structure of AS (programs, aggregate, large program, system, etc.). Programs elements of missile defense were first called modules, programs, then objects, components, services, etc. For the formal specification of these elements were formed the corresponding **programming paradigms**, allowing from the point of view of the theory and graphs to describe the problems of different AS (medicine, biology, chemistry, genetics, etc.).

2 Graph Theory of Programs from Modules

The basis for the creation of systems of modules was the method of assembling the graph (70-80 years of the last century) heterogeneous modules in specialized software packages (Lipaev V. V.) and in the system APROP of ES OS (IBM-360) [1, 2]. Formed Assembly programming [3-5], which "provided the building is

DOI: 10.14738/tmlai.74.6782
Publication Date: 03th August, 2019
URL: http://dx.doi.org/10.14738/tmlai.74.6782

already existing individual pieces of software (such reuses) in the complex structure" [6]. The interface of the modules was described initially in a special description language link, and then in equivalent operations: make BSD, Java (1996); config SPAROL, building, assembling Grid (2002), etc. [7-12]. And after 90-x there were standard languages of the description of these operations of IDL, API, WSDL and the standard statement of config of the IEEE 828-2012 standard (Configuration Management) for receiving a configuration file of any application system from ready modules, objects, components, services and other resources.

A module is a formally described program element that displays certain AS function that has the property of completeness and connectivity with other elements according to the data specified in the interface part of the description. From a mathematical point of view, a module is a mapping of a set of initial data X to a set of output Y in the form $M: X \rightarrow Y$.

A number of restrictions and conditions are imposed on *X*, *Y* and *M* to make the module an independent program element among other types of program objects [1-3].

Types of connections between modules via input and output parameters are as follows:

1) linking of control: $CP = K_1 + K_2$, where K_1 is the coefficient of the calling mechanism; K_2 is the coefficient of transition from the environment of the calling module to the environment of the called;
2) Linking of data: $CI = \sum_{i=1}^{n} K_i F(x_i)$, where *Ki* - the weight coefficient *i*ron of the parameter; $F(x_i)$ - the element function for the parameter xii.

Coefficients $K_{id} = 1$ – for simple variables and $K_{id} > 1$ – for complex variables (array, record, etc.). $F(x_{.l}) = 1$ if x_i - a simple variable and $F(x_i) > 1$ if complex.

The program, modular structure is given by the graph $G = (X, E)$, where *X* - a finite set of vertices; *E* - a finite subset of the direct product of *X* z on the set of relations on the arcs of the graph. The program structure represents a pair $S = (T, \chi)$, where *T* - a model of a program, modular structure; χ - a characteristic function given on the set of vertices *X* of the graph *G*.

The value of the characteristic function χ is defined as:

X(x) = 1 if the module with vertex $x \in X$ is included in the modular system;

X(x) = 0 if the module with vertex $x \in X$ is not included in the modular system and is not accessed from other modules.

Definition 1. Two models of program structures $T_1 = (G_I, Y_1, F_1)$ and $T_2 = (G_2, Y_2, F_2)$ are identical if $G_1 = G_2$, $Y_1 = Y_2$, $F_1 = F_2$. The T_1 model is isomorphic to the T_2 model if $G_1 = G_2$ between sets Y_1 and Y_2 exists an isomorphism ϕ, and for any $x \in X$ $F_2(x)=\phi(f_1(x))$.

Definition 2. Two program structure $S_1 = (T_1, \chi_1)$ and $S_2 = (T_2, \chi_2)$ are identical if $T_1 = T_2$, $\chi_1 = \chi_2$ and the structures S_1 and S_2 are isomorphic, then T_1 is isomorphic to T_2 and $\chi_1 = \chi_2$.

The concept of isomorphism of program structures and their models is used in the specification of the abstraction level at which operations on these structures are defined. For isomorphic graph objects, operations will be interpreted in the same way without orientation to a specific composition of program elements, provided that such operations are defined over pairs (G, χ). The software module is described in the LP and has an interface section in which external and internal parameters are set for data exchange between related modules through interface (Call/RMI) operations, etc.

The interface defines the connection of heterogeneous software modules according to the data and the way they are displayed by programming systems with the *LP*. Its main functions are: data transfer between program elements (modules), data conversion to the equivalent form and transition from the environment and platform of the called module to the caller and back. Functions of conversion of different, non-equivalent data types is carried out with the help of a previously developed library of 64 primitive functions for heterogeneous data types of *LP* in the APROP system [1-5] and included in the common system environments of the OS (IBM, MS, Oberon, UNIX, etc.).

In practice, the assembly method of software modules is performed by operations (link, make, assembling, config. weaver) special programs [1] OS libraries (OS ES, IBM, MS.Net, etc.), a builder of complex applications in OS RV for SM computers, complication modules for ERM "Elbrus" are used. In these operations and interface modules and data type conversion library [1, 2].

Next, we consider the mathematical theory of graphs of software modular structures and mathematical operations (union, projection, difference, etc.) implementation of ways of linking the graph modules and the semantics of the data transformation by the vertices of the graph. Software modules are described in modern LP and with help of the new paradigms programming.

2.1 Definition of a modular structure graph

To represent modular structures, we use the mathematical apparatus of graph theory, in which the graph G is treated as a pair of objects G = (X, E), where X - a finite set of vertices, and E is a finite subset of the direct product of X × X × Z - arcs of the graph, corresponding to a finite vertex (Fig. 1).

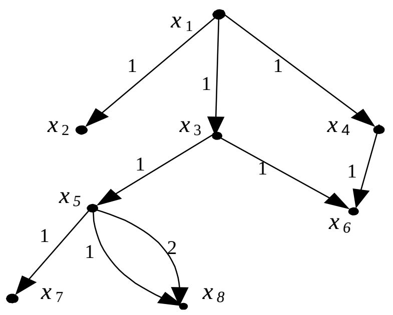


Fig.1. Graph of program from modules

The set of arcs of the graph have the form: $E = \{(x_1, x_2, 1), (x_l, x_3, 1), (x_5, x_8, 1), (x_5, x_8, 2)\}$ [1-7]. Based on this definition, we can say that the graph *G* is a multi-graph, since its two vertices can be connected by several arcs. To distinguish these arcs introduced their numbering positive integers – 1, 2. (Fig.1) and vertices of the graph $x_1, x_2, \ldots, x_8$ form a set of *X*. From the module corresponding to the vertex x_5, there are two calling operators to the modules, with vertices x_7, x_8.

Definition 3. A program aggregate is a pair S = (T, χ), where T - a model of the program modular structure of the aggregate; χ - a characteristic function defined on the set of vertices X of the graph of the modular structure G. The value of the χ function is defined as follows:

χ(x) = 1 if the module corresponding to the vertex x $\in$ X, - included in the unit;

χ(x) = 0 if the module corresponding to the vertex x $\in$ X, - not included in the software unit, but it is accessed from other modules previously included.

Definition 4. The model of the program structure of the program unit is an object described by the triple T = (G, Y, F), where G = (X, E) - a directed graph of a modular structure;

Y is a set of modules included in the program aggregate;

F is a correspondence function that puts an element of the set y at each vertex X of the graph.

$$\text{Function F maps X to Y,} \quad F: X \rightarrow Y. \tag{1}$$

In General, an element from Y can correspond to several vertices from the set X (which is typical for the dynamic structure of the aggregate) [5, 15, and 20].

The graph of software aggregates has the following properties:

1) graph G has one or more connectivity elements, each of which represents an acyclic graph, i.e. does not contain oriented cycles;
2) in each graph G is allocated a single vertex, which is called the root and is characterized by the fact that there are no arcs included in it and the corresponding module of the software unit is performed first;
3) cycles are allowed only for the case when some vertex has a recursive reference to itself. Typically, this feature is implemented by the compiler with the corresponding LP and this type of communication is not considered by the intermodule interface. Therefore, such arcs are not included in the graph. The exception to the consideration of other types of cycles is due to the fact that some modules will have to remember the history of their calls in order to return control correctly, which contradicts the properties of the modules;
4) an empty graph G_0 corresponds to an empty program structure.

Next, the graph G will be used to illustrate mathematical operations on modular structures. For Fig.2. three types of subgraphs are shown and their description is given.

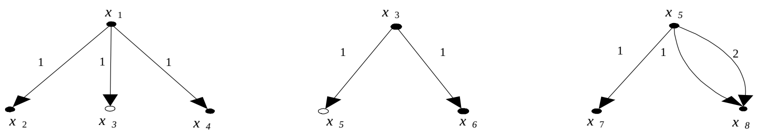


Fig. 2. The graphs of modules structures

A sub graph - a fragment of a software aggregate $G^s = (X^s, E^s)$ for whose functions one of two conditions is satisfied:

$C(S) = 1$, if χ(x) = 1 for any x of X;

C (*S*) = 0, if there is x such that χ(x) = 0;

R (S^s) = 0, if the modular structure is part of a higher-level structure and *R(S) =1* if the software assembly is ready to run.

Given these combinations C and R, the subgraph can be: open (*C =0, R = 0*); closed at the top (*C = 0, R= 1*); closed at the bottom (*C = 1, R = 0*).

The graph of the module (m) is represented as: $G^m = (X^m, E^m)$. It contains a single vertex $x \in X^m$ for which $\chi(x_j)=1$. This vertex is the root. An arc of the form (x_j, x_e, k) means calling the module to the corresponding vertex x_j , i.e. to the module with the vertex x_l. The dark circle on the+ graph corresponds to the vertex for which χ(x) = 1; light – χ(x)=0.

Program graph $G^p = (X^p, E^p)$ which is performed C (S^p) = 1; R (S^p) = 1. An example of a graph of such a program modular structure is shown in Fig. 1.

The graph of the complex $G^c= (X^c, E^c)$ consists of n connectivity components (n > 1), each of which is a graph and includes: $G^c = G_1^p \cup G_2^p \cup ,\dots, \cup G_n^p$,

where $X^c = X_1^p \cup X_2^p \cup ,\dots, \cup X_n^p$ и $E^c = E^p\ E_1^p \cup E_1^p\ E_2^p \cup ,\dots, \cup E_n^p$.

These definitions of the graph of the program module, program and complex are used for the process of assembling the modules. These concepts may differ from similar ones, which are considered in other contexts of the work.

2.2 Matrix representation of graphs from program elements of module type

To determine the main operations on software structures, we use the mathematical apparatus of the matrix representation of graphs in the form of an adjacency and reachability matrix. That is, the graph is represented by the matrix *M= m (i, j)* of adjacency and is proved by the reach ability matrix [5, 11-13]. The element of the matrix m_{ij} determines the number of call operators with index *i*, to the module with index *j.*

In addition to the adjacency matrix (calls), the characteristic vector *Vi* = χ (*xi*) for i-elements is used. For a modular structure graph (Fig. 1) characteristic vector and adjacency matrix have the form:

$$V = \begin{pmatrix} 1\\1\\1\\1\\1\\1\\1\\1 \end{pmatrix} \qquad M = \begin{pmatrix} 0&1&1&1&0&0&0&0\\ 0&0&0&0&0&0&0&0\\ 0&0&0&0&1&1&0&0\\ 0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&1&2\\ 0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0 \end{pmatrix} \tag{2}$$

We analyze adjacency matrices and characteristic vectors for subgraphs and graphs of modular structures corresponding to different types – program, complex, aggregate, etc. For subgraphs (Fig.2) vectors and matrices have the form:

$$V_3^s = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \quad M_3^s = \begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}; \quad V_1^s = \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \end{pmatrix},$$

$$M_1^s = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}; \quad V_5^s = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}, \quad M_5^s = \begin{pmatrix} 0 & 1 & 2 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \tag{3}$$

For the program graph (Fig. 1) the characteristic vector and the matrix of calls coincide with V and M, respectively, and determine the form (2), in which all elements of V are equal to one. In the case of the complex, the characteristic vector and the call matrix have the following form:

$$V^c = \begin{pmatrix} V_1^p \\ V_2^p \\ \dots \\ V_n^p \end{pmatrix}, \quad M^c = \begin{pmatrix} M_1^p & 0 & \dots & 0 \\ 0 & M_2^p & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & M_n^p \end{pmatrix} \tag{4}$$

Here V_i^p and M_i^p *($i = \overline{1, n}$)* denote the characteristic vector and the adjacency matrix for the graph of the i-th program included in the graph of the complex. In the future, the matrix representation is used when performing mathematical operations on software structures.

The relation of the reach ability graph of program structures

Let *G = (X, E)* - a graph of a program of modular structure; x_i, x_j - vertices belonging to *X*. If there is an oriented chain from x_i to x_j in the graph *G*, then the vertex x_j is reachable from the vertex x_i. The following statement is true: if the vertex x_j is reachable from x_l – из x_j, x_l – from x_j , then x_l is reachable from x_l. The proof of this fact is obvious.

Consider a binary relation on the set X that determines the reach ability of one vertex of a graph to another. We introduce the notation $x_i \rightarrow x_j$ - reach ability of the vertex x_j from x_i. The relation is transitive. Denote by *D(x*$_i$*)*) the set of vertices of graph G reachable from x_i..

$$\overline{x_i} = \{x_i\} \cup D(x_i) \tag{5}$$

Then the equality of determines the transitive closure of x_i in relation to the achievability of tops. We prove the following theorems.

Theorem 1. For the selected element of connectivity of the graph of the program structure, any vertex is reachable from the root corresponding to the given vertex of the graph, i.e. the equality (x_1 – root vertex)

$$\overline{x_1} = \{x_1\} \cup D(x_1) = X. \tag{6}$$

Evidence. Suppose the vertex x_i *($x_i \in X$)* is unattainable from x_1. Then $x_i \notin \overline{x_1}$ and the set $X' = X \setminus \overline{x_1}$ - not empty. Since the selected component of the graph is connected, there is a vertex $x_j \in \overline{x_1}$ and a chain *H (x_i, xj),* leading from x_i to x_j. Based on the acyclicity of the graph G, in *X"* there should be a simple chain

$H(x_{i,}.x_j)$, where the vertex x_l does not include arcs (this chain can be empty if X' consists only of x_i). Consider the chain $H(x_l, x_j) = H(x_l, x_i) \cup H(x_i, x_j)$. This means that the module x_i is reachable from vertices x_1 and x_i and both vertices contain no incoming arcs. This contradicts the definition of a graph of a modular structure with a single root vertex.

The theorem is proved.

The results of this theorem are important to substantiate the requirement of the absence of oriented cycles in the graph of the program structure with respect to the notion of reachability. Consider the graph shown in Fig. 3. From this figure it is clear that the graph contains a directed cycle and modules corresponding to vertices x_4, x_5, x_6 will never be executed.

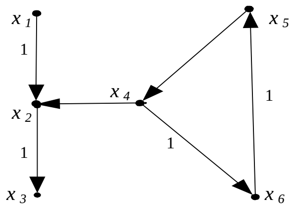


Fig. 3. A graph that contains directed cycle

Thus, the results of theorem1 reinforce the requirement that there are no oriented cycles in the graph of the program structure.

We analyze the matrix representation of the reach ability relation for the graph of the program structure Fig.1 with the reach ability matrix A, which has the form (7). Coefficient $a_{ij} = 1$ if the module corresponding to the index l is reachable from the module corresponding to the index *i* the Following results are based on the following theorem from graph theory.

$$A=\begin{pmatrix} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & x_8 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \qquad (7)$$

Theorem 2. The coefficient m_{ij} of the *l*-th degree of the adjacency matrix M^l determines the number of different routes containing *l* arcs and connecting vertex x_i to the vertex of the x_j –oriented graph. The proof of this theorem is given in [20]. Consider the following three consequences of this theorem. **Corollary 1.1**. Matrix $\overline{M} = \sum_{l=1}^{n} M^i$, where M is the adjacency matrix of a directed graph with n vertices coincides up to the numerical values of the coefficients with the reachability matrix A.

Evidence. In a directed graph containing n vertices, the maximum path length without repeating arcs cannot exceed n. Therefore, the sequence of degrees of the adjacency matrix M^i, where i = 1,2, ..., n determines the number of all possible paths in the graph with the number of arcs ≤ p. Let the coefficient $\overline{m_{ij}}$ of the matrix M be different from zero. This means that there is a degree of matrix M^i in which the corresponding coefficient $\overline{m_{ij}}$ is also nonzero. Therefore, there is a path from vertex x_i to x_j, i.e. vertex x_j is reachable from x_i. This consequence determines the connection of the matrix of calls of the graph of the modular structure *M*, coinciding with the reachability matrix A, and determines the algorithm for constructing the latter.

Corollary 1.2. Let there be a coefficient $m_{ii} > 0$ for some *i* in the sequence of degrees of the adjacency matrix M^i. Then there is a cycle in the original graph.

Evidence. Let $m_{ii} > 0$ for some *l*. Therefore x_l reachable from x_i, i.e. there is a cycle. According to the theorem, this cycle has *l* arcs (generally repeated).

Corollary 1.3. Let the *n*-th degree of the adjacency matrix of the M^n of the acyclic graph coincide with the zero matrix (all coefficients are zero).

Evidence. If the graph is acyclic, then the simplest path cannot have more than $n – 1$ arcs.

If M^n has a coefficient other than zero, then there must be a path consisting of *n* arcs. And this way can only be oriented cycle. Therefore, all coefficients of M^n for an acyclic graph are zero. This consequence provides a necessary and sufficient condition for the absence of cycles in the graph of a modular structure.

For acyclic graphs, the reachability ratio is equivalent to a partially strict order. The transitivity of the reachability ratio was considered above. Anti-symmetry follows from the absence of oriented cycles: if the vertex x_j is reachable from x_j, then the opposite is not true.

We introduce the notation $x_i > x_j$ if vertex x_j is reachable from vertex x_i.

Let G = *(X, E)* be an acyclic graph corresponding to some program structure.

Consider the decreasing chain of elements of a partially ordered set X: $x_{i1} > x_{i2} > ... > x_{in}$ . ...,

where " > ” denotes the reachability ratio.

Since X is finite, the chain breaks. The verte x_{in} has no outgoing arcs, i.e. the element x_{in} is minimal (it corresponds to a module that does not contain access to other modules). The maximum element in the set *X* is the root vertex.

2.3 Mathematical operations on the graph elements

Mathematical operations (U, ∩, /, +, - , *P, C, R*) on graphs are performed at the level of abstractions of elements of program structures that lead to changes in graph elements and characteristic functions of systems: S = (G, χ) [20].

Let $S_1 = (G_1, \chi_1)$ and $S_2 = (G_2, \chi_2)$ be two graphs of program structures $G_1 = (X_1, E_1)$ and $G_2 = (X_2, E_2)$ respectively.

We introduce the following notations:

D (x) – the set of vertices reachable from the vertex x;

$D^*(x)$ – the set of vertices from which vertex x is reachable.

The same symbols are used for the same vertices included in the graphs G_1 and G_2. The main operations on the program structures are discussed below

is intended to form a graph of the structure of the complex and is formally defined as follows S_1 and S_2 – any program structures that satisfy the definitions of claim 1:

Merge (join) operation $S = S_1 \cup S_2$ (9)

$$G = G_1 \oplus G_2, \quad X = X_1 \oplus X_2, \quad E_1 \oplus E_2, \tag{10}$$

where the symbol denotes a direct sum provided:

$$\chi(x) = \chi_1(x), \text{ if } \chi \in X_1,$$

$$\chi(x) = \chi_2(x), \text{ if } \chi \in X_2.$$

The same vertices included in G_1 and G_2 are represented by different objects in the operations of combining program structures. The characteristic vector and adjacency matrix of the program structure S are defined as follows:

$$V_{1,2} = \begin{pmatrix} V_1 \\ V_2 \end{pmatrix}, \; M_{1,2} = \begin{pmatrix} M_1 & 0 \\ 0 & M_2 \end{pmatrix}, \tag{11}$$

where $V_{1,2}$ and $M_{1,2}$ are characteristic vectors and adjacency matrices of modular structures S_1 and S_2 respectively. This operation is associative, but not commutative – the order of the operands determines the order of the components of the complex.

It should be noted that if the operands S_1 and S_2 satisfy the conditions for defining program structures, the result S will also satisfy the same requirements. The join operation increases the number of connected graph elements. In addition, the column structures may themselves have multiple items of connectedness. For the rest of the operation counts of the operands and result are the only element of connection.

The connection operation. We denote by x_i and x_j the root vertices of graphs G_1 and G_2 of program structures S_1 and S_2, respectively. This operation

$$S = S_1 + S_2, \tag{12}$$

which is execute if these structures meet the following conditions:

set $X' = X_2 \cap X_2$ not empty;

vertex $x_j \in X'$ and $\chi(x_j) = 0$;

$D^*(x) \cap D(x) = 0$ for every $x \in X'$, where $D^*(x) \in X_1$ и $D(x) \in X_2$;

$$G = G_1 \cup G_2, \quad X = X_1 \cup X_2, \quad E = E_1 \cup E_2, \tag{13}$$

The characteristic function χ is satisfied under the condition:

$\chi(x) = \chi_1(x)$, *if* $x \in X_1 \setminus X'$;

$X(x) = max(\chi_1(x), \chi_2(x))>$ if $x \in X'$,

$\chi(x) = \chi_2(x)$, *if* $x \in X_2 \setminus X'$.

First condition means that there are common vertices in graphs G_1 and G_2. According to the second condition, the root vertex G_2 belongs to the common part and for S_1 the object corresponding to x_j is not included in the program structure yet.

The third condition prohibits the existence of cycles in the result graph. Indeed, if there is $x_n \in D^*(x) \cap D(x)$,then $x_n > x$ and $x > x_n$, and $x > x_n$, then this means the existence of a cycle.

If S_1 and S_2 satisfy the above conditions, the connection operation is partial.

Let us determine whether the result of the connection operation belongs to the class of program structures. Since X'' is not empty, the graph G has one connected component. The root vertex of the graph G is x_i. The graph G itself has no oriented cycles, i.e. is acyclic.

Thus, S belongs to the class of program structures under consideration.

This connection operation is not commutative and is generally not associative. To show that this operation is not associative, consider the result $S = (S_1 + S_2) + S_3$, where the root vertices of graphs G_2 and G_3 are part of the vertices of graph G_1 and $X_2 \cap X_3 \neq 0$.

Then the result of the $S_2 + S_3$ join operation is undefined.

The operation of projection. Let $S_1 = (G_1, \chi_1)$ be a program structure and $x_i \in X_1$. The operation of projection of this structure to the top of the graph S_1 is denoted as $S = P_{rxi}(S_1)$ and is defined as

$$G(X, E), \quad X = \overline{x}_i , \quad E = \{(x_i, x_j, K) \mid x_i, x_j \in X\}, \tag{14}$$

for the characteristic function is $\chi(x) = \chi_1(x)$, if $x \in X$. The projection operation defines the program structure S_1 in the structure S. let's check the belonging of the structure S to the class of the considered program structures. If the graph of the structure S_1 is connected acyclically, then the same properties will be possessed by the graph G. There is a single root vertex x_i in the graph G. Thus, the program structure S belongs to the class under consideration.

The difference operation for program structures is defined as follows. Let $S_1 = (G_1, \chi_1)$ be a program structure and $x_i \in X_1$. The difference operation is performed on this structure and its projection to the vertex x_i of the corresponding graph (x_i is not the cortical vertex of the graph G_1). Formally, the difference operation of the program structure has the form:

$$S = S_1 - P_{r_{xi}}(S_1), \tag{15}$$

and defined as follows

$$G = \{X, E), \quad X = (X_1 \setminus \overline{x_i}\,) \cup X' \tag{16}$$

$$\Gamma = \{(x_i, x_j, K) \mid x_i, x_j \in X\},$$

where the set X' consists of such elements for which

$$X' = \{x'_j \mid (x_l \in X_1 \setminus x_i) \,\&\, (x'_j \in \overline{x_i}\,) \,\&\, (x_l, x'_j, K) \in E\} \tag{17}$$

Here, the characteristic function χ is defined as:

$\chi(x) = \chi_1(x)$, если $x \in X_1 \setminus \overline{x_i}$;

$\chi(x) = 0)$, если $x \in X'$.

The set *X* includes vertices that are not included in the set $\overline{x_i}$, and those vertices $\overline{x_i}$ that include arcs from vertex $X_1 \setminus \overline{x_i}$ (sets *X'*). The characteristic function for elements $x' \in X'$ is zero. The difference operation is the inverse of the join operation, i.e. the equality is performed:

$$S - P_{r_{xi}}(S) + P_{r_{xi}}(S) = S. \tag{18}$$

Let us check that *S*, defined in (15), belongs to the class of program structures. If the graph is *G*, connected and acyclic, then the graph G_1 will have the same properties. The root vertex *G* is the same as the root vertex G_1. Thus, *S* satisfies the conditions for determining the program structure given in paragraph 1.

Let S* be the set of program structures given by the direct product G* x χ*, where *G** and *χ** are the set of graphs and the set of characteristic functions. Denote by Ω = {U, ∩, /, +, -} - set of mathematical operations on program structures and *P, C* and *R* - predicates of:

$$\Omega = \{U, \cap, /, +, -, P, C, R\}. \tag{19}$$

Thus, an algebraic system $\Sigma = (S, \Omega)$ over a set of program structures and operations on them (union, connection, differences and projections) is defined.

2.4 Characteristics of simple and complex graph structures

Among the variety of program structures there are three main ones – a simple, complex structure with a call of modules from the external environment and a dynamic structure. The main purpose of various structures is the most optimal use of the main memory during the execution of the unit [15-20].

Simple structure. An aggregate with a simple structure is created in the process of building modules based on the operations of link calls. The amount of main memory occupied by an aggregate with a simple structure is constant and equal to the sum of the volumes of individual modules: $V_s = \sum_{i=1}^{n} v_i$, where v_i is the amount of memory occupied by the *i*-th module. The corresponding graph of a modular structure is always connected.

Complex structure. Assembly of complex structures with dynamic invocation of modules in the shared memory is created in the Assembly process of the modules. In such an aggregate, the connections between the modules are not so rigid and their sequence is determined by the modules included in the chain. The modules are loaded into the main memory at the time of processing. When finished, the memory is freed and used to load another module. As in the case of a simple structure, the graph of a complex program structure is also connected (Fig.4) and is reflected in the adjacency matrix (2).

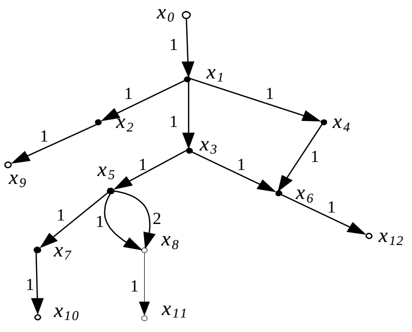


Fig.4. Modification graph of program structure

The amount of main memory required depends on the number and composition of modules and the maximum amount of memory is equal to the sum of individual modules:

$$V_0^{\max}=V_s=\sum_{i=1}^{n} v_i .$$

The minimum amount of memory required when performing the aggregate is calculated by Floyd's algorithm, which determines the shortest path in the graph, in which each arc corresponds to a weight coefficient, called the arc length. The following transformations are performed to apply the Floyd algorithm.

1). Let's add new vertices and arcs to the graph. The vertices are $x_0, x_{n+1}, \dots , x_{n+m},$,

where *m* is the number of end vertices. New arcs include $(x_0, x_1, 1), (x_{r1}, x_{n+1}, 1), \dots, (x_{rn}, x_{n+rn}, 1)$. In them x_1 corresponds to the main module and all x_i – to the end vertices. After performing operations, the graph of the modular structure (Fig. 1) is given to the graph on Fig. 5 with vertices $x_0, x_9, x_{10}, x_{11}, x_{12}$. It vertices correspond to the weight coefficients:

$$v_0 = v_9 = v_{10} = v_{11} = v_{12} = 0$$

2). Each arc of the form (x_i, x_j, k) is assigned a coefficient $v_{ij} = \frac{v_i + v_j}{2}$.

Consider all routes leading from x_0 to one of the other additional vertices. The length of the shortest route path is calculated as follows:

$$l_{0,n+p}=v_{01}+ \dots +v_{rp,n+p} = \frac{v_0 + v_1}{2} + \dots + \frac{v_{2p} + v_{n+p}}{2} = \frac{v_0}{2} + v_1 + \dots + v_{rp} + \frac{v_{n+p}}{2} = v_{1+ \dots +} v_{rp}.$$

This length $l_{0,\, n+p}$ will be equal to the sum of the memory modules for path $x_1, \dots , x_{rp}$.

Thus, applying Floyd's algorithm to the graph in Fig. 2, we solve the problem of calculating the amount of memory for the maximum chain.

3). We replace the adjacency matrix with the path matrix. For each $m_{ij}>0$, the corresponding location will be v_{ij}. The values $m_{ij} = \emptyset$ are replaced by $-\infty$. The program implementing Floyd's algorithm has the following form (it is assumed that the path matrix is described as a two-dimensional matrix $(n \times n)$: this length $l_{0,\, n+p}$ will be equal to the sum of the memory modules for path $x_1, \ldots, x_{rp}$.

```
for $k$ = 1 to $n$ do
for $i$ = 1 to $n$ do
for $j$ = 1  to $n$ do
if M[i, j ] < M [ i , k] + M[k, j] then
M [i, j]: = M [ i , k] + M[k, j].
```

As a result of this algorithm, a matrix of maximum paths will be constructed. The maximum of $l_{0,n+p}$ will determine the minimum amount of $l_{0,n+p}$ memory for the memory-overlapping aggregate.

The most complex structure for the values $V_0^{min} \le V_0 \le V_0^{max}$ can be constructed by following the algorithms proposed in [2-6]. The qualitative dependence of V_0 on the number of dynamic sites is shown in Fig.5. Here n is the number of modules in the unit. Despite the different kind of curves, they have a common pattern – any V_0 is enclosed between the values of v_0^{max} и v_0^{min}.

Dynamic structure. The mechanism of dynamic links between modules is different from the call mechanism. Dynamic objects are loaded into the main memory when they are accessed. By analogy, we call the volume loaded with a single treatment of a dynamic element, has its own program structure, for which the adjacency matrix is composed. If the same modules are found in different dynamic structures, they are different objects.

The original graph is used for illustration (Fig.1). Let the module corresponding to the vertex x_1, be dynamically called from the module corresponding to the vertex x_3. The resulting modified graph is shown in Fig. 6. A dashed arrow indicates a dynamic call. The module corresponding to the vertex x_6, occurs twice.

We construct an adjacency matrix for this aggregate. Each dynamic element will have its own *CALL* . To distinguish a dynamic call, the corresponding matrix elements will contain negative numbers whose absolute values specify the number of dynamic calls between the data of the module pair [20].

The adjacency matrix will look like:

We investigate the qualitative dependence of the amount of the number of dynamic segments (Fig.5. and 6). With one component in the software unit of a simple structure we have $V^1_d = V_S$. If each dynamic component consists of one module, then the modified Floyd algorithm finds the maximum path and $V_d^n = V_0^{min}$.

$$M=\begin{pmatrix} x_1 & x_2 & x_4 & x_6 & x_3 & x_5 & x_6 & x_7 & x_8 \\ 0 & 1 & 1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \qquad (20)$$

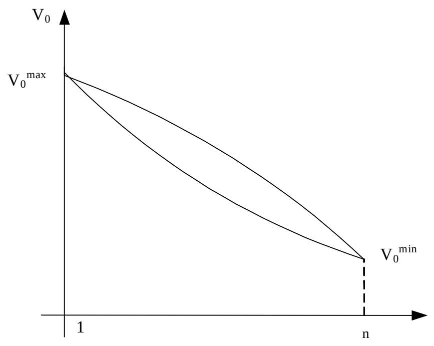


Fig. 5. Graph of qualitative dependence Va from the number of sub graphs

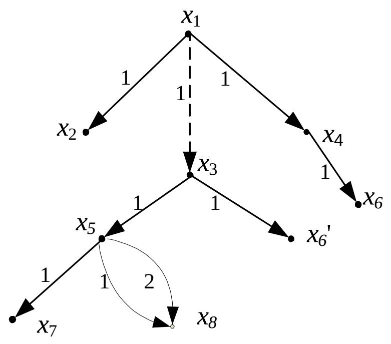


Fig. 6. Graph programs structure with dynamic Calls

For intermediate values, the dependence is more complex. On fig.7 presents two curves (1, 2), and n is the number of modules in the program unit.

Curve 1 defines a relationship in which different segments do not have the same modules. Curve 2 describes the dependence for the case when different segments have the same modules. For them, the required memory increases due to the duplication of such modules. However, dependence 2 is typical for the case when there are no identical modules in dynamic structures and they are written in high-level *LP*. These modules are handled by utility tools – memory management, I/O, emergency handling, etc.

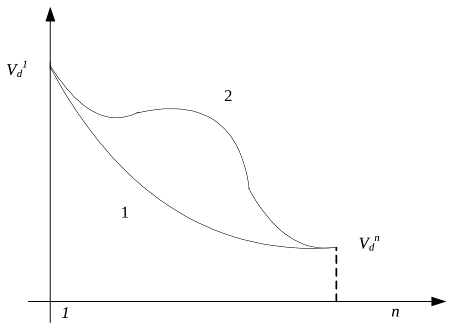


Fig. 7. Grafic dependence *V_a* from the number of dynamic elements

Due to the duplication of modules there is an increase in the main memory of the *OS*. Thus, curve 1 is characteristic of software aggregates of graphs in the form of a tree, which ensures that there are no identical modules in the graph. Despite the lack of dynamic structure in terms of memory savings, there is a significant advantage – independence from editing links. Each dynamic object can be modified, and editing relationships in the *OS* is not required.

3 Operations of Assembling Elements of Graph G

Let the graph G be represented by the set of modules X= {$x_1, x_2, ..., x_m$}, as described in *LP*, and located at the vertices of the graph. The modules are assembled into a software unit. In this case, each pair of modules x_i, x_j *(i, j* – languages from the set of *LP* are connected by the relation of call on the basis of which the module of communication x'_{ij} is formed. In General, for simple program structures, the aggregate contains link communication (call) operators and forward and reverse transformations of data types passed from the calling module (in *i*-language) to the calling module (in *j*-language) and back [23].

LP allows you to describe the information part - passport modules with a description of the transmitted data [8-14] and operations call modules. Taking into account the passports of the modules, the software structure of the unit is built (program - *Prog*, complex - *Comp*, package - *Pac*). The passport describes the special language WSDL containing: - a subset of the operations associate link elements of the graph in the language *L'* that contains a description of the parameters from the list of actual and formal parameters of the invocation; - mathematical operations on the graph and operations of binding modules in a complex structure (*Prog*, *Comp*, *Agr*, *Pac* and so on).

The operator modules link (make, config, assembling, etc. since 1994) takes the form:

Link <aggregate type> <aggregate name> (<main module name>, <additional list of module names>) <execution mode>,

when constructing specific program structures, the vertices of the graph – modules can be marked with special symbols ρ, denoting:

ρ = ¤ – formation of a fragment with the name of the module;

ρ = * – the beginning of the dynamic fragment with the vertex marked by this symbol;

ρ = + the module in the graph G is marked as the main program of the complex;

ρ = / – means enabling debugging or testing of the unit.

Using these designations, the graph G will take the form shown in figure 8 and has a representation: $E = \{(x_5, x_7, 1), (x_5, x_8, 1), (x_5, x_8, 2)\}$.

The aggregate is given a unique name corresponding to the generated root module. For the graph $\Gamma = \{(x_4, x_6, 1)\}$ a fragment of operators providing a dynamic call will be formed in the communication module x'_{46}. For a pair of modules specified in Fig.8 vertices x_4, x_6, the structure of the corresponding part of the unit, including the communication module, is shown in Fig. 9. Similarly links of heterogeneous modules and other types of calls are implemented.

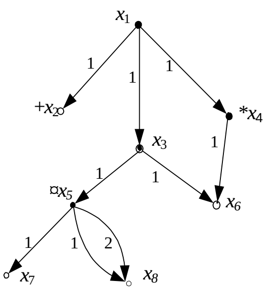


Fig. 8. Graph of software unit with control marks on the graph

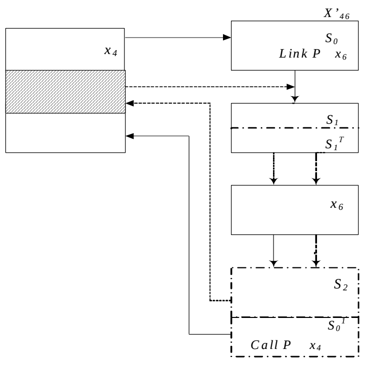


Fig. 9. Graph of modular structure with dynamic call [1]

Thus, for a pair of modules x_i, x_j, a module of connection x_{ij} of the form:

$$x'_{ij} = S_0 * (S_1 \times S_1^T) * (S_2 \times S_2^T) * S_0^1,$$

where S_0 is a fragment of the aggregate that defines the environment of xj module functioning;

S_1 – a fragment of the aggregate, including a sequence of calls to functions from the set {*P, C, S*}, each of which performs the necessary conversion of the actual parameters when referring to the x_j -module;

S2 – a system with a fragment of operators for the inverse transformation of data types transmitted from x_j to x_i after its execution;

S_0^1 – a piece of software structures with operators epilogue for the vertex x_i, for the restoration of the environment.

For the described program structures, set the link operations to build the individual programs in Fig.8:

Link Prog $P_1 (x_1, x_2)$;

Link Prog P_2 (x_1, x_3) (x_3, x_6);

Link Comp P_3 $((x_1, x_3)$ $(x_3, x_5/ x'_{58})+ (x_5, x_7)$; (20)

Link Prog P_4 $(x_1, x_4), (x_4, x_6)$;

Link Comp $(P_1 \cup P_2 \cup P_3 \cup P_4)$.

Programs of the complex (aggregate) are given unique names (P_1, P_2, P_3, P_4) corresponding to the root names of the modules in the chains of the graph.

Thus, the process of constructing the program structure on the graph includes:

1. Enter the module description in the *LP* (*L'*) and perform syntax checking.
2. Select the required modules and interfaces from the repositories and place them in the graph.
3. Translation of the unit modules in the *LP*.
4. Generation of communication modules for each interconnected pair of graph modules.
5. Assembly of the elements of the graph in the finished structure, linking modules in the operating system (IBM, MS, Oberon, Unix и др.) [1-5]**.**
6. Test the system on data sets and assess the reliability of the unit**.**

After the modules are built, the name of the software Assembly is entered into the boot library. If you create a fragment that is later included in another aggregate, its name must match the name of the main module. In connection with the transition to the Internet environment to work with various software and system services in the configuration assembly of such tools provides security, data protection and quality assessment of ready-made modules, service resources and web systems in Internet.

Ready-made software elements configurate to the system

Under *the configuration of the system* is understood the structure of some of its version, including software elements, combined with each other by link operations with parameters that specify the options for the functioning of the system [1, 2, 16-22]. Version or variant of system configuration according to the IEEE Standard 828-2012 (Configuration) includes:

– configuration basis – BC (Configuration Baseline);

– configuration items (Configuration Item);

– program elements (modules, components, services, etc.) included in the graph;

Configuration Management is to monitor the modification of configuration parameters and components of the system, as well as to conduct system monitoring, accounting and auditing of the system, maintaining the integrity and its performance. According to the standard, the configuration includes the following tasks:

1. Configuration identification (Configuration Identification).
2. Configuration Control (configuration Control).
3. Configuration Status Accounting (Configuration Status Accounting).
4. Configuration audit (Configuration Audit).
5. Trace configuration changes during system maintenance and operation;
6. Verification of the configuration components and testing of the system.

A configuration build uses a system model and a set of out-of-the-box components that accumulate in the operating environment repositories or libraries, and selects their operating environment Configurator (for example, in http://7dragons.ru/ru). The Configurator assembles the components according to their interfaces and generates a system configuration file***.***

The Configurator assembly of components and reuses with operation config, which is equivalent to the operations link (20) for figure 8, taking into account their interfaces. The *config* statement generates a program variant or system configuration file of Comp.

4 The Modern and Future Technologies Programming

The theory of system programming is represented by numerous paradigms - mathematical, object-component, ontological, technical, service, aspect, etc. They replace on the *LP* and realized by different paradigms [15-17, 23-25], some of this paradigms are presented below.

4.1 Mathematical programming paradigms

Theory of graphs for design software modular structures with mathematical operations (union, projection, difference, etc.) implementation of linking the graph modules (objects) and the semantics of the transformation of data transmitted by the vertices of the graph *G*.

Object-component methods - OCM

OCM are the mathematical design of systems from ready-made resources (objects, components, services, etc.) to OM (Object Model). It is the formal method which transform the elements *OM* to a component model or a service model [15, 26].

Graph objects is designed on four levels:

Generalizing for determining *SD* base notions without considering of their essences and properties;

Structuring for ordering objects in the *OM* taking into account relationships between them;

Characterization for forming concepts of objects on the base of them properties and descriptions;

Behavioral level for descriptions of conduct depending on events (such as time).

That is, vertices of the graph *G* are objects of two types: *O= (O_0, O_1, O_2. O_n)* with *the* object relations hold $\forall i \ (i > 0) \Rightarrow (O_i \in O_0)$ and interface objects *I* (Fig.10).

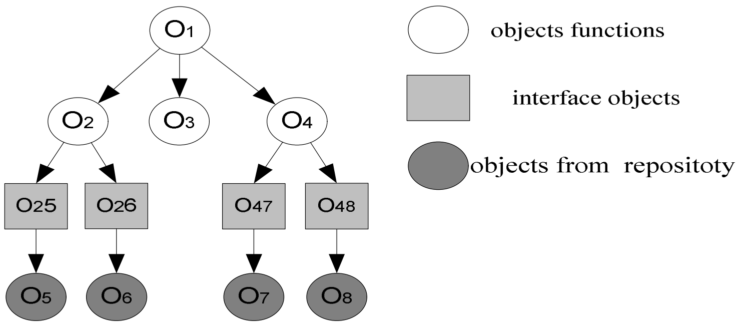


Figure 10. Object-interface graph *G*

At the vertices of a graph *G* contains the functional objects *O_1, O_2, O_3, O_4, O_5, O_6, O_7, O_8* and interface objects —*O'_{25}, O'_{26}, O'_{47}, O'_{48}*, which are placed in the repository of system, and arcs correspond to relationships between all kinds of objects. The parameters of the external characteristics of the interface objects are passed between objects through specified interfaces and are designated in language IDL *in* (input interface), *out* (output) and *inout* (intermediate). Based on the graph *G* we can construct a program P_0— P_5 using mathematical operation $\cup$ Assembly link:

1) $P_0 = (P_1 \cup P_2 \cup P_3 \cup P_4 \cup P_5)$.
2) $P_1 = O_2 \cup O_5$, *link* P_1 =*In* O'_5 $(O_2 \cup O_5)$;
3) $P_2 = O_2 \cup O_6$, *link* P_2 =*In* O'_6 $(O_2 \cup O_8)$;
4) P_3;
5) $P_4 = O_4 \cup O_7$, *link* P_4 =*In* O'_7 $(O_4 \cup O_7)$;
6) $P_5 = O_4 \cup O_8$, *link* P_4 =*In* O'_8 $(O_4 \cup O_8)$;

The set of objects and interfaces of the graph is reflected by general or individual properties and descriptions of the object model. Verification of properties of objects is provided by the specific operations (classification, specialization, aggregation, etc.) [15, 20, 29].

Component paradigm. The basis of this paradigm - OCM graph in which vertexes are the components of the CRP (reuses), interfaces and arcs specify the subject classification and the relationship between the vertices. Components are described by the formalisms of the triangle of Frege [15]:

- sign – identifier of the real function entity;

- denotation – the designation of this entity;

- concept – a set of properties defined by logical connections and must be true.

Operations of OCM and component algebra represented on the website http://7dragons/ru/ru (in the VS environment.MS IBMSphere, Java, Linux, Intel etc.) [20].

Service-component paradigm. System and service-components - web resources implement intellectual knowledge of specialists about applied fields in the Internet environment [22 - 26]. Each implements some function and communicates with the technological interface to interact with other services through protocols and provide Assembly and solution of applications of different nature. The means of describing the application systems include:

XML for description and construction of SSA components;

WSDL to describe web services and their interfaces;

SOAP to determine the formats of requests to the web services;

UDDI for integration of services and their storage in libraries;

building configuration (config) of the service resources in some high-quality and secure systems.

The theory of graphs develop in the school of A.P. Ershov (V.I. Kasyanov, V.E. Itkin, A. A. Evstigneev et al.) for programming Systems [13]. The graph theory has been actively developing in the Russian Academy of Sciences (I.B.Burdonov, A.S. Kosachev, V. V. Kulyamin [19]. The theory of conformity for systems with blocking and destruction for the schematic organization of memory in Linux.

Methods of production of factories (Product Line/Product Family) programs and Appfab and certificate them of the quality are discussed [23].

Application of the ontology language OWL (www.semantic_web.com), resource language (RDF) and intelligent agents of ISO 15926 standard for networking.

Ontology of Life Cycle and *Computational geometry* is a part of computer graphics and algebra. Used in the practice of computing and control machines, numerical control etc. is also used in robotics (motion planning and pattern recognition tasks), geographic information systems (geometric search, route planning), design chips, etc.[25-30].

Cloud technologies (PaaS, SaaS) are related to the Internet and are used to create adaptive applications that interact through agents of web pages.

Device configuring *Big Data* Processing Devices (Big Data) in Smart Data Internet 4.0.

4.2 Intellectualization of systems

The intelligent system implements creative tasks, the knowledge of which is stored in its memory. It includes — knowledge base, output mechanism and intelligent interface. The main tasks of artificial intelligence: symbolic modeling of thought processes, work with natural languages, presentation and use of knowledge, - machine learning, biological modeling of artificial intelligence, robotics [30].

4.3 Application technical programming

Event management paradigm based on the processing of external events (event-driven programming) in the Window environment. Features of the event paradigm are the use of testing methods based on operational (scenario) profiles of programs [20, 25, and 28].

Coordinated and parallel programming provides a division of the computational process into several subtasks (processes) for TRAN's computers and supercomputers, the results of which are sent via communication channels. Languages for parallel programming - PVM, LAM. CHMP and MPI (Message Passing Interface) interface descriptions and OpenMP. The POSIX standard provides messaging between programs in YAP C, C+ and Fortran.

Programming on classes and on a prototype in OOP. The principles of the OOP are:

inheritance – the mechanism of establishing relations "descendant-ancestor" (the ability to generate one class from another with the preservation of all the properties and methods of the class-ancestor); encapsulation (the hiding of class implementation); abstraction (description of interaction only in terms of messages/events in the subject area); polymorphism (the possibility of replacing the interaction of objects of one object with another object with a similar structure). Many modern languages are specially created for programming on classes, for example, Smalltalk, C++, Java, Python, PHP, Object Pascal (Delphi), VB.NET, Xbase++, etc.

Programming by prototype. Creating a new object is done by one of two methods: cloning an existing object, or by creating an object from scratch. Reuse (inheritance) is made by cloning an existing instance of the object —a prototype Clone, a sample. An example of a prototype language is the Self language and it is the basis of such programming languages as **JavaScript**, **Squeak**, **Cecil**, **Newton Script**, **Io**, **MOO**, **REBOL**, **Keno and etc.**

The Agile methodology is focused on the close collaboration of a team of developers and users. It is based on a waterfall model lifecycle incremental and rapid response to changing demands on PP. The team works according to the schedule and financing of the project.

eXtreme Programming *(XP)* implements the principle of "collective code ownership". It any member of the group can change not only your code but also code another programmer. Each module is supplied with the Autonomous test (unit test) for regression testing of modules. Tests written by the programmers and they have the right to write tests for any module. Thus, most of the errors are corrected at the stage of encoding, or when you view the code, or by dynamic testing.

SCRUM is agile methodology project management firm Advanced Development Methods, Inc., used in organizations (Fuji-Xerox, Canon, Honda, NEC, Epson, Brother, 3M, Xerox and Hewlett - Packard etc.) are based on an iterative lifecycle model with well-defined development process, including requirements analysis, design, programming, testing (http://agile.csc.ncsu.edu).

DSDM (Dynamic Systems Development Method) for rapid development of RAD (Rapid Application.

4.4 Perspective directions for the development of the Internet

Promising areas of development for the Internet1 include [28]:

The information objects (IO) that specifies the digital projection of real or abstract objects that use Semantic Web Ontology interoperability interfaces. IO through Web services began more than 10 years ago. Interaction semantics IO is based on RDF and OWL language of ISO 15926 Internet 3.0.

The next step of the development of the Internet is Web 4.0, which allows network participants to communicate, using intelligent agents. A new stage in the development of enterprise solutions-cloud (PaaS, SaaS) who spliced with Internet space and used to create Adaptive applications. Cloud services interact through the Web page by using agents.

Internet of Things Smart IoT to support competitive APPS using: distributed microservices; Hypercat Mobile; GSM-R traffic control. Industrial Internet develops concepts - “smart energy”, “smart transportation”, “smart appliances”, “smart industry”, “smart homes and cities”, etc.

Internet stuff (Internet of Things, Smart IoT) indicates the Smart support competing APPS using distributed micro services such as Hyper cat (mobile communications); industrial Internet (Industrial), covering the new automation concepts-smart energy, transportation, appliances, industry», and another.

4.5 Computer nanotechnology

Today computer nanotechnology is actually already working with the smallest elements, "atoms" similar to the thickness of the thread (transistors, chips, crystals, etc.). For example, a video card from 3.5 million particles on single crystal, multi-touch maps for retinal embedded in the eyeglasses, etc.

In the future, ready-made software elements will be developed in the direction of nanotechnology by "reducing" to look even smaller particles with predetermined functionality. Automation of communication, synthesis of such particles will give a new small element, which will be used like a chip in a small device for use in medicine, genetics, physics, etc.[28].

5 Conclusion

In the early stages of the emergence of the method of assembling large programs and complexes of spent modules in the LP used the theoretical apparatus of graphs to create modular program structures. Graph theory allows us to establish the shortest path of program elements and prove the correctness of binding graph modules using adjacency matrices, reach ability and mathematical operations (association, connection, difference, etc.) in complex program structures (complex, aggregate, system, etc.). Initially, the method of Assembly on the basis of graph theory was widely implemented in the Ruza systems, Prometheus Complex under the leadership of Lipaev V.V. [1-5], and was supported by A. P. Ershov in the IPI SO Academy of Sciences SSSR and his researcher and scientist, who formulated the theoretical aspects of the application of graph theory in programming [6-14]. Since 2013, graph theory has been used in the modeling of complex systems of objects, components, services, etc. (OCM) [15] and has been used in the world practice in the transition to the Internet environment [22, 23]. The paper describes the features of modeling systems using graph theory and mathematical operations on elements of software structures. The new structures Assembly operations – *config* of the IEEE Standard 828-2012 (Configuration) are implemented in different environments of Internet. Elements of the graph set transition labels to obtain reactions at the time of exposure to test sets and proof of completeness of testing systems of AS.

The graph theory, programming paradigms and ontology of mathematical modeling of applied problems for vital areas of society (medicine, biology, physics, mathematics, economics, etc.) will become the main tools of smart machines and AS of the 21st century [15, 24-31].

REFERENCE

[1] Lavrischeva E. M. , Grishchenko V. N. The connection of multi-language modules in the OS of the ES.- Moscow, 1982.-127p.

[2] Glushkov V. M., Stogniy A. A., Lavrischeva E. M. and others. System of automation of production of programs (The APROP).-Kiev, 1976.-134p.

[3] Lavrischeva E. M., Grishchenko V. N. Assembly programming. –K.: Of Sciences. Dumka.1991.-136p.

[4] Lipaev V. V., Posin B. A. ,Shtrik A. A. the Technology of Assembly programming.-M.: 1992.-284 p.

[5] Lavrishcheva E. M. , Grishchenko V. N. Assembly programming Basics of software industry products'. K.: Of Sciences.Dumka.-2009.-371p.

[6] Rimsky G. V. Structure and functioning of the modular automation system programming.- Artificial intelligence: application in chemistry.-1987.-№5.-p. 36-44.

[7] Halstead M. H. The beginnings of a science about the programs.- Perevod. with ang. –M.: Finance and statistics.-1981.-201p.

[8] Horn, E., Winkler, F., Design of modular structures.– Computer technology of the socialist countries.- 1987.- Issue .21.-p. 64-72.

[9] Koval G. I., Korotun T. M., Lavrishcheva E. M. On one approach to solving the problem of intermodule and technological interface// All. the collection of the Academy of Sciences and Min.University of the USSR.-1987.-p.52-68.

[10] Agafonov V. N. Program specification: conceptual tools and their organization.- Novosibirsk.- Science, 1987.-380p.

[11] Kotov V. E., Introduction to the theory of program schemes, Novosibirsk, 1978.

[12] Nepeyvoda N. N. Program logic.- Programming, 1979, № 1, p. 15-25.

[13] Evstigneev A. N. Graph theory in programming, Moscow, Nauka.- 1985. -351p.

[14] Ershov A. P., Introduction to the theory of programming.-Moscow.-1977. - 287p.

[15] Lavrishcheva E. M. The theory of object-component modeling of software systems. Preprint the Russian Academy of Sciences, No. 29, 2016 - M: 48 p. ISBN 078-5-91474-025-9.13.

[16] Lavrischeva E. M. Ryzhov A. G. Application the theory of General data types of ISO/IEC 11404 GDT standard in relation to Big Data.- The conference “Actual problems in science and ways their development”, 27 December 2016, http://euroasia-science.ru.- p. 99-110.

[17] Lavrischeva E. M., Mytulyn V. S., Kozin S. V., Ryzhov A. G. creation of the application and information Systems from ready-made Internet resources. The proceedings of ISP RAS.-M.: Volume 30. Issue.1 .p.27- 40.

[18] Lavrischeva E. M. , A. G. Ryzhov. Approach to modeling systems and sites from ready-made resources.- .XX All-Russian conference , September 17-22, 2018. Novorossiysk.-IPM im. M. V. Keldysh.- Report presentation. Publication in the collection.-p. 321-345.

[19] Burdonov I. B., Kosachev A. S., Kulyamin V. V. Theory for systems with locks and destructions.-Moscow, 2008.- 411p.

[20] Lavrischeva E. M. Software Engineering of computer systems. Paradigms, technologies, CASE- means – Science Dumka.- 2014.-284p.

[21] Bruno Courcelle, Joost Engelfriet Graph structure and monadic second-order logic. A language-theoretical approach (hal id: hal-oo646514) and Theory graph (wikipedia.ru, Foxford.ru).

[22] Lavrischeva E. M., Pakulin N.V., Ryjov A.G., Zelenov S. V. Analysis of methods of assessment reliability of equipment and systems. Practice of application of methods of reliability.-Scientific- practical conference - OS DAY, Moscow, 17-18th 2018. The proceedings of ISP RAS, том5 DOI: 10.15514/ISPRAS-2018-30(3), 2018.- .(http://0x1.tv/20180517F).

[23] Ekaterina M.Lavrischeva. Assemblling Paradigms of Programming in Software Engineering.- 2016, 9,p.296-317, http://www.scrip.org/journal/jsea, http://dx.do.org/10.4236/jsea.96021

[24] Lavrischeva E. M. The Scientific basis of software engineering.- International Journal of Applied And Natural Sciences (IJANS). ISSN(P): 2319-4014; ISSN(E): 2319-4022 Vol. 7, Issue 5, Aug Sep. 2018; p. 15-32.

[25] Gorodnyaya L. V. Programming Paradigms. Analysis of the state and prospects.-SORA9N, 2018.-282p.

[26] Ekaterina Lavrischeva, Andrey Stenyashin, Andrii Kolesnyk. Object-Component Development of Application and Systems. Theory and Practice. Journal of Software Engineering and Applications, 2014, http://www.scirp.org/journal/jsea.

[27] Lavrischeva Ekaterina. Ontological Approach to the Formal Specification of the Standard Life Cycle, "Science and Information Conference-2015", Jule 28-30, London, UK, www.conference.thesai.org.- p.965-972.

[28] Lavrishcheva E.M. Petrov I.B. Ways of Development of Computer Technologies to Perspective Nano.- Future Technologies Conference (FTC), 29-30 November 2017| Vancouver, Canada-p.540-549.

[29] Lavrischeva E.M. Development of the theory programs and systems in the USSR. History and modern Theory. - Sorucom-2017, IEEE Springer-2017.-p. 31-47.

[30] Lavrischeva E.M.. Scientific Basis of System Programming.- Journal of Software Engineering and Applications (JSEA), Vol. 11 No. 8 of August issue, 2018.-N 11.-p.408-434, ISSN online 1945-3124, ISSN Print 1945-3116. http://www.scirp.org/journal/jsea

[31] E.M. Lavrischeva, A.K..Petrenko. Informatics -70. Computerization aspects of programming software and informatic systems technologies.- ISP RAN/Proc. ISPRAS, 2018.- P.7-23.

SOCIETY FOR SCIENCE AND EDUCATION
UNITED KINGDOM

TMLAI TRANSACTIONS ON
MACHINE LEARNING AND ARTIFICIAL INTELLIGENCE

Volume 7 No 4
ISSN 2054-7390

I-AFYA: Intelligent System for the Management of Diabetes in Kenya

Nguku N. Joshua Elisha T.O. Opiyo
University of Nairobi, KENYA.
joshuanguku@gmail.com opiyo@uonbi.ac.ke

ABSTRACT

Computational Intelligence approaches have gained increasing popularity given their ability to cope with large amounts of clinical data and uncertain information. The treatment offered for diabetes aims to keep a patients' blood glucose level as normal as possible and to prevent health complications developing later in their life. Researchers and developers have created diabetes applications and systems that already are frequent on various application stores and shelves. Applications running on artificial intelligence (AI) and cognitive computing models offer promise in diabetes care. This is given the fact that diabetes is a global pandemic. An estimated 425 million people worldwide have diabetes, accounting for 12% of the world's health expenditures and yet one in two persons remain undiagnosed and untreated. Type 2 diabetes is driven by the global obesity epidemic and a sedentary lifestyle that overwhelms the body's internal glucose control requiring exogenous insulin. In Kenya alone, diabetes is a leading cause of kidney failure, lower limb amputations and adult-onset blindness. Thus, research on diabetes care using technological (ICT) solutions will continue to dominate the discussion for quite some time. The early detection of diabetes is of paramount importance. Generally, a physician diagnoses diabetes by evaluating the current test results of a patient or by comparing the patient with other patients who have the same condition. The early detection and screening for individuals with impaired glucose tolerance can help lower risk of developing diabetes and reduce the long-term burden to individuals and health services. For this reason, artificial intelligent systems for diagnosing diabetes have been an item for research for some time. The use of intelligent systems in the Kenyan health care system can help lower the cost of diabetes treatment besides increasing the access and quality of health care provided to diabetic patients.

Keywords: Diabetes mellitus, Artificial Intelligence, Decision tree, Conceptual Design, KNN, Agile, I-Afya, Regression Analysis, Diabetes Kenya, Support Vector Machine, Reinforcement Learning, and Knowledge Discovery in Databases.

1 Introduction

The World Health Organization estimates that 80 percent of the responsibility for chronic disease management rests with patients. The patients need to follow daily care routines, make life style changes and improve communication with caregivers (Sobel, 2003; NHS Modernization Agency, 2004). Computational Intelligent systems offer a unique opportunity to empower individual patients in improving their compliance with care management and improving health outcomes and reducing the cost of

DOI: 10.14738/tmlai.74.6882
Publication Date: 24th August, 2019
URL: http://dx.doi.org/10.14738/tmlai.74.6882

treatment. By use of an Intelligent and interactive decision-making system, the application of diabetic health care can be improved at key touch points.

Application of artificial intelligence (AI) and cognitive computing models have increased efficiency in the detection and treatment of diabetes. This is a useful issue for development and research given that diabetes is a global pandemic. Type 2 diabetes is driven by the global obesity epidemic and a sedentary lifestyle that overwhelms the body's internal glucose control requiring exogenous insulin. Optimal care for persons with diabetes often is hampered by the absence of real-time, key health data necessary to make informed choices associated with intensive therapy and tight diabetes control.

Artificial Intelligence offers the promise of making both real-time structured and unstructured health data available for the care of diabetic patients. The Turing Archive for the History of Computing defines AI as "the science of making computers do things that require intelligence when done by humans." AI covers a broad range of approaches to simulating human intelligence and performing various reasoning tasks, such as visual perception, speech recognition, analytics, decision-making, and translation between languages. The purpose of this study is to build a computational model for the effective treatment of diabetes in the Kenyan health care system. Such a system would take into consideration underlying factors that influence diabetic prescriptions such as individual history, key patient parameters such as glucose level, body mass index (BMI), insulin, blood pressure, age and diabetic pedigree.

2 Issues

Diabetes Mellitus refers collectively to a group of diseases resulting from dysfunction of the glucoregulatory system. The International Diabetes Federation estimates that, by 2017, diabetes affected 425 million people worldwide, of whom, 4 million died in the same year. These figures are expected to increase dramatically in the coming decades, placing a rising burden on health care systems. This is especially so in the developing countries such Kenya. A wide range of therapeutic options is available for patients with diabetes.

Intelligent algorithms are widely used in data driven methods to support advanced analysis and provide individualized medical aid. In the Kenya, the treatment of diabetes is both costly and sometimes even unaffordable. Many complications have occurred in cases where diabetes was not detected on time and treated. The complex identifying process usually results in the patient visiting a diagnostic centre and consulting a doctor. The complexity of diabetes prognosis and management provides a problem window for computational models to provide key solutions that empower both patients and caregivers in their everyday life. This will provide a computational prototype model that prognosticates the likelihood of diabetes in a patient with maximum accuracy.

3 General Objectives

The general objective of the research was to build an Intelligent Computational Model that could improve diabetic management for patients in the Kenyan healthcare system as well as help solve the attendant challenges of cost, access and accurate diagnosis. The first objective was to develop machine learning computational model for the detection and diagnosis of diabetes. This would improve on the accuracy in the prognosis of diabetes. Secondly the research sought to develop an intelligent system capable of processing of symptoms data and information that could help improve the treatment outcomes of a diabetic patient. Thirdly the system developed would have a data visualization capability for effective communication as well a data model for diabetes- diabetes types, risk factors, symptoms and the subsequent diagnosis or prediction.

4 Related Work

4.1 Introduction

Diabetes mellitus (DM) is defined as a group of metabolic disorders exerting significant pressure on human health worldwide. Extensive research in all aspects of diabetes (diagnosis, etiopathophysiology, therapy, etc.) has led to the generation of huge amounts of clinical data. Design of computational models for diabetes diagnosis has been an active research area for the past decade.The potential of AI to enable diabetes solutions has been investigated in the context of multiple critical management issues. In this research, we use the diabetes management categories that include blood glucose control, prediction, detection of adverse glycaemic events, insulin calculation, life-style tendencies and the daily-life support in diabetes management. AI is attracting increased attention in this field because the amount of data acquired electronically from patients suffering from diabetes has grown exponentially. By means of complex and refined methods, AI has been shown to provide useful management tools to deal with these incremental repositories of data.

Artificial intelligence is defined as a branch of computer science that aims to create systems or methods that analyse information and allow the handling of complexity in a wide range of applications (in this case, diabetes management). Although the application of AI algorithms involves highly technical and specialized knowledge, this has not prevented AI from becoming an essential part of the technology industry and contributing to major advances within the field. This section will provide a short review of several well-known computational intelligence paradigms. In this study, we categorized methodologies with respect to the objective sought: to explore and discover information, to learn using information, or to extract conclusions from information. The figure 1 below depicts a taxonomy of artificial intelligence methods.

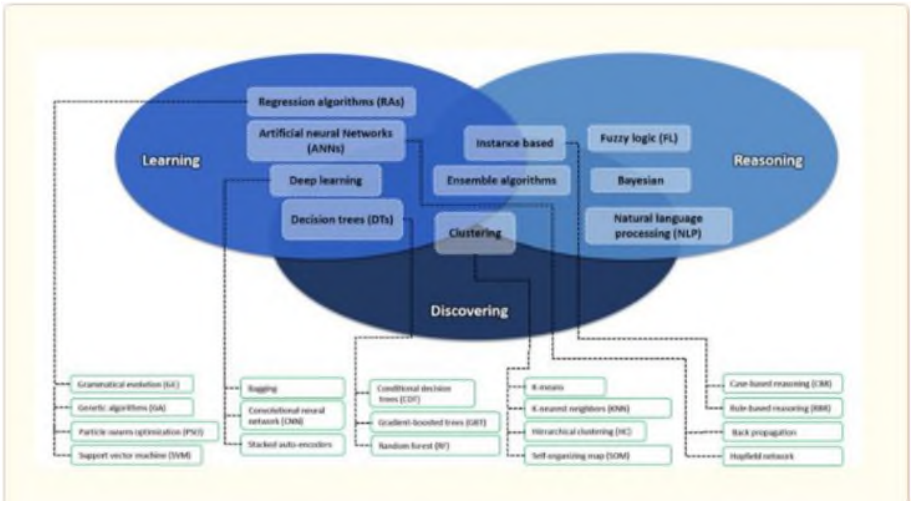

Learning
Reasoning
Discovering
Regression algorithms (RAs)
Artificial neural Networks (ANNs)
Deep learning
Decision trees (DTs)
Instance based
Ensemble algorithms
Clustering
Fuzzy logic (FL)
Bayesian
Natural language processing (NLP)
Grammatical evolution (GE)
Genetic algorithms (GA)
Particle swarm optimization (PSO)
Support vector machine (SVM)
Bagging
Convolutional neural network (CNN)
Stacked auto-encoders
Conditional decision trees (CDT)
Gradient-boosted trees (GBT)
Random forest (RF)
K-means
K-nearest neighbors (KNN)
Hierarchical clustering (HC)
Self-organizing map (SOM)
Case-based reasoning (CBR)
Rule-based reasoning (RBR)
Back propagation
Hopfield network

Figure 1: Artificial Intelligence Methods

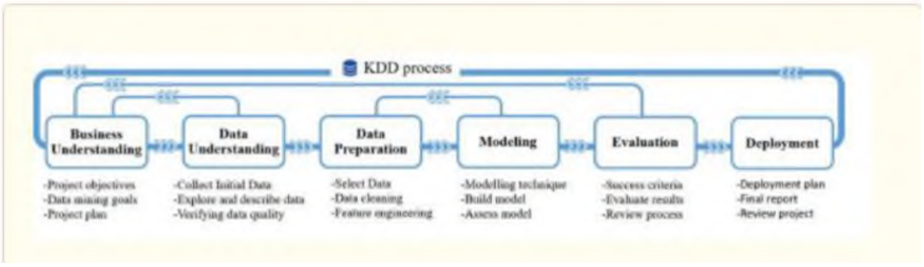


Figure 2: Knowledge Discovery Process

4.2 Clinical and Computational Intelligence

Chronic conditions like cancer, diabetes, mental disorders, cardiovascular and respiratory diseases account for 36 out of the 57 million deaths annually, thus the need for Business Intelligent systems to help in the management. According to Ngemu, 2015, Computational Intelligence is a broad category of applications and technologies for gathering, storing, analyzing and providing access to data to help enterprise users make better data models. Computational and business intelligence improves decisions by supplying timely, accurate, valuable, and actionable insights. With the rapid advancement and development of Information and Communication Technologies (ICT), health care providers are now able to generate, collect and distribute huge amounts of data from internal and external sources, and use this data in creating efficient models for the detection, diagnosis and treatment of diabetes.

Several studies applied artificial intelligence to systems aimed at supporting patient decisions by issuing advice regarding meals, exercise, or medication. Research groups at the Imperial College London performed an extensive study of an insulin bolus calculator based on case-based reasoning methodology. Their approach, which manages various dynamically optimized diabetes scenarios, was proven in a clinical trial to be a safe decision support tool. Additionally, this approach was demonstrated to improve glycaemic control in diabetes management. A similar approach was presented recently by another group, which also proposed an insulin bolus calculator based on case-based reasoning but, in contrast to other bolus calculators, it used a novel temporal retrieval algorithm. More recently, another study presented an approach based on artificial neural networks (ANN) and K-Nearest Neighbours to optimize bolus calculation by patients. The results revealed that it was better at reducing the blood glucose risk index value than other approaches. Finally, Lee et al proposed an advisory treatment system that provides insulin, meal, and exercise recommendations using a decision tree algorithm. The study, which compared rule-based reasoning and k-nearest neighbour algorithms, concluded that the decision tree based algorithms are best suited to this approach.

4.3 Conceptual Design

Diabetes is a disease that occurs when the insulin production in the body is inadequate or the body is unable to use the produced insulin in a proper manner, as a result, this leads to high blood glucose. The body cells break down the food into glucose and this glucose needs to be transported to all the cells of the body. The insulin is the hormone that directs the glucose that is produced by breaking down the food into the body cells. Any change in the production of insulin leads to an increase in the blood sugar levels and this can lead to damage to the tissues. Diabetes is a disease that occurs when the insulin production in the body is inadequate or the body is unable to use the produced insulin in a proper manner, as a result, this leads to high blood glucose.There are three main types of diabetes:

Type 1 – Though there are only about 10% of diabetes patients have this form of diabetes. The disease manifest as an autoimmune disease occurring at a very young age of below 20 years hence also called juvenile-onset diabetes. In this type of diabetes, the pancreatic cells that produce insulin have been destroyed by the defence system of the body. Injections of insulin along with frequent blood tests and dietary restrictions have to be followed by patients suffering from Type 1 diabetes.

Type 2 – This type accounts for almost 90% of the diabetes cases and commonly called the adult-onset diabetes or the non-insulin dependent diabetes. In this case, the various organs of the body become insulin resistant, and this increases the demand for insulin. At this point, pancreas does not make the required amount of insulin. To keep this type of diabetes at bay, the patients have to follow a strict diet, exercise routine and keep track of the blood glucose.

Gestational diabetes – is a type of diabetes that tends to occur in pregnant women due to the high sugar levels as the pancreas do not produce sufficient amount of insulin. Taking no treatment can lead to complications during childbirth. Controlling the diet and taking insulin can control this form of diabetes.

Though both Type 1 and Type 2 diabetes cannot be cured, they can be controlled and treated by special diets, regular exercise and insulin injections. The complications of the disease include neuropathy, foot amputations, glaucoma, cataracts, increased risk of kidney diseases and heart attack, stroke, and many more.

The data is collected from real time repository and it conforms to Type II diabetes based on the given attributes. The research explored the use of Decision Tree and K-Nearest Neighbor Classifier as machine learning techniques in diagnosing diabetes. The main objective being to forecast if the patient has been has diabetes using data mining tools from the medical data available.

Several prototypes of diabetes care systems have been designed and implemented. The first general steps is to ensure that that system will capture new patient data accurately. In most cases, it is upon the patient to provide these data and there should be a support mechanism to provide the data. The proposed i-Afya System for diabetes care is composed of an interactive and graphical user interface for capturing data and a core back end functionality that does the data interpretation and processing. After the processing, visual presentations of the data will be presented. The core processing includes a classification algorithm using decision tree model. The system has transformation capabilities for the learning, which includes replacing missing values and normalization of values. Figure 4 below gives a high-level view of the system model.

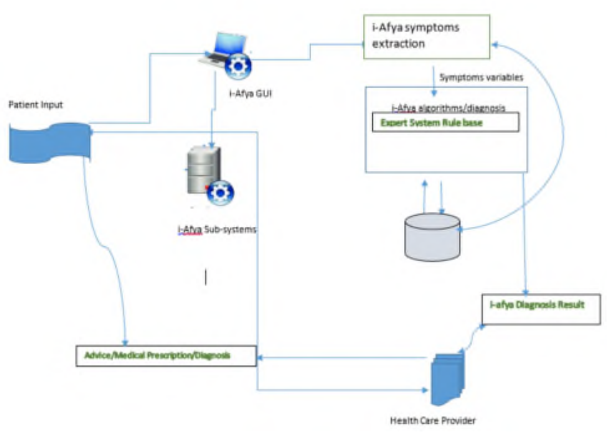


Figure 4: System High Level Architecture

5 Research Approach

Two algorithms namely decision tree classification and regression tree (CART) algorithm and KNN have been used to create the model for diagnosis. The data was divided into a training set and formatted using CVS. It was validated using the cross-validation technique and percentage split technique.

5.1 Data Sources

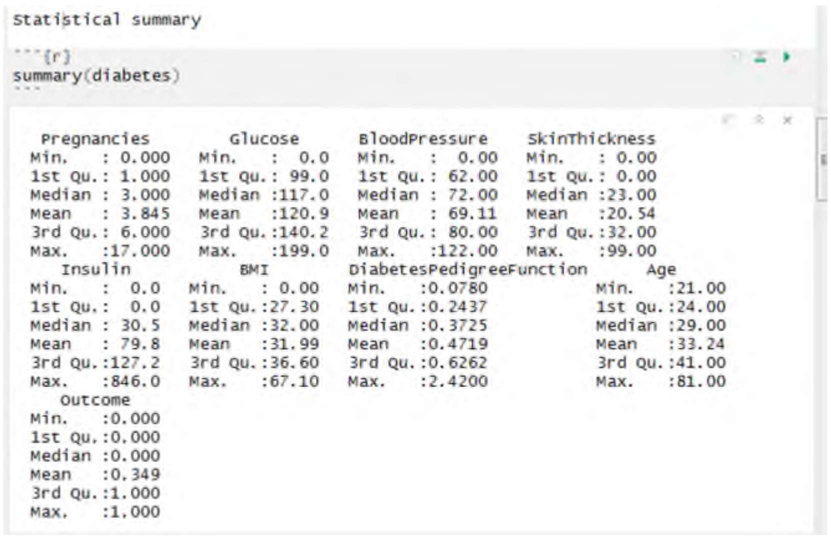
A cross sectional study conducted for people living with diabetes in Kenya by Novartis research team in Nairobi and other towns by June 2015 pre-released data indicated that diabetes care receivers under Novartis were about two thousand. Data for carrying out this research and project was sourced from the Diabetes Association of Kenya, which presented the status of various health care providers as well as the quality of health care provided to diabetic patients. Selected interviews were also done with Physicians specializing in diabetes treatment. Nairobi provided the bigger samples of the data as it had 70% of the registered patients. The estimated sample size of the data was arrived by the satisfaction of the criteria.

The data was tested using the cross-validation technique and the percentage split technique. The dataset pre-processing was done using R programming language. Algorithm, which has libraries for normalizing data. Additional data operations were performed on the dataset to replace missing values. The Processed dataset was then parsed through feature selection wherein sets of attributes are typically deleted from the dataset. The final processed dataset was parsed through R scripts for prediction of any new instances using the developed i-Afya system. Both the KNN and decision tree algorithms were used.

5.2 Data Description and Pre-Processing

The clinical data harvested from Diabetes Kenya and the data sources was unstructured and had irrelevant attributes and thus required be prepared using relevant formats, processing and transforming for data evaluation and validation. The data was collected from real time repository and conformed to both Type 1 and Type II diabetes based on the given attributes. The data was collected and keyed into the model for learning purposes.

The data set had ten attributes, which when modelled using R language. The data had eight attributes for processing namely, BMI, skin thickness, insulin, age, number of pregnancies(for female patients), glucose, blood pressure and a variable known as diabetespedigreefunction. Exploratory data analysis and feature selection were carried out using R libraries and a statistical summary developed as shown in figure 6. After modelling of the data, the algorithms for implementing the system prediction module were developed and employed. The raw data was run on CSV data. The correlation between numeric variable was also implemented as well as the correlation between the variables and the outcome as shown in figure 9 and figure 10.



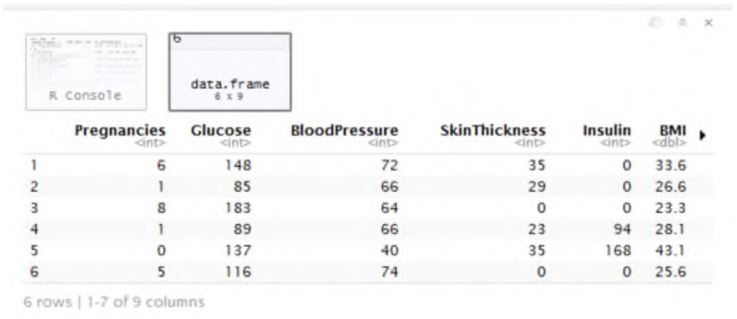
~~~
Statistical summary

```{r}
summary(diabetes)
```

  Pregnancies        Glucose        BloodPressure     SkinThickness
 Min.   : 0.000   Min.   :  0.0   Min.   :  0.00   Min.   : 0.00
 1st Qu.: 1.000   1st Qu.: 99.0   1st Qu.: 62.00   1st Qu.: 0.00
 Median : 3.000   Median :117.0   Median : 72.00   Median :23.00
 Mean   : 3.845   Mean   :120.9   Mean   : 69.11   Mean   :20.54
 3rd Qu.: 6.000   3rd Qu.:140.2   3rd Qu.: 80.00   3rd Qu.:32.00
 Max.   :17.000   Max.   :199.0   Max.   :122.00   Max.   :99.00
    Insulin           BMI        DiabetesPedigreeFunction      Age
 Min.   :  0.0   Min.   : 0.00   Min.   :0.0780          Min.   :21.00
 1st Qu.:  0.0   1st Qu.:27.30   1st Qu.:0.2437          1st Qu.:24.00
 Median : 30.5   Median :32.00   Median :0.3725          Median :29.00
 Mean   : 79.8   Mean   :31.99   Mean   :0.4719          Mean   :33.24
 3rd Qu.:127.2   3rd Qu.:36.60   3rd Qu.:0.6262          3rd Qu.:41.00
 Max.   :846.0   Max.   :67.10   Max.   :2.4200          Max.   :81.00
    Outcome
 Min.   :0.000
 1st Qu.:0.000
 Median :0.000
 Mean   :0.349
 3rd Qu.:1.000
 Max.   :1.000
~~~

Figure 5

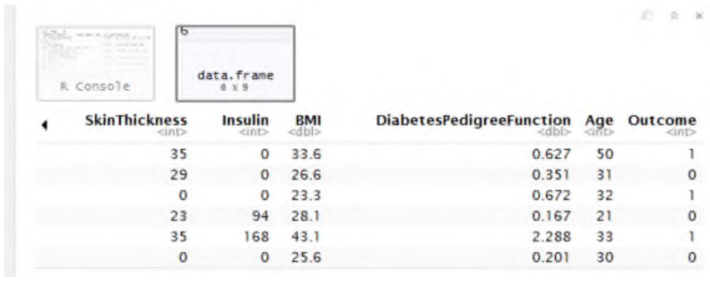
Figure 6 and 7 shows a brief description of the dataset that were being modelled and the relevant attributes.



	Pregnancies <int>	Glucose <int>	BloodPressure <int>	SkinThickness <int>	Insulin <int>	BMI <dbl>
1	6	148	72	35	0	33.6
2	1	85	66	29	0	26.6
3	8	183	64	0	0	23.3
4	1	89	66	23	94	28.1
5	0	137	40	35	168	43.1
6	5	116	74	0	0	25.6

6 rows | 1-7 of 9 columns

Figure 6 R Dataset Model



SkinThickness <int>	Insulin <int>	BMI <dbl>	DiabetesPedigreeFunction <dbl>	Age <int>	Outcome <int>
35	0	33.6	0.627	50	1
29	0	26.6	0.351	31	0
0	0	23.3	0.672	32	1
23	94	28.1	0.167	21	0
35	168	43.1	2.288	33	1
0	0	25.6	0.201	30	0

Figure 7 R Dataset Model

Figure 8 shows the distribution which shows that all variables have reasonable broad distribution.

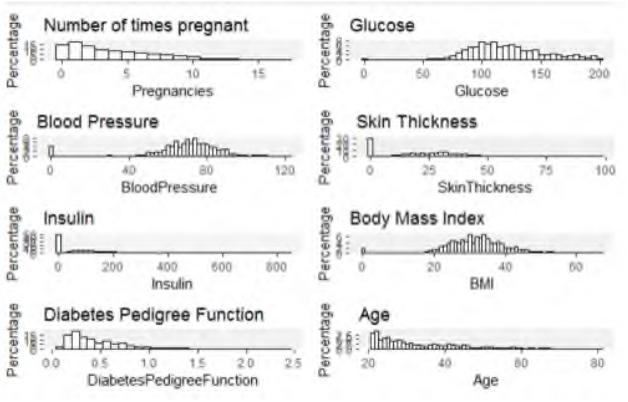


Figure 8 R Dataset Distribution

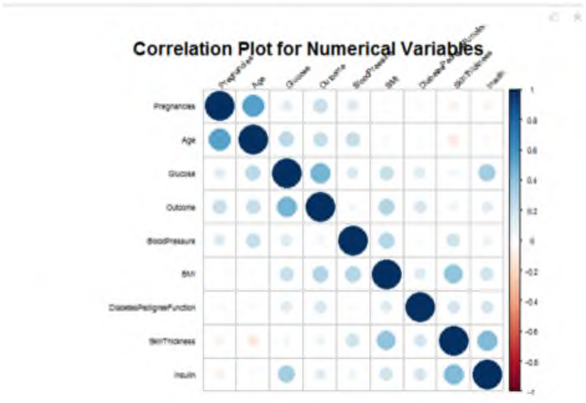


Figure 9 on the Correlation of the variables.

5.3 Algorithms Used

In this, study both models for Decision Tree classification (CART) and KNN were used and the more appropriate model chosen. It is important to note that Since KNN performs on-the-spot learning; it requires frequent database lookups, hence, can be computationally expensive. Decision Tree Classifier does not require such lookups as it has in-memory classification model ready. Since KNN performs instance-based learning, a well-tuned K can model complex decision spaces having arbitrarily complicated decision boundaries, which are not easily modelled by other "eager" learners like Decision Trees.

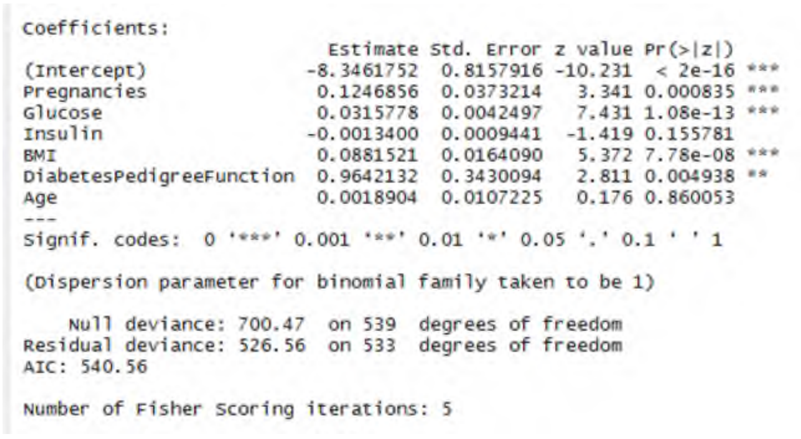
The main advantage of memory-based approach [the KNN] is that the classifier immediately adapts as we collect new training data. However, the downside is that the computational complexity for classifying new samples grows linearly with the number of samples in the training dataset in the worst-case scenario—unless the dataset has very few dimensions. The decision tree, however, can rapidly classify new examples. Therefore, given the high data availability, quick processing and accuracy needed, the i-Afya system adopted to use the decision tree algorithm. Decision tree are also easier to interpret in terms of representation of data and the complexity.

6 Results and Discussion

6.1 The i-Afya System Results

The logical computation was done using the system computational model and diagnoses done to show whether the patient showed diabetes according to the WHO criteria. The parameters used are real-valued between zero and one, transformed into a binary decision using a cut-off of 0.448. There were 769 training instances in the data set, thus 768 instances and 8 attributes namely Number of Times Pregnant, Glucose Level, Insulin (mu U/ml), Diastolic Blood pressure (mmHg), Skin Thickness measured in mm, Diabetes pedigree function, Age in years and finally the Pedigree Function which translated to the computational loads and efficiency in tree formation. Thus, this model focused on the output of the R libraries and the code implementation using the decision tree algorithm and the k-nearest (KNN) algorithm. The computational model developed using relative comparison of both the KNN and Decision Tree algorithm performances.

The findings from the i-Afya system model is that Blood pressure and skin thickness show little variation with diabetes, as such they can could considered to be little statististical value in the computation. The other variables show average correlation with diabetes, that is the glocuse level, insulin, body mass index and number of pregrancies. The top three most relevant features are "Glucose", "BMI" and Number of times pregnant" because of the low p-values. Insulin and age appear not statistically significant in the model. This was computed using regression analysis.



```
Coefficients:
                          Estimate Std. Error z value Pr(>|z|)
(Intercept)             -8.3461752  0.8157916 -10.231  < 2e-16 ***
Pregnancies              0.1246856  0.0373214   3.341 0.000835 ***
Glucose                  0.0315778  0.0042497   7.431 1.08e-13 ***
Insulin                 -0.0013400  0.0009441  -1.419 0.155781
BMI                      0.0881521  0.0164090   5.372 7.78e-08 ***
DiabetesPedigreeFunction 0.9642132  0.3430094   2.811 0.004938 **
Age                      0.0018904  0.0107225   0.176 0.860053
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 700.47  on 539  degrees of freedom
Residual deviance: 526.56  on 533  degrees of freedom
AIC: 540.56

Number of Fisher Scoring iterations: 5
```

From the prototype model, a classification is derived with the variables with the highest deviance being the root node. The top three most relevant features are "Glucose", "BMI" and Number of times pregnant" because of the low p-values. The classification tree is shown below. This means if a person's BMI less than 45.4 and her diabetes pedigree function less than 0.8745, then she or he is more likely to have diabetes. Decision tree classification implements algorithm for generating a pruned tree. The tree generated CART algorithm can is used for classification problem of whether a patient has tested positive or negative for diabetes. The data mining technique uses the concept of information gain. The output of the decision tree classification is show in figure 10.

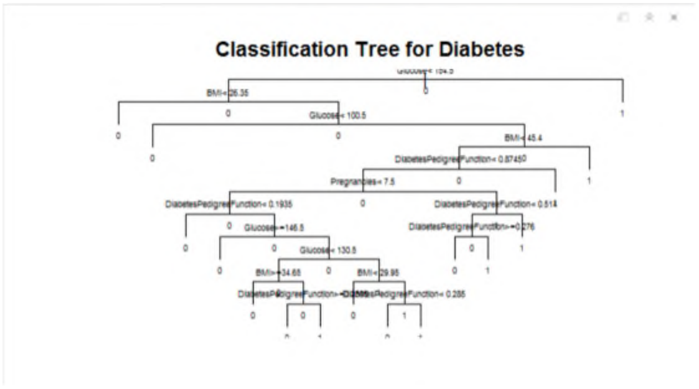

Classification Tree for Diabetes
BMI< 26.35
Glucose< 100.5
BMI< 45.4
DiabetesPedigreeFunction< 0.8745
Pregnancies< 7.5
DiabetesPedigreeFunction< 0.1935
DiabetesPedigreeFunction< 0.514
Glucose>=146.5
DiabetesPedigreeFunction>=0.276
Glucose< 130.5
BMI>=34.65
BMI< 29.95
DiabetesPedigreeFunction< 0.285

Figure 10

From the table of deviance, we found that adding insulin and age have little effect on the residual deviance. From the statistics and computational model the findings indicate that means if a person's BMI less than 45.4 and her diabetes pedigree function less than 0.8745, then she is more likely to have diabetes.

The i-Afya system model developed in this study is a diabetes self-management system that captures personal information, blood glucose levels, Blood pressure, Physical activities, Insulin dosage and Insulin variables that the patients utilizes during the self-management program. The system allows easy access to the knowledge database history through the graphical user interface (GUI), graphs, and charts for easier understanding of the data. The data is shared among the caregivers of the patient, health institutions and the General Practitioner portal.

7 Discussion

To accurately design a good prediction diagnostics system with a good model it is important to implement accurate and relevant algorithms, which can learn quickly in large data sets, and capture input characteristics. With a good prediction model and an accurate detection technique, diagnosis can be made more efficient for dynamic use of for disease detection tools. Based on the prediction methodology, medical practitioners can envision biomedical diagnosis by engineering tools, which can adapt to any future unexpected conditions automatically. A long-term prediction algorithm can definitely play a very important role in planning and provisioning. Thus, the behaviour of a real time data can be forecasted using machine-learning algorithms such as KNN and Decision tree. Ideally, such processes should be capable of accurately representing the statistical properties of the real data, which is not always possible because of several complex issues. In this research, the use of decision tree for the classification problem proved to be quite accurate and relevant for the study and purpose.

Our findings also show the increasing importance of AI methods for diabetes management. We think these methods will encourage further research into the use of AI methods to extract knowledge from diabetic data. In general, the most striking advances in the application of AI techniques come from data-driven methods that learn from large datasets. The ability to collect information from individual diabetic patients has led to a shift in diabetes management systems; accordingly, systems that lack access to valuable data will face substantial hurdles. Diabetes management will be geared towards tailored management therapies, at the level of smaller strata of patients or even individuals. Thus, management systems provided to diabetic patients should be tailored to address their needs at various points during their illness.

This study was able to confirm the positive effect of using a digital system for diabetic's diagnostics and management. The research showed that technology solutions for enhancing treatment of diabetes have a net positive effect on the treatment outcomes. The clinical decision support capabilities used in the system shows that the i-Afya intelligent computational power can be harnessed to provide data analytics for enhanced management of diabetes patients in Kenya.

The limitations of this study include not analyzing the long-term effects of the use of the i-Afya system and the selection bias of the subjects. This can be done by having more resources and timelines to implement the system across the country at public facilities. However, this study still has great significance in that the statistically significant positive changes in the clinical course of diabetes, which were displayed among users of the widely available application.

8 Conclusion

One of the important real-world medical problems is the detection of diabetes at early stage. In this study, systematic efforts were made in designing and implementing a system, which could result in the accurate detection and prediction of a disease like diabetes. The study thus successfully showed the application of the decision tree and KNN algorithms in disease diagnosis and the subsequent computation of large diabetes datasets to provide the correct solution. This study also revealed that the i-Afya Intelligent System resulted to a high level of user satisfaction and had a positive effect on diabetes management were it to be was used within the Kenyan health system. This has the potential to ultimately improve the treatment outcomes as well as lowering the cost of treatment, access and management of diabetes in Kenya.

REFERENCES

[1] Bonfa, I. et al, 1993. HERMES: An expert system for prognosis of hepatic disease, Proceeedings of First New Zealand International Two Stream Conference on Artificial Neural Networks and Expert Systems, pp. 240-246.

[2] Breault, J. L. et al, 2002. Data mining a diabetic data warehouse, Artificial Intelligent in Medicine, Vol. 26, pp. 37–54. Han, J. and Kamber, M. 2006. Data mining: Concepts and techniques, 2nd ed., Morgan Kaufmann Publishers: California.

[3] Hani, M. et al, 2010. Dengue confirmed-cases prediction: A neural network model, Expert System Application, Vol. 37, pp. 4256–4260.

[4] Vimala B, Vithyatheri G, 2011. An Intelligent Diabetes Diagnostic System for Diabetes Using Rule Based Reasoning and Object Oriented Methodology.

[5] P. Mbatha,2016. "Diabetic e-Care System. 30-49.

[6] American Diabetes Association Diagnosis and classification of diabetes mellitus. Diabetes Care. 2010 Jan;33 Suppl 1:S62–9.

[7] Greenwood NJC, Gunton JE, 2014. A computational proof of concept of a machine-intelligent artificial pancreas using Lyapunov stability and differential game theory. J Diabetes Sci Technol.

[8] Han J. Kamber. M, 2012. "Data Mining; Concepts and Techniques", Morgan Kaufmann Publishers.

[9] S. Priya, "An improved data mining model to predict the occurrence of Type 2 diabetes" ICON3C 2012, Proceedings published in IJCA. [4] T.Mitchell, "Machine Learning", McGraw -Hill, New York- 2 edition, 2010 [5] JIanchao Han, Juan C.Rodriguze, Mohsen Beheshti, "Diabetes Data Analysis and Prediction model discovery" IEEE, Second International conference on future generation communication and networking, pp 96-99, 2011.

[10] Asma A.Aljarullah, "Decision tree discovery for the diagnosis type 2 diabetes" IEEE, International conference on innovation in information technology, pp 303-307

[11] Aishwarya, R., Gayathri, P., Jaisankar, N., 2013. A Method for Classification Using Machine Learning Technique for Diabetes. International Journal of Engineering and Technology (IJET) 5, 2903–2908.

[12] Shumway R. (2005). Time Series Analysis with Applications in R. Springer Texts in Statistics.

[13] Zhou, Z. J., & Hu, C. H. (2008). An effective hybrid approach based on grey and ARMA for forecasting gyro drift. Chaos, Solitons and Fractals, 35, 525–529.

[14] Demuth, H., & Beale, B. (2004). Neural network toolbox user guide. Natick: The Math Works Inc.

[15] Arifovic, J., & Gencay, R. (2001). Using genetic algorithms to select architecture of a feed-forward artificial neural network. Physica A, 289, 574–594.

[16] Arora, R., Suman, 2012. Comparative Analysis of Classification Algorithms on Different Datasets using WEKA. International Journal of Computer Applications 54, 21–25. doi:10.5120/8626-2492.

[17] Joarder Kamruzzaman & Ruhul A. Sarker(2004). ANN-Based Forecasting of Foreign Currency Exchange Rates. Neural Information Processing - Letters and Reviews, 3, 2-2004.