Scalable Computing:
Practice and Experience

Scientific International Journal
for Parallel and Distributed Computing

ISSN: 1895-1767

Universitatea de Vest din Timişoara

Volume 21(4) December 2020

SUBSCRIPTION INFORMATION: please visit http://www.scpe.org

Scalable Computing: Practice and Experience

Volume 21, Number 4, December 2020

TABLE OF CONTENTS

Scalable Computing: Practice and Experience, ISSN 1895-1767, http://www.scpe.org
© 2020 SCPE. Volume 21, Issues 4, pp. 569–582, DOI 10.12694:/scpe.v21i4.1604

INVESTIGATION ON AGRICULTURAL LAND SELECTION USING HYBRID FUZZY LOGIC SYSTEM *

SUDHAKAR SENGAN† V. VIJAYAKUMAR‡ SUJATHA KRISHNAMOORTHY§ S. GUNASEKARAN¶ C. SATHIYA KUMAR‖ SARAVANAN PALANI** AND V. SUBRAMANIYASWAMY††

Abstract. For maintaining the horticultural generation, Land Selection Investigation (LSI) is essential. Though incorporates estimation of the criteria assortment from the soil, territory to financial, market, and foundation, and these components are considerably enigmatically characterized and described by their inherent ambiguity. Multi-criteria basic leadership systems like positioning, rating, and so on are utilized for reasonableness examination. Master learning and judgment by leaders at different levels is integrated into this process. In the field of farming sciences, the Fuzzy Logic (FL) strategy has been effectively used to take care of numerous issues. Fuzzy with AHP is a Hybrid Fuzzy Logic (HFL) methodology. The policies Analytic Hierarchy Process (AHP), Fuzzy Numbers, Fuzzy Degree Investigation, Alpha Cut, and Lambda capacity are associated with it. As expressed, the procedure of necessary leadership includes a scope of criteria and a considerable measure of master learning and decisions. The components result from impacts extraordinarily. The capacity of three methods to demonstrate the affectability of the necessary leadership procedure is researched. Alpha cut and lambda esteem give and encourage considerable affectability investigation. All techniques are actualized to examine the reasonableness of the crop in the Indian nation. Test results when performed on Various Datasets, demonstrate that the proposed procedure removes more highlights just as gives more exactness when contrasted with existing techniques.

Key words: Analytic Hierarchy Process, Fuzzy Logic, Land Selection Investigation, Soft Computing

AMS subject classifications. 94D05

1. Introduction.

Geographic Information System (GIS) of Land Selection Investigation (LSI)- based procedure connected to decide the appropriateness of a particular region for utilizing, i.e., it uncovers the suitability of territory for its reasonable or unsatisfactory. Likewise, this examination engaged with considering wide furies of criteria including ecological, social, and financial elements. Suitable treatment of such expansive and heterogeneous guide requires applying a flexible device. We will take a look at the reasonableness of parameters, similitude of settings with the notable climate and soil information gathered utilizing HFL. The community-oriented arrangement of suggestion falls under the recommender proposed yield. Conversely, the cost forecast and climate expectations. The blend of the Analytic Hierarchy Process (AHP) [1] and fuzzy sets settles on the decisions and choices progressively adaptable. HFL mirrors human personality when settling on choices dependent on surmised information and vulnerability. These techniques have scarcely been utilized in developing nations like India. This examination plans to show how incredible the FAHP (Fuzzy Analytic Hierarchy Process) strategy is in taking care of the LSI. For this propose, India, Tamil Nadu. Rangelands secure a broad zone of the district, and creature farming is the principal occupation of residents. The present investigation considered predominantly two parts of the environmental change effect over farming land, which

*The authors are grateful to the Indian Council of Social Science Research (ICSSR), New Delhi, for the financial support (No. IMPRESS/P580/278/2018–19/ICSSR) under IMPRESS Scheme. Authors express their gratitude to SASTRA Deemed University, Thanjavur, for providing the infrastructural facilities to carry out this research work.

†Department of CSE, Sree Sakthi Engineering College, Coimbatore-641 104, India (sudhasengan@gmail.com)

‡School of Computer Science and Engineering, University of New South Wales, Sydney, Australia (vijayakumar.varadarajan@gmail.com)

§Department of Computer Science, Wenzhou Kean University, Wenzhou Zhejing, China (sujatha.ssps@gmail.com)

¶Dept. of Computer Application, King College of Arts & Science, Tiruchengode-637 215, India (drsguna9596@gmail.com)

‖Department of Computational Intelligence, Vellore Institute of Technology, Vellore, India (csathiyakumar@yahoo.com)

**School of Computing, SASTRA Deemed University, Thanjavur, India (sharan.doit@gmail.com)

††School of Computing, SASTRA Deemed University, Thanjavur, India (vsubramaniyaswamy@gmail.com), Corresponding Author

incorporates [2], (i) Harvest profitability impact, and (ii) Soil natural carbon impact.

This study aims to present how powerful integrate the hybrid fuzzy set theory provides more sophisticated results as fuzzy set theories use advanced algorithms to address uncertainties, incompleteness, and vagueness and increase robustness associated with suitability criteria. The analytic hierarchy process is a multi-criteria method for assessing land-use suitability based on the Geographic Information System (GIS). The objective of this study was to identify suitable lands for crop production using Hybrid Fuzzy Logic techniques is in handling the land suitability analysis of Tamil Nadu. The paper highlights the use of different methods of land suitability evaluation for sustainable agriculture in developing India.

2. Literature Review.

2.1. An Overview of Land Evaluation. Just as various ways to deal with the procedure of land assessment, this section depicts an assortment of definitions and clarifications concerning arriving estimates. The [3] characterizes land assessment as the way toward evaluating the potential for elective sorts of land use and to anticipate the results of progress. It can recognize quantities of powers behind land assessment rising as a particular subject. Right off the bat, there is expanding the accessibility of biophysical information, and this information can be handled and exhibited in an assortment of ways. Also, nations are focusing on the difficulties of land-use scheduling. Dry Land areas (Ex: South Africa and Libya), have connected their feasible advancement objectives and land use scheduling. Thus, land-use scheduling can direct choices onto land use with the goal that they are put to the most practical method for the present while preserving a similar land for the future populace and their needs.

To decide its reasonableness/capacity, land assessment procedure might be done subjectively or quantitatively. Previously, a land assessment was utilized as a component of soil overview learning. Notwithstanding, since 1972 [4], land advancement has moved concentration to harvest development and creation, which incorporates viewpoints relating to atmosphere conditions, soil, and land administration. The following methodologies are used for LSI:

1. Farming creation by using mathematical terms influenced by parametric frameworks fuse land attributes. Numerous parametric methods have been used for land assessment. These methodologies shift in the particular parameters they incorporate and in their precise control.
2. Creation units as per the units shifting possibilities and restrictions influencing crop development concentrated on the arrangement of the property by Categorical frameworks [5].

The creative ability to utilize land deliberates investigation of both land's physical conditions and their effect on the present and future property for surveying Land assessment. The land assessment offers a system for looking at the changing ways that the area can be utilized, just like the advantages that might be gotten from these utilizations, considering the present and future financial and social conditions [6].

The land assessment procedure won't characterize the land use or any proposed changes in it. Instead, it gives information that can fill in as a reason for choosing which land choice utilization is appropriate. So, the land assessment helps land proprietors, provincial land advancement offices, and countries to touch base at intelligent land-use choices. In any case, there are absolute necessities for land assessment to be effectively used. A considerable lot of these prerequisites are explicit to the kind of land use, and they incorporate both the natural necessities of the harvest or other organic items and the requirements of the administration framework used to create it. The assessment of land assets is a mix of the properties of the land with the essentials of proposed land use [7].

The Framework for LSI is introduced by the standards of land assessment follows and represented in Fig. 2.1.

- Initial consultation, about the goal assessment concerned, information and presumptions characterizing
- The sorts of land use description to be considered, and foundation of their prerequisites that can bolster specific land use
- Base unit's asset description
- Land use of sorts' comparison
- Social examination and Economic
- Characterization of land appropriateness
- Presentation of the aftereffects of the assessment into a structure usable via land clients.

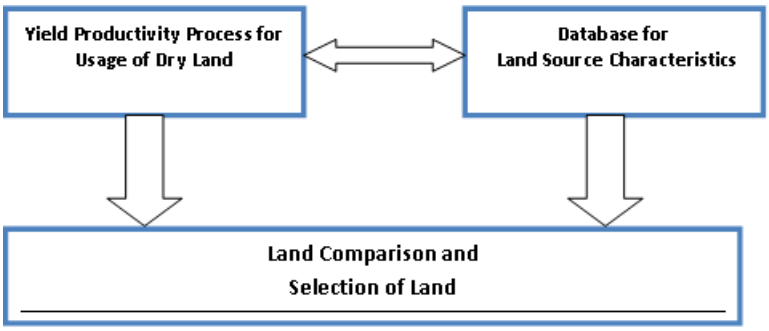


FIG. 2.1. *Process of land selection evaluation.*

2.2. The Need for Land Evaluation. The FAO contends that before, land use changes frequently occurred by slow development because of many separate choices taken by people. The expanded interest for physical space and sustenance from extending populace, the accessibility of reasonable land for generation making the land a rare asset, and even the less appropriate or minimal terrains had been exposed to development. A thorough evaluation of land is required. In developing nations, the developing requirement for increasingly profitable land types, the booking and safeguarding of land for horticulture, in addition to the growing worry to ensure the earth has made an interest for a complete survey of land space and its sanity. To accomplish this, what is required is a comprehensive stock of natural assets for a legitimate appraisal of land's appropriateness for creation purposes.

2.3. LSI and usage formation. Deciding reasonable land for a specific purpose is an intricate procedure, including numerous choices that may identify with biophysical, financial, and institutional/authoritative perspectives. An organized and steady way to deal with LSI is along these lines fundamental. Abiotic, biotic, and economic elements choose the accomplishment of a yield. Decisions concerning harvest worth ought to incorporate the abiotic, biotic, and business components that decide the productivity [8].

Land assessment of the FAO Framework is created from before land ability draws near. Here, by and large, land reasonableness of a land zone for specific land purposes is assessed from a lot of comparatively much autonomous land characteristics, which may each restrain the land-utilize probable. These assessments frequently arrange map units of healthy asset inventories. Like this, legend classifications of a soil study are characterized into appropriateness sub-classes, in light of the number and seriousness of confinements to land use. The FAO Framework detects the types of growing details indicated in Table 2.1.

TABLE 2.1
Formation of Land Selection Taxonomy.

Taxonomy of Land Type	Description
LSI: Class I	Type of selection
LSI: Class II	Selection of Land with degrees
LSI: Class III	Grouping the measurable scale factors of Land
LSI: Class IV	Land Service Management

The size of the estimation view appropriateness has two sorts of groupings in LSI structure [9, 10].

- Subjective: It utilized to assess ecological, social, and monetary criteria. The classes are determined depending on the physical generation capability of the land, generally used in surveillance examines.
- Measurable The types are characterized in like manner numerical relations, where the correlation between the goals is conceivable. Here extensive measure of monetary criteria is utilized.

Land appropriateness is a part of the assessment maintainability of land purpose. Rationality, together with weakness, characterizes the supportability of land. The supportable area should have the greatest appropriateness and least helplessness, and it represented in Fig. 2.2.

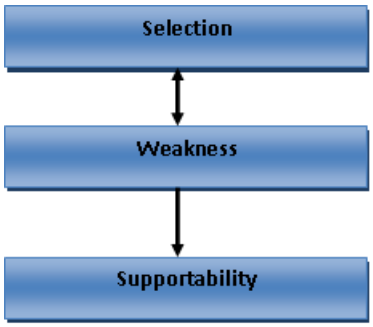


FIG. 2.2. *Land Utilize Supportability.*

2.4. Calculation Logic for Land Evaluation. Scientific models based on a land assessment on which there are numerous sorts of estimation logic. The primary ones are Boolean logic, HFL, and artificial neural systems.

2.4.1. Boolean Logic. A ruled-based methodology followed by the Boolean logic, where the cutoff points of sets are characterized, with the goal that a component does or does not have a place with a determinate set. It is the logic of true or false, generally utilized in the handy science, the rationale that pervades the FAO strategy. As indicated by that approach, the soil might be entirely reasonable, modestly appropriate, hardly intelligent, or not wise. There is no plausibility of depicting the slight differentiation between the classes, as in-between classes are not considered. As this strategy neglects to join the estimated idea of land information, there is developing mindfulness for an evaluation pattern that catches fuzziness, as found in the accompanying area [11].

2.4.2. Fuzzy Logic. FL characterized the term fuzziness, and he expressed, as *multifaceted nature rises, exact articulations lose meaning and significant proclamations lose exactness* [12]. From this announcement, Zadeh (1965) presented the idea of FL, where the reality of any report turns into a matter of degree. This hypothesis is an augmentation of regular Boolean logic that was acquainted with purpose the term of incomplete truth between totally evident and false. Zadeh has utilized this term as a way to demonstrate the uncertainty of natural language. However, the methodology has been connected to displaying numerous procedures that are mind-boggling and not well characterized.

An FL is a numerical method to describe and manage uncertainty in regular daily existence. FL demonstrated that one of the reasons that people have greater control than machines is that they are equipped for settling on effective choices due to inaccurate linguistic information. It specifies old-style sets hypothesis in which the enrollment level of any article to a collection constrained to the whole numbers 0 and 1 just by enabling the participants to take any certain amount somewhere in the range of 0 and 1. By this definition, a fuzzy set is defined with uncertain limits in which the progress starting with one set then onto the next is continuous as opposed to sudden. Prior, the multi-criteria land reasonableness was surveyed more non-spatially, expecting the spatial uniformity over the region under thought. This, in any case, is ridiculous in cases like land reasonableness studies, where choices made utilizing criteria that change transversely over in space. Non-spatial predictable Multi-criteria Decision Making (MCDM) procedures routine or absolute the effects that are decided suitable for the entire zone under thought. To address the essential spatial leadership, MCD and Geographic Information Systems (GIS) can be coordinated.

Analytic Hierarchy Process (AHP) utilizes the framework by [13], Ideal Vector Approach, and Fuzzy AHP(FAHP). A multi-criteria basic leadership strategy is created using FL, and land reasonableness is investigated for horticultural harvests. Considerably more factors like soil, atmosphere, water system, framework, and financial elements considered. Be that as it may, restricted to a tiny region (620 sq. km) and confined to a solitary harvest (rice).

3. Material and Methods.

3.1. Study area. Ramanathapuram District (RD) lies on the Southern Zone (SZ) and is limited on the east by Palk strait of RD–SZ, in the northwest by Virudhunagar and in North East (NE) by Sivagangai districts of the SZ, in the south by Gulf of Mannar of the SZ and in the NE by Pudukkottai district of the SZ. It is located at 9.05° to 9.50°/ 78.10° to 79.27°. The study is mentioned in Fig. 3.1.

On the SZ, RD lies and is limited in the east by Palk Strait of RD of the SZ, in the northwest by Virudhunagar and in NE by Sivagangai regions of the SZ, in the south by Gulf of Mannar of the SZ and in the North by Pudukkottai locale of the SZ. It located: 9.05° to 9.50°/78.10° to 79.27°. The study is referenced in Fig. 3.1.

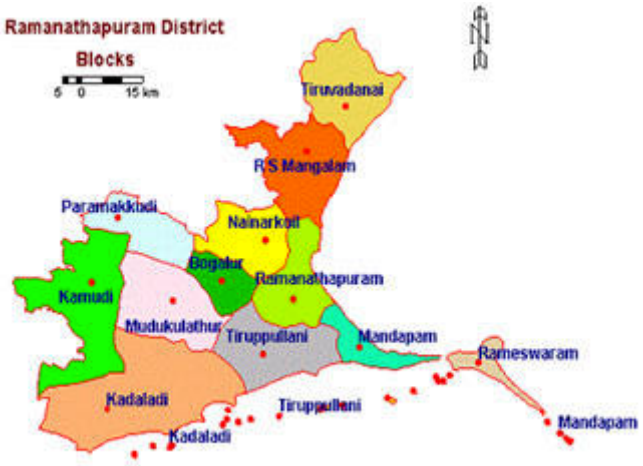


FIG. 3.1. *LSI examination of RD.*

The maximum temperature standard varies 29.2°-37.8° C/19.5°-24.8° C. During the long stretches of March, the contrast between the mean greatest and the mean least temperature is most elevated. By and large, the hum'idity rate extends 75% to 79%. The most noteworthy relative percentage level of 85% is recorded during November, and the least relative moistness level of 75% is documented during May in this RD. The Wind speed is minimal during the period of October-November /October-March. The wind blows by and large from NE directions. South West (SW) breezes are transcendent from May to September. The primary wellspring of Irrigation water occurs in NE Monsoon. Typical yearly precipitation: 827.0 mm, Average Winter Rain: 67.4 mm; Summertime: 122.7 mm; SW Rainy season: 135.3 mm and NE Monsoon Drizzle: 501.6 mm.

The RD soils can be varied into the primary types viz., clay, coastal alluvium, etc., Coastal alluvium happens in neighbor areas of RD. There are vast stretches of saline and underlying soils found in the waterfront squares. Rameswaram Island (RI) contains mostly sandy soil. The nitrogen status of the soil is demonstrated by the productiveness status of soil block is low, and phosphorus status of the soil is likewise moderate in all blocks aside from the nearest regions of RD where it is medium. The potash substance of soil is high in every one of the blocks. The mineral assets of the soil incorporate gypsum, limestone, and magnesium. While Mudukulatur and Kilakarai districts represent sizable stores of gypsum. RI contains enormous amounts of limestone stores. The area of the total cropped zone is 172469 Hz. The territory under inundated farming is 63800. Hectare, while 137099 hectares, is under rainfed Agriculture. The significant nourishment grain yields developed are Paddy, Cholam, Cumbu, Ragi, and Blackgram. The significant non-sustenance yield produced in cotton.

3.2. The process of planning and decision-making. GIS was limited distinctly for the way toward mapping [14] in the early days for the utilization of remote detecting. In time progress in the data innovation created devices to utilize these maps during the time spent arranging and essential leadership. Land, being a

valuable asset, requires to be overseen economically to help life on earth. Maintainable administration implies the usage of the accessible land assets so that the occupation, which is directed over a real portion of the property, is without or with least control over the assets.

The land utilization is feasible; the area should be utilized for a particular reason, which suits the confined conditions best. A horticultural territory should be described and assessed over its probability, restrictions, and imperatives that are affected by various Land Use Types (LUTs) [15]. The precise presentation of horticulture requests for the assessment of the land for the particular land use types in it. This LSI includes the interdisciplinary criteria running from financial to ecological. These numerous criteria that are affecting the LUT change over space, for example, standards, esteem change here and there, and are interrelated. Henceforth, there is an incredible requirement for the assessment of these criteria in spatial space. A few choices should be taken, and master information must be consolidated at different stages in the reasonableness examination, it indicates the following flow chart Fig. 3.2.

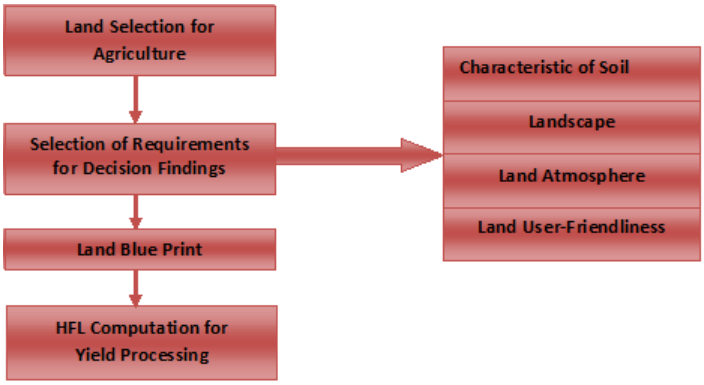


FIG. 3.2. *Proposed Methodology for LSI.*

3.2.1. The procedure of MCDM ordered on a few criteria.

1. Multi-Attribute / Multi-Objective Decision Making in light of how the criteria are being dealt with, as a trait/target.
2. Singular/Group Decision Making because of the number of individuals associated with the necessary leadership process.
3. Essential leadership under Certainty and Decision Making under Uncertainty, in light of the circumstance under which necessary guidance is being done and the idea of the criteria.

4. Proposed Framework of LSI-decision making. The decision-making problem of LSI for crops is analyzed using the Simons ideal with required adaptations. Fig. 4.1 depicts the abstract flow of the investigation method. LSI for farming crops investigated utilizing the Simons model with the fundamental leadership issue with necessary changes.

4.1. HFL approach in LSI. The HFL is an AHP position that evaluates various criteria using fuzzy numbers. While AHP depends on utilizing the Crisp numbers, FAHP has overcome the blemishes of AHP. Since uncertainty is a typical normal for some underlying leadership issues, the FAHP technique [16] has been created to make up for that defect. Hence HFL can dispense with the uncertainty and uncertainty from the appraisal with regards to confounded and multi-list issues, and it pointed in Fig. 4.2.

A triangular fluffy number communicates the overall quality of pair components in similar order and can be indicated as $M = (l, m, u)$ where $l \leq mc \leq u$. The limits l, m, u show the min. Feasible value, the most encouraging value, and the most significant conceivable significance, separately, in a fuzzy occasion. A three-sided sort enrollment capacity of M fuzzy number can be portrayed as in Eqn. (4.1). Fig. 4.3, at the point when $l = m = u$, it is a non-fuzzy integer by principle.

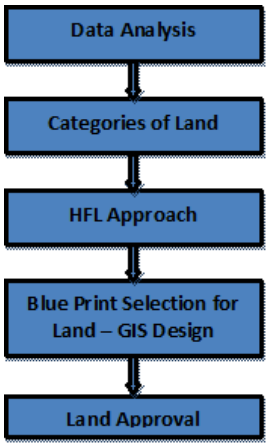


FIG. 4.1. *Flow Chart of Land Selection Investigation.*

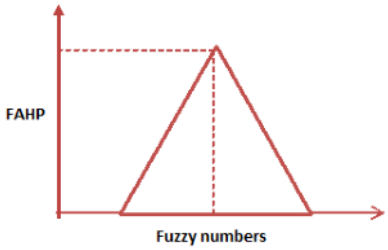


FIG. 4.2. *Fuzzy triangular number.*

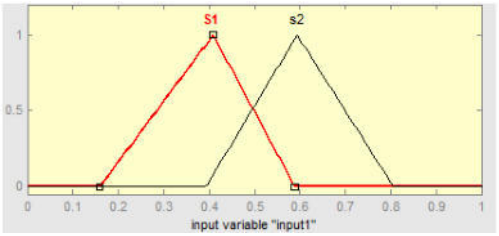


FIG. 4.3. *The point of probability* $V(S_i \geq S_j)$

$$f(x) = \begin{cases} \frac{x-l}{m-l}, & l \leq x \leq m \\ \frac{u-x}{u-m}, & m \leq x \leq u \\ 0, & \text{otherwise} \end{cases} \tag{4.1}$$

Various strategies have been displayed in literature, and the fuzzy examination [17] [18] is one of the techniques recommended. In the current analysis, the fuzzy investigation is connected because it is a more straightforward computation technique in correlation with other HFL strategies. Triangular fuzzy numbers utilized in A pairwise correlation network $\tilde{A}(a_{ij})$, which could be scientifically communicated as pursues the following Eqn (4.2).

$$\tilde{A} = (\tilde{a}_{ij}) = \begin{bmatrix} (1,1,1) & (l_{12}, m_{12}, u_{12}) & \dots & (l_{1n}, m_{1n}, u_{1n}) \\ (l_{12}, m_{12}, u_{12}) & (1,1,1) & \dots & (l_{2n}, m_{2n}, u_{2n}) \\ (l_{12}, m_{12}, u_{12}) & (l_{2n}, m_{2n}, u_{2n}) & \dots & (1,1,1) \end{bmatrix} \tag{4.2}$$

$$\tilde{a}_{ij} = (l_{ij}, m_{ij}, u_{ij})$$
$$\tilde{a}_{ij}^{-1} = (1/u_{ij}, 1/m_{ij}, 1/l_{ij})$$
$$i \text{ and } j = 1, \ldots, n, i \neq j$$

4.1.1. The steps of HFL exploration could be described as follows.

1. Sum each row of the fuzzy comparison matrix $\tilde{A}$. Then standardize the row sums by the fuzzy arithmetic operation:
   The means of fuzzy Chang's degree investigation could be clarified as pursues:
   Initial step: Sum each line of the fuzzy examination framework $\tilde{A}$. At that point, standardize the column aggregates by the fuzzy number arithmetic activity, its denoted Eqns. (4.3) and (4.4).

$$\tilde{A} = (\tilde{a}_{ij})_{n*n} = \begin{bmatrix} (1,1,1) & (l_{12}, m_{12}, u_{12}) & \ldots & (l_{1n}, m_{1n}, u_{1n}) \\ (l_{12}, m_{12}, u_{12}) & (1,1,1) & \ldots & (l_{2n}, m_{2n}, u_{2n}) \\ (l_{12}, m_{12}, u_{12}) & (l_{2n}, m_{2n}, u_{2n}) & \ldots & (1,1,1) \end{bmatrix} \tag{4.3}$$

$$\tilde{a}_{ij} = (l_{ij}, m_{ij}, u_{ij})$$
$$\tilde{a}_{ij}^{-1} = (1/u_{ij}, 1/m_{ij}, 1/l_{ij}) \tag{4.4}$$
$$i \text{ and } j = 1, \ldots, n, i \neq j$$

2. Compute the level of chance for $S_i \geq S_j$ by condition, refer the below Eqn. (4.5) and (4.6):

$$V(S_i \geq S_j) = \text{supply} \geq x[\min(S_j(x), S_i(y))] \tag{4.5}$$

$$V = (\tilde{S}_i \geq \tilde{S}_j) = \begin{cases} \frac{u_i - l_j}{(u_i - m_i) + (m_j - l_j)}, & m \leq x \leq u \\ 9, & \text{otherwise} \end{cases} \tag{4.6}$$

3. Estimate the precedence vector $W = (w_1, \ldots, w_n)T$ of the fuzzy examination matrix $\tilde{A}$ as pursues, Eqn. (4.7):

$$W_i = \frac{l_i + u_i + m_i}{3}, \quad i = 1, 2, \ldots, n \tag{4.7}$$

4. Standardize the determined loads of every foundation as Eqn. (4.8):

$$NW_i = \frac{W_i}{\sum_{j=1}^{n} w_j}, \quad i = 1, 2, \ldots, n, \text{ where } \sum_{j=1}^{n} w_j = 1, \ i = 1, 2, \ldots, n \tag{4.8}$$

To play out a pairwise correlation among fuzzy factors, semantic elements have been characterized for a few degrees of inclination in Table 4.1.

The following equation determines a Stability Ratio (SR).

$$SR = \frac{S_i}{R_i}, \quad SR = \frac{\lambda_{mx} - m}{m - 1} \tag{4.9}$$

where

SR – random table

λ – stability direction of the regular cost

m – number of measures

ZR – subjective table.

SR is planned so that if $ZR < 0.10$, the proportion shows a reasonable degree of consistency. Be that as it may, $ZR > 0.10$ demonstrates conflicting decisions. The capacity of HFL in consolidating various kinds of information and the vulnerability technique for pair-wise examinations utilized to all the while by analyzing two bounds for the reasons for ordering land appropriateness for rice development in the investigation locales in India.

TABLE 4.1
Triangular fuzzy number of semantic limits used in the test case.

Linguistic Limits	Crisp Pairwise Number	Pairwise Number	Fuzzy Values	Triangular Function	Common Triangular Fuzzy Values	Reciprocal Triangular Functions
Extreme	9		(9,9,9)		(0.11,0.11,0.11)	
Very Strong	7		(6,7,8)		(0.12,0.14,0.16)	
Strong	5		(4,5,6)		(0.16,0.2,0.25)	
Moderate	3		(2,3,4)		(0.25,0.33,0.5)	
Equal	1		(1,1,1)		(1,1,1)	
Intermediate	2,4,6,8		(7,8,9), (5,6,7), (3,4,5), (1,2,3)		(0.11,0.12,0.14), (0.14,0.16,0.2), (0.2,0.25,0.33), (0.33,0.5,1)	

5. Result and Discussion.

5.1. Implementation of the HFL in the context of LSI. HFL Contribution is the fundamental FL utilized in the HFL. The FL given by the specialists is fuzzified using triangular fuzzy numbers [19] (Table 5.1) to crop HFL (Table 5.2).

TABLE 5.1
LSI assessment environment

ec	*e*						
	NB	NM	NS	ZO	PS	PM	PB
NB	PB	PB	PM	PS	PS	ZO	NS
NM	PM	PM	PM	PS	ZO	NS	NS
ZO	PM	PM	PS	ZO	NS	NM	NM
PM	PS	PS	ZO	NS	NS	NM	NM
PB	PS	ZO	NS	NM	NM	NM	NB

PB – Positive Big, PM – Positive Medium,
PS – Positive Small, ZO – ZERO,
NB – Negative Big, NM – Negative Medium,
NS – Negative Small

TABLE 5.2
Fuzzified pair-wise assessment environment

CE	S_1	S_2	S_3	N_1	N_2
S_1	(1,1,1)	(1,3,5)	(4,6,8)	(6,8,10)	(7,9,11)
S_2	(0.2,0.3,1)	(1,1,1)(1,3,5)	(1,3,5)	(2,4,6)	(3,5,7)
S_3	(0.1,0.1,0.2)	(0.2,0.3,1)	(1,1,1)	(1,2,4)	(1,3,5)
N_1	(0.1,0.1,0.1)	(0.1,0.2,0.5)	(0.2,0.5,1)	(1,1,1)	(1,2,4)
N_2	(0.09,0.11,0.14)	(0.14,0.2,0.3)	(0.2,0.3,1)	(0.2,0.5,1)	(1,1,1)

HFL presentation of the matrix is deliberate, as given by the HFL presentation below Fig. 5.1.

Taking into account that the *ph* can be estimated with reasonable assurance, an alpha estimation of 0.612 is picked. It will produce the exhibition framework that limits esteems in Fig. 5.2.

To get a new weight framework from the range esteem grid is 0.578 is connected. The reason behind this is that measure how sure the master is with deference the factor being assessed. Worth 0.578 shows the master

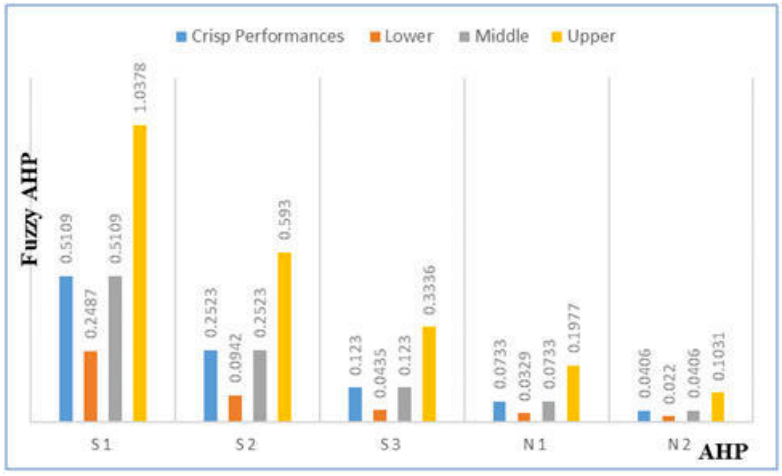


FIG. 5.1. *Performances comparison of AHP and HFL.*

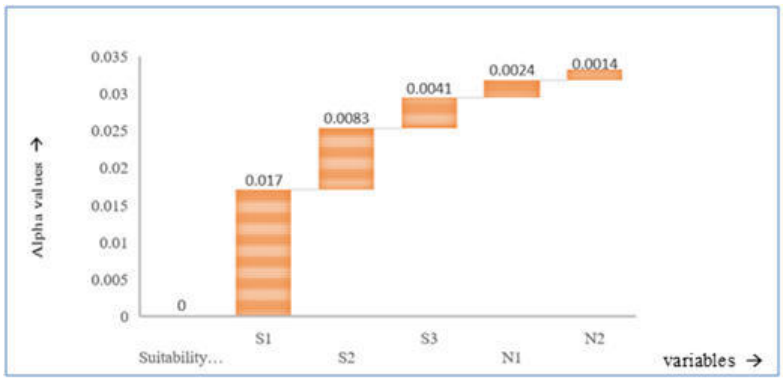


FIG. 5.2. *Purpose of Alpha Cut review.*

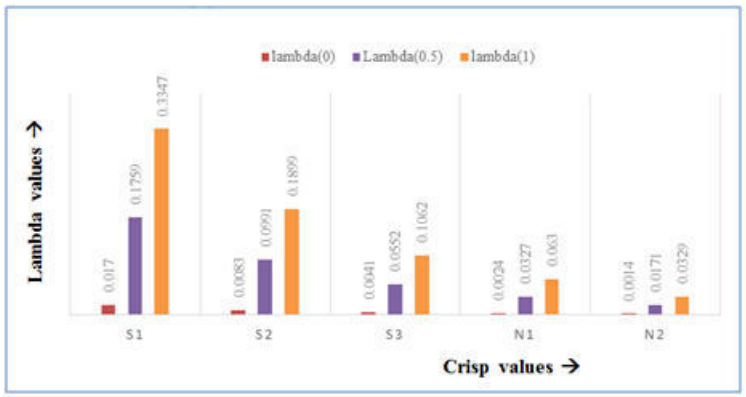


FIG. 5.3. *Performance principles obtained at lambda values.*

isn't that sure for his choices or inclinations, a certain measure of vulnerability exists in these inclinations and represented Fig. 5.3.

5.2. Implementation of HFL. Contributions for the HFL approach are the fresh FL. The new FL is fuzzified utilizing the triangular enrollment works, as portrayed in passage 4.6. The FL for every reasonableness class is the contributions for fuzzy degree investigation to bring about fluffy exhibitions per appropriateness class. Similarly, the FL built by the examination among criteria in a gathering in the chain of command order is fuzzified to get fuzzy shows per standards. The fuzzy demonstrations for measures increased with fluffy presentations for classes. The augmentation executed over the chain of importance up to the primary level. In the last arrangement, these exhibitions handled with the alpha cut investigation and lambda capacities. The outcome of the methodology is portrayed in Fig. 5.4.

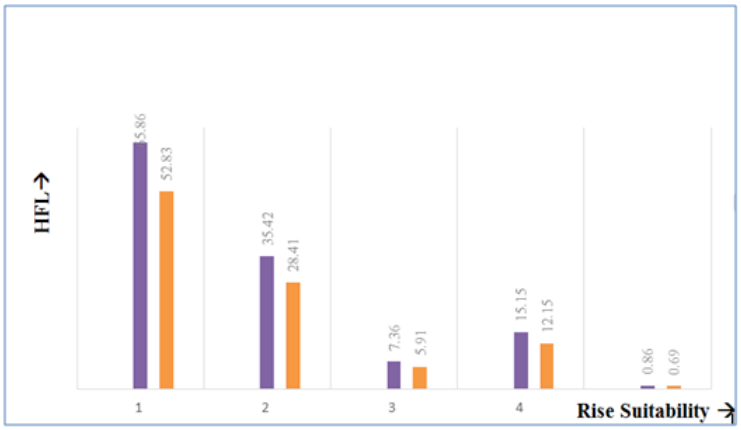


FIG. 5.4. *Sample productiveness LSI area underclasses by HFL.*

The appropriateness of the rice reasonableness is dissected utilizing HFL, with the alpha estimation of 0.6 showing the 60% vulnerability in the master learning about choosing the harvest appropriateness constraints and their prerequisites by the yield and the susceptibility over settling on their significance is connected is consolidated through the assurance list, lambda. At lambda = 0.578, rice is exceptionally appropriate over 53% of the entire territory accessible for refinement 28.79% of the region is under reasonable appropriateness, 6.16 % under minimal reasonableness.

5.3. Proportional Estimation of LSI. The consequences of the three land reasonableness methodologies are assessed here for their capacities to display land appropriateness assessment and tending to vulnerabilities engaged with it the below Fig. 5.5.

It is apparent from the consequences of every one of the three approaches that most of the zone is appropriate for the harvested rice. In any event, over 70.12% of the region is reasonable for rice development. Less of the territory is under lower appropriateness classes. It is seen from the outcomes that the perfect vector method

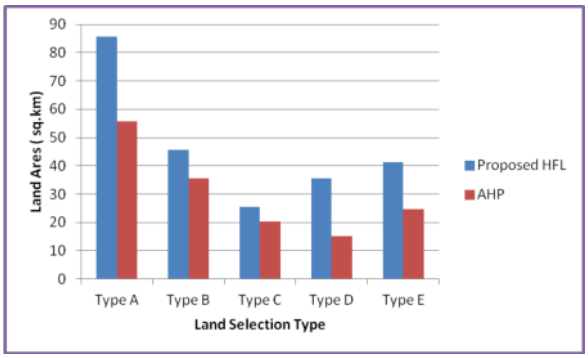


FIG. 5.5. *Performance comparison of Land Utilization.*

has some biases towards (−) and (+) ideal qualities (S_1, N_2). (+) definite overstated, and the contrary goals smothered, which is unreasonable is the reason is that the likeness file, determined from (+) and (−) goals, prompts higher scores of S_1 and lower ratings of N_2. The consequences of the AHP approach are reasonable. These outcomes are practically identical to that of the HFL. Even though AHP fuses master information, it neglects to combine the vulnerability engaged with the master learning, his judgment, and conclusions. HFL gives impressively excellent outcomes. The methodology joins the weakness of master assessments while looking at the criteria.

Moreover, this methodology provides a chance to join a gap that may emerge while communicating the inclination over these criteria. For instance, one can't express his preference for the waste over a surface with high conviction. One can express his feeling like waste is progressively liked to surface. The alpha cut and negative values utilized in the estimation of the fluffy exhibitions join the vulnerability of different sorts. Alpha cut consolidates the vulnerability in deciding the yield prerequisite extents. For instance, when the alpha worth 0.658 considered for pH, it thinks about the potential exhibitions between the range 0.0260: 0.457 for the class S1, which incorporates the qualities that may be scored by the class S_2 (0.0185 : 0.2518). From this, it very well may be deduced that the alpha cut capacity tends to the vulnerability associated with the info information (e.g., pH guide) and it likewise contemplates the weakness that may emerge from the meaning of as far as possible (S_1, S_2, S_3, N_1 and N_2). If the criteria estimated with more noteworthy vulnerability, at that point, quite possibly estimation of basis in a specific pixel may have a more extensive questionable range than one gauged with high vulnerability. Along these lines, the opinion of the alpha cut towards 0 demonstrates the higher vulnerability and thinks about the higher vulnerability with criteria. Those towards 1 speak to the assurance and have a tight scope of qualities. The esteem likewise measures weakness. Addresses the vulnerability that is associated with choosing the range of conditions gotten by the alpha cut. The worth will be towards 1 if the master or the leader is sure that the estimation of the foundation score is towards the most extreme evaluation of the unsure range. The worth will be towards 0 if the chief is increasingly sure that the estimate of the criteria score is towards the base estimation of the uncertain range.

5.3.1. Climate forecast of HFL. The climate forecast is assessed utilizing ten times cross approval, where 90.15% of information was being used for preparing, and 9.85% was being used for testing. Root means square of the distinction among genuine and anticipated qualities were found. Fig. 5.6 demonstrates the dispersion of average temperature blunder over every one of the areas. The legends speak to the scope of root mean square blunder rate while the pies speak to the level of locale falling under each range.

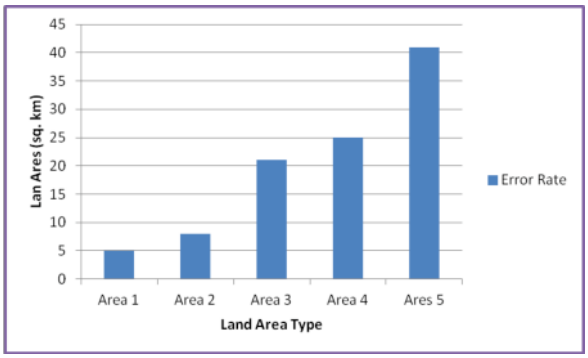


FIG. 5.6. *Demonstrates the diffusion of standard temperature error rate.*

The temperature consistently increments over time, and thus the root means square blunder worth is low than that of precipitation in the event of average temperature. If there should be an occurrence of rainfall, since precipitation sum won't wholly rely upon earlier year information, the blunder rate is high. Taking overcast spread, climatic weight can diminish the blunder rate for precipitation expectation.

5.3.2. Crop Forecast based on HFL. The harvests suggested have been contrasted, and the correct generation measurements dependent on zone and profitability in the locale. The Weightage given to the cost

of the autumn in the last proposal is distinguished by setting it with an extent of 1/4 and 1/3. The relating results acquired are depicted in the chart that appeared in Fig. 5.7.

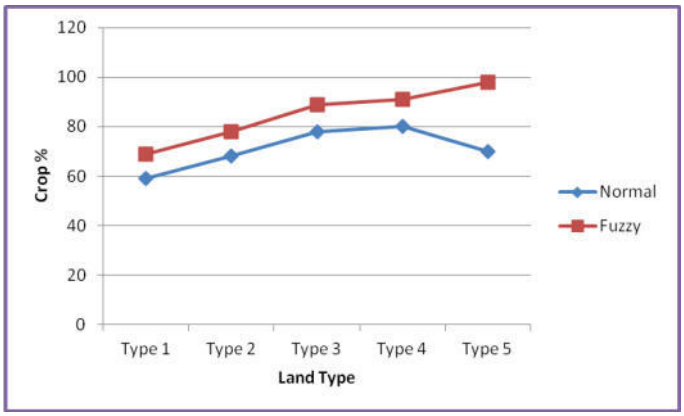


FIG. 5.7. *Crop Forecast based on HFL.*

A high Weightage for the expense isn't practical, as not every person can stand to deliver those yields. Consequently, authentic information additionally demonstrates less generation for those harvests. Henceforth the general review is low. The analysis tremendously influenced by the report of Rice harvest and Sugarcane. Indeed, even in a dry locale, for example, Ramanathapuram noteworthy information indicates the most astounding creation for rice, though the framework does not propose rice. It has caused a low review of 0.098, 0.1346 for the rice crop, for expense Weightage 1/4 and 1/3 separately. It demonstrates the ill-advised development in numerous locales. Correspondingly a few appropriate yields, for example, cashew nut, are not developed. Henceforth the approval with unique datasets influences a couple of explicit yields, which results in the general accuracy and review being controlled. The explanation behind a few yields not having a decent accuracy review is additionally because the help confirmations are a smaller value. It demonstrates that different yields may have smothered them during the fuzzy standard implication.

6. Conclusions and future work. Land reasonableness assessment is being completed without thinking about the vulnerability in the information, master learning. The land appropriateness assessment includes the criteria, which are in various scales extending from ostensible to proportion. Numerous contributions to the GIS-based land reasonableness assessment are the maps of the requirements, which are speaking to the perplexing, ceaseless, and unsure data in a primary ordered guide with the current limits among them. The Boolean procedures and other straightforward methods utilized for the land appropriateness assessment, which exasperates yields in the evaluation. To beat these issues, the present research investigates the probability of HFL. The objective of the examination is to expand the possibilities of the HFL into Land appropriateness necessary leadership. The aftereffects of execution assessment demonstrate an HFL of 65.554%.

This exploration has a great deal of extension for further improvements. The proficiency of agribusiness HFL can be improved utilizing all the more preparing information and standards, which will enable it to be used in cultivating procedure recovery calculation additionally, which will profoundly lessen even the requirement for measurable cleaning. Fluffy principles can be stretched out to think about past soil utility, and soil surface utilizing remote detecting on agrarian land, which will build the exactness. Increasingly horticultural parameters can be distinguished to be incorporated into the framework either in HFL or as a different module. The cross-sectional and top view pictures of soil can be handled to show signs of improvement thought regarding the dirt surface. The framework can likewise be incorporated with sensors that will give the daily report of soil and climate to help in system recommendation. As a sprouting space, there are as yet numerous prerequisites in horticulture that have not been investigated.

REFERENCES

[1] Y. Y. Jiang, H. X. Zhang, K. Meng, and J. Jie, *Research and application of multi-criteria decision-making method based on order relation and rough set*, Syst Eng Theory Pract., 27(6) (2007), pp. 161–165.
[2] S. Das, and S. Kar, *Group decision making in the medical system: an intuitionistic fuzzy soft set approach*, Appl. Soft Comput., 24 (2014), pp. 196–211.
[3] K. Gong, Z. Xiao, and X Zhang, *The bijective soft set with its operations*, Comput. Math Appl., 60(8) (2010), pp. 2270–2278.
[4] D. Stanujkic, B. Djordjevic, and M. Djordjevic, *Comparative analysis of some prominent MCDM methods: a case of ranking*, Serbian banks. Serbian J Manag, 8(2) (2013), pp. 213–241.
[5] P. A. Burrough, *Fuzzy mathematical methods for soil survey and land evaluation*, Journal of Soil Science, 40 (1989), pp. 477–492.
[6] T. R. Nisar Ahamed, K. Gopal Rao and J. S. R. Murthy, *GIS-based fuzzy membership model for crop-land suitability analysis*, Agricultural Systems, 63(2) (2000), pp. 75–95.
[7] FAO Soils Bulletin 32, *A framework for land evaluation*, 1981, Soil resources development and conservation service land and water development division, FAO and agriculture organization of the united nations, Rome 1976.
[8] H. Jiang, and J. R. Eastman, *Application of Fuzzy Measures in Multi-criteria evaluation in GIS*, International Journal of Geographic Information Science, 14(2) (2000), pp. 73–184.
[9] J. Malczewski, *GIS-Based Land Use Suitability Analysis: A Critical Overview*, Progress in Planning, 62(1) (2004), pp. 3–65.
[10] V. R. Thakare and H. M. Baradkar, *Fuzzy System for Maximum Yield from Crops*, Proceedings of National Level Technical Conference, (2013), pp. 4–9.
[11] S. C. Brown, P. J. Gregory, P. J. M. Cooper, and J. D. H. Keatinge, *Root and shoot growth and water use of chickpea (Cicer arietinum) grown in dryland conditions: effects of sowing date and genotype*, Journal of Agricultural Science, Cambridge, 113 (1989), pp. 41–49.
[12] F. Torrieri, and Batà, *A Spatial multi-criteria decision support system and strategic environmental assessment: A case study*, Buildings, 7(4) (2017), pp. 96.
[13] H. Kazemi, and H. Akinci, *A land-use suitability model for rainfed farming by multi-criteria decision making analysis (MCDA) and geographic information system (GIS)*, Ecological Engineering, 116 (2018), pp. 1–6.
[14] M. B. Mesgaran, K. Madani, H. Hashemi, and P. Azadi, *Iran's land suitability for agriculture*, Scientific Reports, 7(1) (2017), pp. 7670.
[15] Ingole, Kartik, et al., *Crop Prediction and Detection using Fuzzy Logic in Matlab*, International Journal of Advances in Engineering & Technology, 6.5 (2013), pp. 2006.
[16] Jawad, Fahim, et al., *Analysis of Optimum Crop Cultivation using Fuzzy System*, Computer and Information Science (ICIS), 2016 IEEE/ACIS 15th International Conference on. IEEE, 2016.
[17] J. Bosch, *From software product lines to software ecosystems*, Proceedings of the 13^{th}International software product line conference. Carnegie Mellon University, (2009), pp. 111–119.
[18] Bingen, Jim, Serrano, Alex, and Howard, Julie, *Linking farmers to markets: different approaches to human capital development*, Food Policy, 28(4) (2003), pp. 405–19.
[19] T. N. Prakash, *Land suitability analysis for agricultural crops: a fuzzy multi-criteria decision-making approach*, ITC, (2003, December).

Edited by: Rajkumar Rajasekaran
Received: Sep 19, 2019
Accepted: Apr 28, 2020


Scalable Computing: Practice and Experience, ISSN 1895-1767, http://www.scpe.org
© 2020 SCPE. Volume 21, Issues 4, pp. 583–589, DOI 10.12694:/scpe.v21i4.1699

A REVIEW ON THE ROLE OF MACHINE LEARNING IN AGRICULTURE

SYAMASUDHA VEERAGANDHAM* AND H SANTHI†

Abstract. Machine learning is a promising domain which is widely used now a days in the field of agriculture. The availability of manpower for agriculture is not enough and skill full farmers are less. Understanding the situation of the crop is not that much easy to detect and prevent the diseases in the crop. It is also widely employed in various agricultural fields such as topsoil management, yield management, water management, disease management and climate conditions. The machine learning models facilitate very fast and optimal decisions. The model of machine learning involves with training and testing to predict the accuracy of the result. The use of machine learning in agriculture helps to increase the productivity and better management on soil classification, disease detection, species management, water management, yield prediction, crop quality and weed detection. This article aims at providing detailed information on various machine learning approaches proposed in the past five years by emphasizing the advantage and disadvantages. It also compares different machine learning algorithms used in the modern agricultural field.

Key words: Machine Learning, Agriculture, Data Analysis, Training Methods and Sensors.

AMS subject classifications. 68T05

1. Introduction.

Now a days, agriculture is the major source in all over the world, it plays a vital role in the global economy. Many of the research projects and funding projects are continuously implement with the latest technologies. Increasing the yield of the crop, automating the work done for the crop, minimizing the manpower, reducing the disease ratio by detecting on initial stages with existing matched patterns and finally harvesting with machinery in minimum time [1]. Machine learning in agriculture implemented with different crops from sowing to harvesting using many techniques from different technologies like big data, artificial intelligence, drones and data mining. Using these technologies, for mapping with existing data to identify and fix the solution of frequent problems. And defining the machine with related scenario based on the weather conditions and the moisture position using sensors detectors.

Agriculture in normal processing with manpower may have many problems with less expertise for knowing the crop complete processing, disease identification, appropriate pesticides usage on initial stages instead of spreading on whole crop, huge manpower utilization from sowing to harvesting of the crop. There may be many consequences to know everything about the crop and all types of crop manually, without having any experience. So, the new people working on the crop have difficulties in knowing everything. Mainly the cost of the crop is very high because of utilizing everything on manpower. Even now a days many machineries are using in cultivating crops in agriculture and also manpower is needed to form sowing to harvesting, but many crop automation is not possible because of the crops depends on environment conditions and soil management [1].

The Machine learning algorithms are processed on deep learning and artificial intelligence algorithms based implementations. Train the system with deep learning and automated with artificial intelligence combination mechanism. Using machine learning in agriculture trained the system with deep learning mechanism inbuilt with sensors intermediate for collecting and processing the data. Once the current status data is processed by the sensor then the machine understands the process of deep learning and also mapping with artificial intelligence. Machine learning in agriculture mainly depends on the existing dataset mapping with new data set processed by the sensor. Deep learning in agriculture also is a trend nowadays. Using the latest technologies on image processing with data analysis of existing models which matches with the results and its excellent outcome.

*Research scholar, School of Computer Science and Engineering, Vellore Institute of Technology, Vellore, Tamilnadu, India (syamasudha.veera2019@vitstudent.ac.in).

†Correspondence author: Associate Professor, School of Computer Science and Engineering, Vellore Institute of Technology, Vellore, Tamilnadu, India (hsanthi@vit.ac.in).

These days we can overcome many problems and challenges on different agriculture fields and food production using deep learning mechanism. Deep learning used on different crop fields with natural language processing, speech recognition with learning, sensors to detect temperature on soil based on the crop and weather condition from sowing to harvesting of the crop [2].

The usage of machine learning helps farmers to collect the information and data by using it in agriculture with the aid of information technology to make the best decisions on high output from the farms [3]. Machine learning algorithms can be used for many applications in agriculture; like crop suggestions based on the pest detection in plants, soil fertility, weed detection, yield cultivation of crops and plant disease detection based on the disease identification on early stages will recovery the plant so that automatically increase the crop. It is very important to minimize the utilization of pesticides as considering food quality and health measures of the people. Apart from monitoring environmental conditions on a farm, intelligent agriculture must analyse understand how weather circumstances cause environmental changes at the farm and how long-term crop cultivation brings about soil erosion or changes in the soil structure. Through minimizing water management we can save the water instead of wastage and use the same water for another crop. Monitoring a farm using machine learning can prevent low productivity of crops [4]. Using crop management improves the yield of the crop can take care from sowing to harvesting.

2. Literature survey. In agriculture, using machine learning indicates using many crops. A previous study [5] explains the harvesting in date fruit orchard using robotics and Deep Learning mechanism. There are two pre-learning CNN mechanisms; namely, AlexNet and VGG-16. To construct a study machine imaginative and prescient system, also based on the high-quality image dataset of 5 data types for all maturity stages. The suggested method accomplish extremely good classification based on the difficult dataset with matching ration. The high-quality pixel images data sets are used in future to improve mapping of different date fruit orchards.

In another study [6] conducted on the graphical representation of modelling, primarily with the references of sparse linear additive and proposed processes to discover a sparse in part linear additive shape on development of directed open-chain graphical representation. The study updates the outcomes as a case study the use of variable dataset accumulated on the runtime of the plant manufacturing and also proposed view is tremendous for discovering strengthened variable or fixed based on temporary with proper output graph. They proposed a method regarding the energy-efficient improvement management. Anyhow, the new approach that the team invented predicted to be applicable for various surveys which are primarily based on the statistical methods. Here, they mainly focused on the classification of normal additive samples, but their scheme can be without difficulty, moved further with elaboration and naturally through involving arbitrary link models. Here they assumed the hill mountaineering approach for getting associated best results inside the fixed period.

Machine learning is famous with its ability to achieve maximum on many domain-based technologies. Machine learning can be regarded the top-rated analytic tool for fog computing applications. Instead of modern day's achievements, machine learning purposes and literature also plays a major role. The latest research gap defines fog computing. The research achievement of machine learning in three elements are resource management, accuracy and security. Machine learning is implementing mainly on resource management instead of accuracy and security in fog computing. Machine learning include cloud computing in one of the layers. Even many problems and challenges have been open-ended with these combinations. Even most of the challenging problems regarding safety measures used in cloud computing. Supervised learning in fog computing is one of the famous machine learning assignment. In the healthcare domain most of the applications have utilized machine learning and also many open challenges and issues are there in fog computing in machine learning [7].

A prior study [8] has planned a correct and strong algorithm for a new mechanism to critically find the growth of cucumber using robotic harvesting automated process in agriculture. This algorithm is a different sort of implementations and mining methodologies of existing data to gain of cucumber field with extraordinary components. This mechanism combines vector elements with image pixels match to get the starting stage itself on next level onwards. Many outcomes of the yield were taken as the feedback as input for modelling and testing the final algorithm. This algorithm outcome of the application is more efficient for harvesting the cucumber.

Deep Learning resolves many diseases from detecting with proper data set and using pesticides with minimal quantity for curing the plants of the crop [9]. Another study [10] explains crop prediction, implementing many techniques including artificial neural networks. Based on the artificial intelligence finding the soil state

depending on the weather condition with maximizing the yield of the crop. Later, a study [11] referred around 40 research papers defined as a survey on all the aspects of machine learning in agriculture. The reference papers appeared in standard journals with high cited papers and many are the reputed papers with implemented on all over the world on machine learning in agriculture. Machine learning in agriculture using sensor data on the agricultural field using artificial intelligence with high suggestions from sowing to harvesting.

3. Machine Learning In Agriculture. The latest trend in agriculture using machine learning are implementing smart farming with latest techniques. Many recent mechanisms are using for farming to find soil moisture based on the type of crop with water management, disease detection and selection of pesticide with existing patterns, crop management. Machine learning consists of five major components as shown in Figure 3.1. These five major components are 1). Collecting data from the farm 2). Stored data 3). Data pre-processing 4). Train the model and 5). Performance metrics.

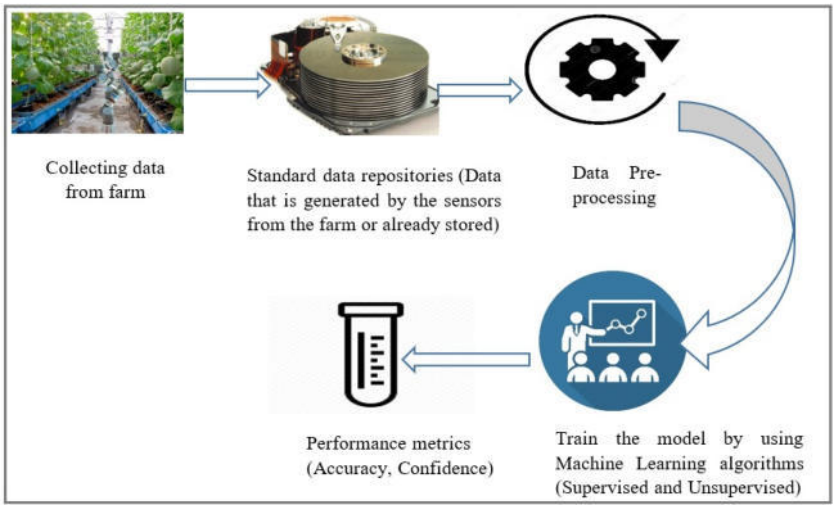


FIG. 3.1. *General process of machine learning*

3.1. Collecting data from the farm. Most of the researchers used different sensors such as temperature sensor, passive Infrared sensor, ultraviolet sensor, pH sensor, soil moisture sensor, humidity sensor, gas sensor, barometric Pressure, hyperspectral camera, multispectral camera, DSLR camera, NPK sensor, obstacle sensor, acoustic sensor, water level sensor, water quality Sensor, GPS sensor and MooMonitor sensor to collect the data about the soil fertility, yield of crops, climate condition, pest detection in plants, weed detection, identifying diseases, etc. This data gives more accurate results, but it is cost-oriented, time-consuming and also more difficult to collect. A study [12] developed a robot with different sensors and it is collected various environmental factors that effects on soil such as soil temperature, pH value in soil, the intensity of sunlight, soil moisture, humidity etc. to grow a plant. Later, another study [13] proposed and designed an optical transducer to measure soil nutrients such as Nitrogen, Phosphorus and Potassium, and these nutrients are called primary nutrients. Soil nutrient not only depends on primary nutrients and also having macronutrients and micronutrients [14]. Generally, the soil fertility and environmental factors both are interrelated.

Another study [15] developed a smart irrigation system for agriculture by using temperature, humidity and UV sensors. Soil fertility, types of crop and climate condition are the essential parameters for designing a smart irrigation system. So, while collecting soil nutrients, also collect environmental factors and type of crop, it will generate more accurate results for decisions from the data analysis. Hyperspectral Image of different crops by using hyperspectral cameras for crop classification have used in another study [16]. Later a study [17] has used the acoustic sensor for detecting insect's pests in the underground. This sensor is mostly useful for detecting insects in underground crops like carrot, potato, groundnuts, onion, taro, turmeric, garlic, etc. A research collected [18] 83,260 and 16,652 colour images format as JPEG from wheat planting of Shandong

Province, China, to train and test a model respectively for winter wheat leaf diseases by using Canon EOS 80D camera.

3.2. Standard data repositories. The main benefit of the existing data is the researcher doesn't want to spend money on collecting data. This data takes less time and also easy to collect, but it gives the less accurate results [19]. This data is especially available in various websites. The researchers can download from the websites and performs the data analysis. The forms of data include images, tables, text, graphics, audios and videos. The following industries are mainly providing datasets for analysing the data by using machine learning algorithms in agriculture. UCI, MIT, Kaggle, University of Arkansas, Live tree, China Agro. and Econ. Data, Open Government Data, OpenAg, GitHub, Data. world, Knoema, USDA-ARS, V2 Plant Seedlings Dataset, Food and agriculture data, Pesticide Use in Agriculture and Soil Resources Development Institute are providing data to the data analysis.

A study [20] used soil related data of upazila of Khulna district in Bangladesh for soil classification and crop suggestion, collected data from soil resources development institute, bangladesh. Another study [21] are collected from 3010 images of rice plants with diseases from the high-standard rice experimental field of the hunan rice research institute in China to detect the rice plant diseases. Another study [18] collected eight categories of 16652 images from Shandong Province, China to identify 8 different diseases from the wheat winter crops. And another study [22] collected 1070 real-time mango tree leaves images from shri mata vaishno devi university in the district of Katra located in Jammu and kashmir, India to identify the fungal disease named as anthracnose. This method is less expensive, easier to collect and cost-effective but the error rate is high and not suitable for all areas.

3.3. Data pre-processing. The major issues in real-time data are inconsistent, duplication, noise and missing values. This type of dataset is very critical for analysing the process and increases the error rate. Data pre-processing performs on the raw data for further processing to enhance the quality of the data. Data pre-processing is major crucial step in machine learning to improve efficiency while data processing. Pre-processing can removing the noise, inserts the missing values, the appropriate data range, and extracting the functionality etc. A study [21] used Two-Dimensional Filter Mask Combined with Weighted Multistage Median Filter (2DFM-AMMF) to remove the noise from the rice plant images. Majority of image related works are used segmentation and feature enhancement to improve the quality in data. Another study [23] removed salt and pepper noise by using Gaussian Median and Gaussian filter respectively to enhance the image quality to the 4-different crops and 2-weeds namely Paragrass and Nutsedge are chosen for classification. And another study [24] used Principle Component Analysis to remove the dimensionality and multicollinearity problems for water supply forecasting in the US West.

3.4. Train the model by using machine learning algorithms. Machine Learning algorithms are classified into two types; Supervised Learning and Unsupervised learning. The use of machine learning in agriculture helps to a). Soil classification b). Disease Detection c). Species management d). Water Management e). Yield Prediction f). Crop Quality g). Weed Detection.

3.4.1. Soil classification. Soil is classified based on its strength and property, it can be helps to grow the crop. Former uses the soil classification system for predicting the soil behaviour. Based on the chemical and physical properties of the layers of soil, soil can be classified and named. Soil classification can be used to identify the best crops and type of fertilizer based on the type of soil. Climate changes also plays a major role in soil management with water management. Using machine learning techniques, suggests associated procedures, moisture techniques concerning the temperature. A study[20] used Machine learning algorithms such as Gaussian kernel-based Support Vector Machines (SVM), k-Nearest Neighbour (k-NN), and Bagged Trees are used for soil classification, but proposed Gaussian kernel-based Support Vector Machines (SVM) based method performs better than the k-Nearest Neighbour (k-NN), and Bagged Trees.

3.4.2. Disease Detection. Disease mainly depends on the weather and climatic conditions, soil characteristics and plant strength. Detection of disease based on climatic conditions and viral diseases are broadly pest and managing of disease utilization simultaneously. Disease detection also compares with a dataset as per the weather conditions and age of the plant. A study [21] compared two machine learning algorithms

such as combination of Fuzzy C-means and K-mean clustering(FCM-KM) + faster Region Convolutional Neural Network(R-CNN) and faster Region Convolutional Neural Network(R-CNN) for detecting the rice plant disease such as rice blast, bacterial blight and sheath blight, but proposed combined of Fuzzy C-means and K-mean clustering(FCM-KM) + faster Region Convolutional Neural Network(R-CNN) performs better than the faster Region Convolutional Neural Network(R-CNN). This method is not suitable for large scale rice plant disease detection. A another study [3] compared 5-different machine learning algorithms such as Support Vector Machines (SVM), Artificial Neural Networks (ANN), K-Nearest Neighbour (KNN), Fuzzy and Conventional Neural Networks (CNN) for identification of different diseases in different crops. Among these 5-machine learning algorithms, Conventional Neural Network classification is given more accurate results and also suitable for more crops. And another study [18] used Machine Learning algorithm such as Matrix-based Convolutional Neural Network (M-bCNN) for winter wheat leaf diseases by classifying 8 different leaf diseases such as normal leaf, mechanical damage leaf, powdery mildew, cochliobolus heterostrophus, bacterial leaf streak, bacterial leaf blight, leaf rust and stripe rust.

3.4.3. Species management. Species Breeding: The selection of species is an important mechanism, based on the soil along with region weather conditions and water associated vitamins with good taste. Using deep learning procedures existing patterns, data sets are mapped for solving the many challenges instead of assumptions. A study [25] used Cascaded Support Vector Machine algorithm for classifying 9-different sunflower seeds.

Species Recognition: Manual selection of plants can be based on the leaves colour, age and shape of the plant. The selection of plant will be the first step of the crop and the roots of the plant concerning the age and veins colour on the leaves. A study [26] used 3- different classifiers like Color-Shape-Texture, Pixel and SIFT based to classify 5 categories of species like Flowers, Fruit, Leaf, LeafScan and Stem. A another study [16] used combination of Feature Band Set (FBS) and Object Oriented Classification (OOC) to classify different crops using Hyperspectral Images.

3.4.4. Water Management. Water management plays a major role in every crop. Using a machine learning mechanism we can efficiently use the water so that excess water will be used for another crop. Based on the crop and soil type we can provide the water daily, weekly and monthly. A study [27] used soil temperature, moisture and pH sensors to find the soil water content level in automatic water dripping system for agriculture. The main benefit of an automatic irrigation system is to correct water usage, power and time saving but it consumes more money from farmers.

3.4.5. Yield Prediction. Every crop we suppose to concentrate as per yield prediction. Yield prediction defines mainly mapping of yield, demand based on the crop outcome and evaluation. The yield prediction can be defined from the earlier dataset and what type of latest technologies available and applicable on all the ways based on the current crop, climatic and financial situations for improving the yield. A study [28] used Tensor Flow with Convolutional Neural Networks and Linear Regression for estimating the yield from Sorghum field.

3.4.6. Crop Quality. Crop quality is the way to finalize the crop outcome in the form of financial. Based on the final quantity of yield, minimal wastage and the quality of the crop after harvesting can be detected. As per these parameters, we can define the crop quality and also compare with the dataset. A study [29] connected different sensors to the drone to monitor the crop quality. This drone can monitor the crop, gives alert to the farmer when any issues identify on the crop.

3.4.7. Weed Detection. Weed detection is the main problem on every crop, based on this the final yield defines. It is a very important threat on the crop that effects on the yield. Concerning the age of the plant and the weed, the stage needs to detect. Once we minimize the weed then only the yield will be good otherwise the fertilizers and the pesticides also not working on the crop. The work of the weed is to eat the whole energy of the soil. So the crop quantity will be by default minimum. Machine learning has many mechanisms to detect the type of weed on every crop and intimate. A study [30] used to Support Vector Machine and Conventional Neural Network for detecting Broad-leaf weed detection in the pasture from the images. Another study [23] used SVM, ANN and CNN for classifying the 4- different crops and 2- different weeds, but CNN gives better results compared with the remaining methods.

3.5. Performance metrics. The best method is decided by using the accuracy. The method which has the highest accuracy is the best. The more accurate models can give the better decisions as an outcome. The following Table 3.1 shows different machine learning algorithms used in agriculture and with its accuracy.

TABLE 3.1
Comparison of different machine learning algorithms are used in agriculture.

S.No	Subdomain	Ref.No	Crop	Algorithm	Accuracy
1.	Soil Classification	[20]	—	J48	92.30
				Support Vector Machines	94.95
2.	Disease Detection	[21]	Paddy Diseases	Faster R-CNN	91.28
				FCM-KM+ Faster R-CNN	97.50
3.	Disease Detection	[18]	Wheat Diseases	Matrix-based CNN	96.50
4.	Species Breeding	[25]	Sunflower Seeds	Support Vector Machines	98.82
5.	Yield Prediction	[28]	Sorghum field	CNN and Linear Regression	74.50
6.	Weed Detection	[30]	Pasture	Support Vector Machine	89.40
				CNN	96.88
7.	Species Recognition	[26]	5-Species	SIFT based	98.00
8.	Weed Detection	[23]	4-crops,2-weeds	SVM, ANN and CNN	CNN best
9.	Disease Detection	[3]	6-crops	SVM	92.31
			2-crops	ANN	93.70
			2-crops	KNN	88.75
			1-crop	FNN	88.00
			4-crops	CNN	98.62

4. Conclusion. Machine learning is widely used in modern agriculture. In addition to the Machine learning, deep learning, artificial intelligence and robotics using much more for minimizing the manpower and manual mistakes and almost everything needs to do automation form sowing to harvesting the crops. Using the latest technologies and mechanisms for minimizing the manual mistakes for detecting the type of crop to pesticides selection concerning the dataset mapped. The use of machine learning in agriculture helps in the different sub areas like soil classification, disease detection, species management, water management, yield prediction, crop quality and weed detection processing also implanting with machine learning. In this paper our focus is to provide detailed survey about how various machine learning algorithms were used in different fields of modern agriculture. This paper provides a detailed comprehensive comparative analysis of various machine learning algorithms.

REFERENCES

[1] SEVEN REASONS WHY MACHINE LEARNING IS A GAME CHANGER FOR AGRICULTURE, *Available in:* https://towardsdatascience.com/7-reasons-why-machine-learning-is-a-game-changer-for-agriculture-1753dc56e310.

[2] WHAT IS ML AND WHY DO FARMING ENTREPRENEURS CARE, *Available in:* https://medium.com/sciforce/machine-learning-in-agriculture-applications-and-techniques-6ab501f4d1b5.

[3] SHRUTHI, U., NAGAVENI, V., & RAGHAVENDRA, B. K. (2019, MARCH). *A review on machine learning classification techniques for plant disease detection. In 2019 5th International Conference on Advanced Computing and Communication Systems (ICACCS) (pp. 281-284). IEEE.*

[4] TSENG, F. H., CHO, H. H., & WU, H. T. (2019). *Applying big data for intelligent agriculture-based crop selection analysis. IEEE Access, 7, 116965-116974.*

[5] ALTAHERI, H., ALSULAIMAN, M., & MUHAMMAD, G. (2019). *Date fruit classification for robotic harvesting in a natural environment using deep learning. IEEE Access, 7, 117115-117133.*

[6] Fujimoto, Y., Murakami, S., Kaneko, N., Fuchikami, H., Hattori, T., & Hayashi, Y. (2019). *Machine Learning Approach for Graphical Model-Based Analysis of Energy-Aware Growth Control in Plant Factories. IEEE Access, 7, 32183-32196.*

[7] Abdulkareem, K. H., Mohammed, M. A., Gunasekaran, S. S., Al-Mhiqani, M. N., Mutlag, A. A., Mostafa, S. A., et al. (2019). *A Review of Fog Computing and Machine Learning: Concepts, Applications, Challenges, and Open Issues. IEEE Access, 7, 153123-153140.*

[8] Fernandez, R., Montes, H., Surdilovic, J., Surdilovic, D., Gonzalez-De-Santos, P., & Armada, M. (2018). *Automatic detection of field-grown cucumbers for robotic harvesting. IEEE Access, 6, 35512-35527.*

[9] Yang, X., & Sun, M. (2019, April). *A Survey on Deep Learning in Crop Planting. In IOP Conference Series: Materials Science and Engineering (Vol. 490, No. 6, p. 062053). IOP Publishing.*

[10] K, Sriram. (2019). , *A Survey on Crop Prediction using Machine Learning Approach. International Journal for Research in Applied Science and Engineering Technology. 7. 3231-3234. 10.22214/ijraset.2019.4542.*

[11] Liakos, K. G., Busato, P., Moshou, D., Pearson, S., & Bochtis, D. (2018). *Machine learning in agriculture: A review. Sensors, 18(8), 2674.*

[12] Krishna, K. L., Silver, O., Malende, W. F., & Anuradha, K. (2017, February). *Internet of Things application for implementation of smart agriculture system. In 2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC) (pp. 54-59). IEEE.*

[13] Masrie, M., Rosman, M. S. A., Sam, R., & Janin, Z. (2017, November). *Detection of nitrogen, phosphorus, and potassium (NPK) nutrients of soil using optical transducer. In 2017 IEEE 4th International Conference on Smart Instrumentation, Measurement and Application (ICSIMA) (pp. 1-4). IEEE.*

[14] About the soil nutrients, *available in:* https://emeraldlawnsaustin.com/macronutrients-micronutrients-soil/.

[15] Goap, A., Sharma, D., Shukla, A. K., & Krishna, C. R. (2018). *An IoT based smart irrigation management system using Machine learning and open source technologies. Computers and electronics in agriculture, 155, 41-49.*

[16] Zhang, X., Sun, Y., Shang, K., Zhang, L., & Wang, S. (2016). *Crop classification based on feature band set construction and object-oriented approach using hyperspectral images. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 9(9), 4117-4128.*

[17] Bayrakdar, M. E. (2019). *A Smart Insect Pest Detection Technique With Qualified Underground Wireless Sensor Nodes for Precision Agriculture. IEEE Sensors Journal, 19(22), 10892-10897.*

[18] Lin, Z., Mu, S., Huang, F., Mateen, K. A., Wang, M., Gao, W., & Jia, J. (2019). *A unified matrix-based convolutional neural network for fine-grained image classification of wheat leaf diseases. IEEE Access, 7, 11570-11590.*

[19] Farooq, M. S., Riaz, S., Abid, A., Abid, K., & Naeem, M. A. (2019). *A Survey on the Role of IoT in Agriculture for the Implementation of Smart Farming. IEEE Access, 7, 156237-156271.*

[20] Rahman, S. A. Z., Mitra, K. C., & Islam, S. M. (2018, December). *Soil classification using machine learning methods and crop suggestion based on soil series. In 2018 21st International Conference of Computer and Information Technology (ICCIT) (pp. 1-4). IEEE.*

[21] Zhou, G., Zhang, W., Chen, A., He, M., & Ma, X. (2019). *Rapid Detection of Rice Disease Based on FCM-KM and Faster R-CNN Fusion. IEEE Access, 7, 143190-143206.*

[22] Singh, U. P., Chouhan, S. S., Jain, S., & Jain, S. (2019). *Multilayer convolution neural network for the classification of mango leaves infected by anthracnose disease. IEEE Access, 7, 43721-43729.*

[23] Sarvini, T., Sneha, T., GS, S. G., Sushmitha, S., & Kumaraswamy, R. (2019, April). *Performance Comparison of Weed Detection Algorithms. In 2019 International Conference on Communication and Signal Processing (ICCSP) (pp. 0843-0847). IEEE.*

[24] Fleming, S. W., & Goodbody, A. G. (2019). *A Machine Learning Metasystem for Robust Probabilistic Nonlinear Regression-Based Forecasting of Seasonal Water Availability in the US West. IEEE Access, 7, 119943-119964.*

[25] JayaBrindha, G., & Subbu, E. G. (2017). *Ant colony technique for optimizing the order of cascaded SVM classifier for sunflower seed classification. IEEE Transactions on Emerging Topics in Computational Intelligence, 2(1), 78-88.*

[26] Purohit, S., Viroja, R., Gandhi, S., & Chaudhary, N. (2015, December). *Automatic plant species recognition technique using machine learning approaches. In 2015 International Conference on Computing and Network Communications (CoCoNet) (pp. 710-719). IEEE.*

[27] Padalalu, P., Mahajan, S., Dabir, K., Mitkar, S., & Javale, D. (2017, April). *Smart water dripping system for agriculture/farming. In 2017 2nd International Conference for Convergence in Technology (I2CT) (pp. 659-662). IEEE.*

[28] Zannou, J. G. N., & Houndji, V. R. (2019, April). *Sorghum Yield Prediction using Machine Learning. In 2019 3rd International Conference on Bio-engineering for Smart Technologies (BioSMART) (pp. 1-4). IEEE.*

[29] Saha, A. K., Saha, J., Ray, R., Sircar, S., Dutta, S., Chattopadhyay, S. P., & Saha, H. N. (2018, January). *IOT-based drone for improvement of crop quality in agricultural field. In 2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC) (pp. 612-615). IEEE.*

[30] Zhang, W., Hansen, M. F., Volonakis, T. N., Smith, M., Smith, L., Wilson, J., ... & Wright, G. (2018, June). *Broad-leaf weed detection in pasture. In 2018 IEEE 3rd International Conference on Image, Vision and Computing (ICIVC) (pp. 101-105). IEEE.*

Edited by: Rajkumar Rajasekaran
Received: Feb 21, 2020
Accepted: Apr 2, 2020

Scalable Computing: Practice and Experience, ISSN 1895-1767, http://www.scpe.org
© 2020 SCPE. Volume 21, Issues 4, pp. 591–599, DOI 10.12694:/scpe.v21i4.1714

PRINCIPLES AND PRACTICES OF MAKING AGRICULTURE SUSTAINABLE: CROP YIELD PREDICTION USING RANDOM FOREST

SYED MUZAMIL BASHA*, DHARMENDRA SINGH RAJPUT†, J JANET‡, SOMULA RAMASUBBAREDDY§, AND SAJEEV RAM¶

Abstract. Agriculture has advanced tremendously over the last 100 years. In fact it is been keeping up with food production at a very high rate. In fact, some scientists feel that agriculture already produces enough food to feed the world, but of course there are issues and problems with food availability, agricultural production practices, preservation and transportation, and probably more that one can think of that hinder many people in this world from getting adequate food. The basic challenge is to provide food for the needy people. This need can be fulfilled with the help of the farmers taking responsibility in increasing the food production by 50% by the year 2050. The objective of the present work is to increase this food production, protecting the environment with managing natural resources. Mainly focusing on water, nutrients and other inputs to produce foods without degrading the environment. The Goal is to develop the social, environmental, and the economic aspects of possible solutions to minimize the agricultural footprint, and become more sustainable. The dataset considered in our experiment is used in yield prediction based on historic yield and weather information. Implemented both the versions of Thomson model and compared the result with segmentation model, Random Forest (RF). Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) are used as evaluation metrics in estimating the performance of models implements and stated that Random forest algorithm is providing 0.07(RMSE). The outcome of the present research work helps farmers in adopting best management practices and trying to give them the economical and technical support in making easier for them to adopt best management practices.

Key words: Food production, Sustainability, Thomson model, segmentation model, Random Forest, Root Mean Square Error, Mean Absolute Error.

AMS subject classifications. 97R40

1. Introduction. Farmers do indeed feed the world. Indian farmer is said to produce enough food for his or her family and for at least 150 other people in the country. Researchers estimates the need to increase food production by 50 to 70% along with maintaining sustainably. Getting pressure on agriculture on farmers to increase this food production and at the same time make sure that they're not doing any harm to the environment. More focus will be on the factors like nutrients, nitrogen and phosphorus and water quality. Increasing crop productivity through breeding and genetics, many such aspects has come from research and development that allow agriculture to persist and be efficient. The outcome of the proposal is to help farmers to adapt management strategies that will protect the environment. More scientists are focusing now on nutrient mass budgets for farms. In which the following questions can be answered. Agriculture has advanced tremendously over the last 100 years. In fact it's been keeping up with food production at a very high rate. In fact, some, some scientists feel that agriculture already produces enough food to feed the world, but of course, but of course there are issues and problems with food availability, food, low food or agricultural production practices, preservation and transportation, and probably more that one can think of that hinder many people in this world from getting adequate food. Getting pressure on agriculture on farmers to increase this food production and at the same time make sure that they're not doing any harm to the environment. To focus on nutrients, nitrogen and phosphorus and water quality. An attempt was made to improve nutrients in the form through

*Associate professor, Sri Krishna College of Engineering and Technology, Coimbatore-641008, India (muzamilbashas@skcet.ac.in)
†Associate professor, SCOPE, VIT, India (dharmendrasingh@vit.ac.in)
‡Professor, Sri Krishna College of Engineering and Technology, Coimbatore-641008, India (janet.fredrick@gmail.com)
§Assistant Professor, Department of Information Technology, Vallurupalli Nageswara Rao Vignana Jyothi Institute of Engineering &Technology, Hyderabad, India, (svramasubbareddy1219@gmail.com)
¶Associate professor, Sri Krishna College of Engineering and Technology, Coimbatore-641008, India (sajeevrama@skcet.ac.in)

Bio fortification process, which are useful to human beings. Also, deficiency of micro nutrients in plants is addressed using plant breeding works [17].

1.1. Challenges in making farmers to adopt new kinds of practices.

- To get water out of a river and raise it up and put it in a canal to help irrigate their crops in between floods, flood seasons.
- Land Management
- Competition for good form land
- Harvesting techniques have grown to include very, large, efficient machines to harvest large areas of produce.
- Irrigation is a very important aspect to sustainable crop production causing the leeching of nutrients.
- pest management, new varieties of crop plants have all played a role
- protecting soil and minimizing nutrient loses from farms.

Competition with urbanization will remove good agriculture land. Available productive land has grown only by 8% while food production doubled between 1967 and 2017. Land availability is going to be a challenge for producers. Obviously farmers compete with urban development and expansion. And so, the value of the land for those alternative uses is going to be very important. The average age of farmers in the India is approaching 60 years. And so often wonder how many farms will stay as family farms under the ownership as farmers become older. What happens to those farms in the years as farmers approach retirement. While the availability land, available availability of good agriculture production land has only increased about 8 to 10% globally.

1.2. Sustainable Development Goals. The Sustainable Development Goals (SDGs) are a part of the 2030 agenda. This makes the SDGs relevant for every person, country, and company on Earth. The total interconnection between the goals and the necessity to move forward on all goals at the same time. But of course, there are companies who have special qualifications and special issues. Business community now arguing for higher standards and higher national ambitions in order to push forward the technologies and products. The overarching goal of the SDGs taken as a whole is human welfare. That is to improvement of the human condition. Indeed, the first eight goals, no poverty, zero hunger, good health and well-being, quality education, gender equality, clean water and sanitation, clean affordable energy, and decent work and economic growth, deal directly with humans and their immediate condition. The remaining nine goals address the living and non-living infrastructure or resources and social structures that one way or another, provide support for the maintenance and improvement of the human condition. Achieving any one of these 8 goals will have positive effects on most. All of these welfare goals are however also clearly interlinked with one or more of goals 9 to 17, which focus on environment and infrastructure issues. Climate change, goal 13 interacts with goal 1, poverty, as it is generally speaking, the poorest that are most impacted by climate change. Climate change makes it harder to escape poverty. Similarly, inequality plays a role in SDG 3, access to healthcare, SDG 4 access to education, SDG 5 gender equality, SDG 7 access to energy, SDG 8 access to decent work, and so on. Addressing inequality is however so essential to achieving sustainable development that inequality has earned an SDG in its own right. SDG 10, focuses on reducing inequality in all of its forms but where many of the indicators used to assess progress against the goal deal with income inequality. While SDG 10 focuses primarily on income inequality, which is the difference in annual income received by individuals or countries per year. The research happened on sustainable indicator and its advantages are presented in Table 1.1.

However, the direct relationship between income or total wealth and well-being breaks down at higher levels of income. In other words, income or wealth are not in themselves adequate metrics for assessing societal development or human well-being. This is, of course, one of the reasons why a target in SDG 17 is to develop metrics that can supplement traditional economic metrics for assessing societal development. Wealth inequality is largely driven by the unequal ownership of capital in its many forms, not just financial capital, but also human, natural, physical, and social capital all of which contribute to human opportunity and well-being. The role of business in the world achieving the SDGs are classified as three key roles is: First of course business has to be responsible, make sure that our own operations and our supply chain up their performance on all the important areas, from environmental issues to people related issues. The second is to innovate and deliver solutions to environmental or social problems. So that one can deliver a positive contribution to the world,

TABLE 1.1
Sustainable Indicators

Author Details	Problem Addressed	Names of the Indicators	Future Scope
[9]	Introduced monitoring tool on PRIMA research	Multidimensionality poverty Index, Agriculture value added, Crop water productivity, Amount of agriculture residuals used for energy purpose.	Make use of Technology in traditional agriculture practices.
[10]	Designed a conceptual framework for selecting appropriate indicators	Multidimensionality poverty Index, Agriculture value added, Crop water productivity	Need of integrating agriculture and policy maker for better decision making.

with high impact, at speed and scale.

The contributions made in the present work are:

1. Literature review on sustainable indicators and improvement in yield responses.
2. Implements both the versions of Thomson's model and compared the result with the segmentation, Random forest model.
3. Analyzed the model performance with respect to yield responses
4. Error (RMSE and MAE) in prediction yield is estimated and proved that Random forest algorithm is provided good results.

The organization of the present work is as follow: In Introduction, the challenges being faced in adopting sustainable agriculture is addressed along with the importance given to agriculture in achieving goals and indicators used. In Literature section, the past research work carried out in improving the methods of agriculture protecting the environment is discussed. In Methodology section, the work carried out in the present research work is described in detail. In Result and discussion, the experimental details and discussion on result obtained is described with comparison. Finally, the conclusion and future work section, describe the objective addressed in the present work and the steps followed in achieving the goal of the work as discussed in introduction section.

2. Literature Study. Farmers understand that protecting the soil and conserving is critical to the success of the farm. So, soil conservation practices are extremely critical in that regard. sustainability and particularly sustainable agriculture can held with attention to both water quality and nutrients. Farmer need both of those inputs nutrients and chemicals that might impact water quality. The organic material on the immediate surface of the earth that serves as a natural medium for the growth of land plants [1]. The role of soil in environmental quality is high. Soil properties affecting water management (texture, hydraulic conductivity, water-holding capacity and natural drainage) [2] and nutrient management (organic matter content, soil PH, Cation exchange capacity and coatings on sand grains) [3]. Soil texture can be defined as relative proportion of sand, silt and clay in a mineral soil. Texture affects the amounts of water and nutrients a soil can hold [4]. Soil organic matter includes anything that was once alive from freshly deposited plant residues to highly decomposed organic matter. Can range from less than 1% of the soil by weight to nearly 100%. Mixture of living organisms recently dead, decomposing and stable materials. It adds water and nutrient holding capacity to the soil. It get lost rapidly by Oxidation during warm and humid climates. Oxidation also adds nutrients to the soil. Soils that tends to stay wet also tend to have more organic matter [5]. Hydraulic conductivity is the ability of the soil to transmit water when saturated. For sandy soil high hydraulic conductivity as a result they drain fast. Similarly, clayey soil have low hydraulic conductivity as a result water drains slowly and these soils stay wet longer than sandy soils [6]. Water Holding Capacity (WHC) is another important property of soil. The amount of water a soil can hold against gravity. Related to the proportion of silt, clay and organic matter in the soil. So, sandy soils have low WHC. It can affect irrigation management i.e., smaller irrigations on soils with low WHC [7]. Cation Exchange Capacity (CEC) is a measure of the ability of the soil to hold positively

charged nutrients called cations (calcium, magnesium, potassium and ammonium-N against leaching). It is largely imparted by the clay and organic matter particles in the soil [8]. Mostly sandy soils have low quantities of clay and organic matter. So, that would be expected to have low CEC. In general the higher the CEC means greater fertility. The types of soils that encounter can be classified based on soil orders.

Nutrient mass balance is an analysis of the quantities of nutrients brought onto the farm and those leaving the farm. Especially those that end up in the environment. To help the farmer maximize the efficiency of the nutrients that the farmer purchases and brings onto the farm to grow the crops. The goals here are to return on investment in fertilizer and minimize losses to the environment. Nutrients can be imported to the farm (fertilizer, manure, Animal feed and Inflow Rivers and streams). It can be recycled on the farm (or) urban landscape. Buildup of the same can be happen. They are lost to the environment different from export (Runoff, Leaching and Gaseous). Mass balance approach, aim is to quantify these pools and to find out how much nitrogen would be associated with the crop that's taken off the farm or fed to the animals. The amount of nutrients that might be returned to the soil, through crop refuse. The amounts of nutrients that might be applied, to the fields and fertilizer and manure, and quantify the nutrients that might potentially be lost [28, 29, 30].

2.1. Soil Management Techniques.

2.1.1. Soil tillage advantages.

1. Turn under, incorporate organic matter to decompose
2. Mix soil and its constituents, organic matter, nutrients.
3. Turn under weeds
4. Turn under disease organisms insects.
5. Help dry out the soil for earlier planting
6. Prepare the seed bed

There are also disadvantages:

1. Loosens the soil and exposes soil to drying, can lead to wind and water erosion
2. Exposes soil organic matter to oxidation can lead to soil loses organic matter content

Cover crops is the solution for the soil management. In which, one can reduce soil erosion (wind (or) runoff). It can Recover unused nutrients, add organic matter to the soil. Survey on Sustainable Agriculture and its practices is presented in Table 2.1.

3. Methodology. The dataset considered in our experiment is used in yield prediction based on historic yield and weather information [20]. Implemented both the versions of Thomson model and compared the result with segmentation model, Random forest [21]. The complete description of the dataset used in the project is described [22]. The steps carried out in the present work to achieve the outcome of the listed objectives are listed in Fig. 3.1.

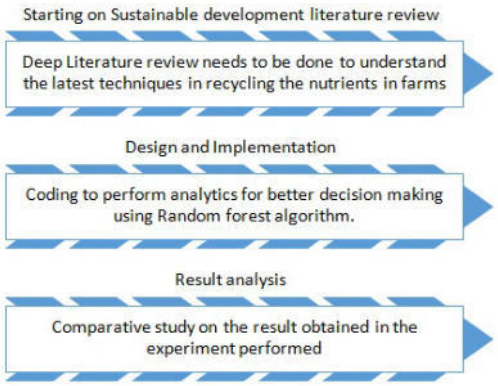


FIG. 3.1. *Steps followed to achieve the outcome*

TABLE 2.1
Survey on Sustainable Agriculture and its practices

Author Details	**Contribution**	**Features**	**Gaps**
[11]	Suggested three principles to be used in Circular Economy1. Waste is Food2. Make use of renewable energy3. Taking inspiration from nature	1. Food Production2. Food Distribution3. Food Consumption	Need to redesign food systems in line with circular principles to present a feasible solution.
[12]	Provides overview on the characteristics of regulations of organic farming and agroecology	1. Soil tillage2. Soil fertility3. Crop and cultivar choice4. Crop rotation5. Intercropping6. Management of landscape elements and habitats7. Pest, disease and weed management8. Water quantity and quality9. Agroforestry.	Focus on the restriction of external inputs and the limitation of chemical inputs
[13]	Describing the need for a link between agronomy and education in sustainability	To overcome epistemological boundaries between the natural and social sciences	Strong contrast between sustainable intensification in high-external-input agriculture
[14]	Addressees the possible ways to improve natural capital by generating more food.	1. Management of application of pesticides2.Agroecological system and habitat redesign	There are some regional scale exemplars of positive policy practice.
[15]	Cropland monitoring using time series analysis on satellite image	Random forest algorithm is used in image classification and achieved 95% accuracy	Make use of Google Earth Engine (GEE) platform to handle peta bytes of data.
[16]	Focused on developing precision agriculture and precision conservation to maintain sustainable development of agriculture at field level. Developed Geo-informatics sustainable agriculture framework	1.Modern form Management techniques2. Natural resources conservation service.3. Both agriculture people and policy makers are considers in decision making	Deployment of new virtual agriculture practices in finding out better environment outcome with the help of IOT, drones and Data servers.
[23]	Focused on crop management	Crop water productivity	Implications and roles of public and private sectors
[24]	Listed out Plant Growth Promoting Microorganisms (PGPM)	Efficient delivery system of PGPM	Amount of agriculture residuals used for energy purpose
[25]	Addressed key precision agriculture milestones like Global Navigation Satellite Systems (GNSS), Global Positioning System (GPS), Variable Rate Technology (VRT).	Amount of agriculture residuals used for energy purpose	Estimates the VRT adoption may exceed 50%

The reason for selecting random forest algorithm in predicting the yield production value is, as this algorithm allows the developer to incorporate the human drafted rules in most easier way. The implementation details of the thomson model is compared with the recent study made by [22]. In the research work [26], [27] the author had used optimization algorithm to improve the weights of each attribute in improving the accuracy of classification algorithms.

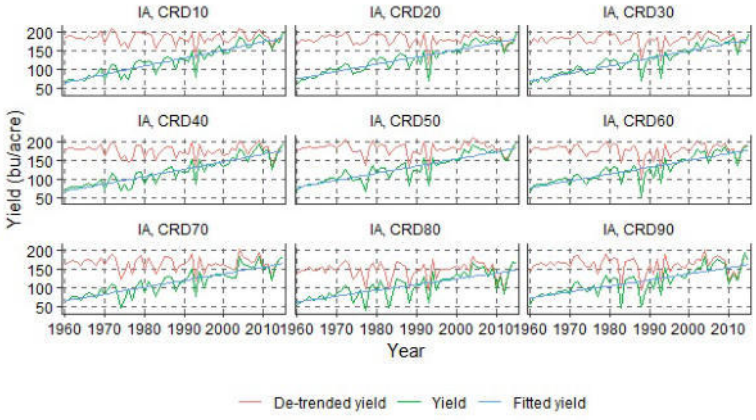


FIG. 4.1. *Plot on different types of yield*

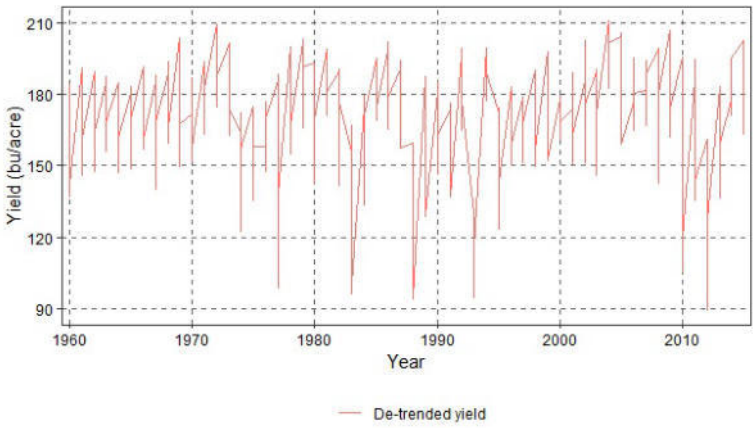


FIG. 4.2. *Effect of yielding from 1960 to 2010.*

4. Result and Discussion. To better understand the behavior of the attributes and its distribution the is plotted in Fig. 4.1 [18].

Van Eeuwijk et al. made in 2019 a similar kind of study from the year 1960 to 2000. The dataset is updated using cross validation technique with 10 folds and thereby improved the number of instances from the dataset collected. Fig. 4.2 helps in interpreting the impact of yield from 1960 to 2010.

The mean calculated mean value of the Temperature year wise is plotted to understand the impact of temperature on yielding crops like corn and soybeans Fig. 4.3. The performance of Random forest using Mean Square Error is estimated and plotted in Fig. 4.4. The comparative performance of the models implemented in the present work is plotted in Fig. 4.5. The Error (MSE) in prediction and its interpretation is done with the help of Fig. 4.6.

5. Conclusion and Future scope. The Goal is to develop the social, environmental, and the economic aspects of possible solutions to minimize the agricultural footprint, and become more sustainable. . Implemented both the versions of Thomson model and compared the result with segmentation model, Random forest. Root Mean Square Error and Mean Absolute Error are used as evaluation metrics in estimating the performance of models implements and stated that Random forest algorithm is providing 0.07 (RMSE). The outcome of the present research work helps farmers in adopting best management practices and trying to give

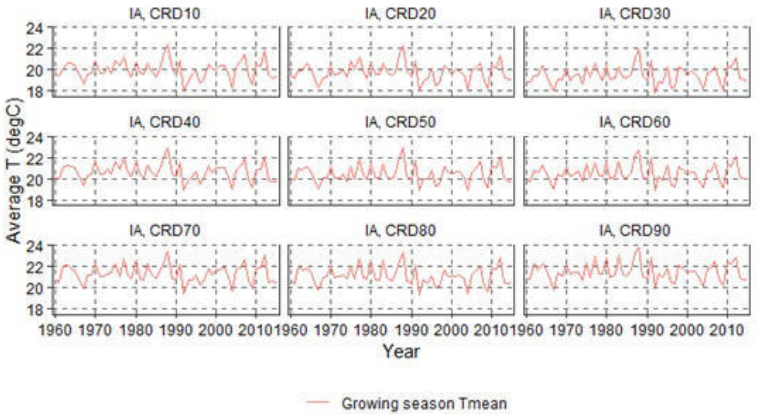


FIG. 4.3. *Plot on Season factor on Temperature.*

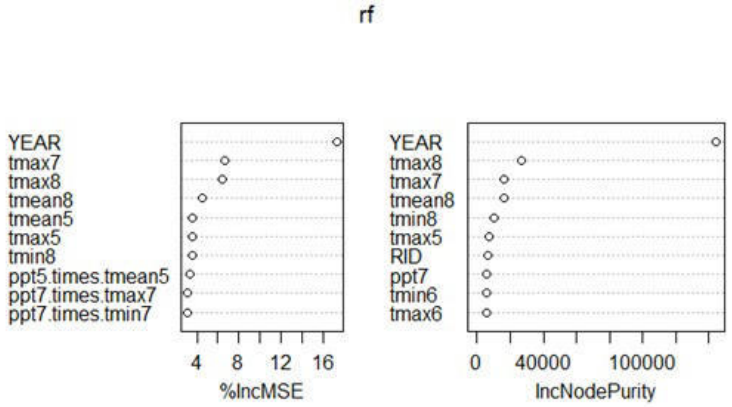


FIG. 4.4. *Performance of Random Forest*

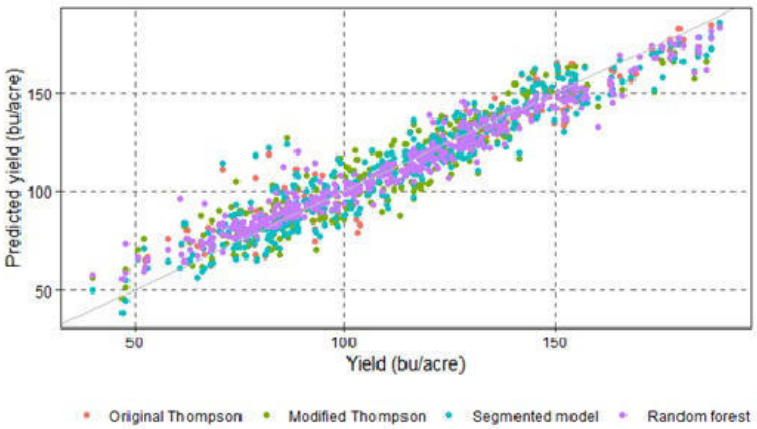


FIG. 4.5. *Comparison on yield prediction Level*

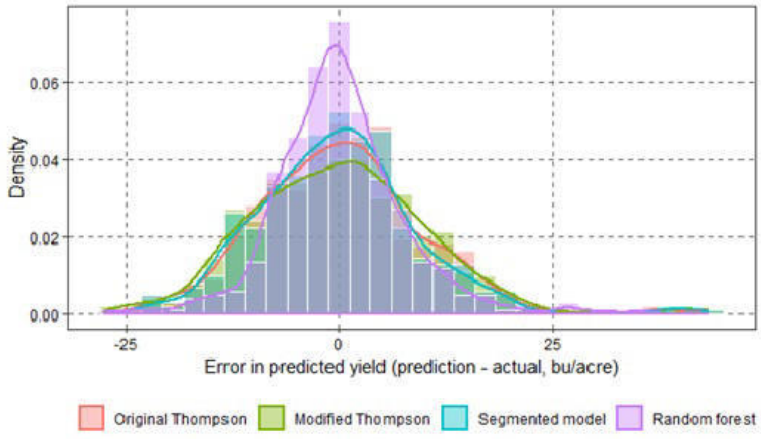


FIG. 4.6. *Comparison on Error in yield prediction*

them the economical and technical support in making easier for them to adopt best management practices. The outcome of the present work helps farmers to manage/monitor more profit and production without effecting the environment. The farmer can maintain the cycle of nutrients with the help of environment. To put forward sustainable use of natural resources and inputs. In future, the parameters like Soil Texture, Hydraulic Conductivity, Water Holding Capacity, Cation Exchange Capacity to be considered in our experiments to support sustainable agriculture.

REFERENCES

[1] BISWAS, H., RAIZADA, A., MANDAL, D., KUMAR, S., SRINIVAS, S., & MISHRA, P. K. *Identification of areas vulnerable to soil erosion risk in India using GIS methods* Solid Earth, 6(4), 1247, 2015

[2] OMONDI, M. O., XIA, X., NAHAYO, A., LIU, X., KORAI, P. K., & PAN, G., *Quantification of biochar effects on soil hydrological properties using meta-analysis of literature data* Geoderma, 274, 28-34, 2016.

[3] OLORUNFEMI, I. E., &FASINMIRIN, J. T. *Land use management effects on soil hydrophobicity and hydraulic properties in Ekiti State, forest vegetative zone of Nigeria* Catena, 155, 170-182, 2017.

[4] KOHLER, J., ROLDÁN, A., CAMPOY, M., &CARAVACA, F. *Unraveling the role of hyphal networks from arbuscularmycorrhizal fungi in aggregate stabilization of semiarid soils with different textures and carbonate contents* Plant and Soil, 410(1-2), 273-281, 2017.

[5] STIRLING, G., HAYDEN, H., PATTISON, T., &STIRLING, M. *Soil health, soil biology, soilborne diseases and sustainable agriculture: A Guide* Csiro Publishing, 2016.

[6] NAVARRETE, J. L. *Evaluation Of Recycled Gypsum Application Dosages To Enhance The Water Infiltration Rate At Water Retention Ponds*, 2018.

[7] MOHAMED, B. A., ELLIS, N., KIM, C. S., BI, X., &EMAM, A. E. R. *Engineered biochar from microwave-assisted catalytic pyrolysis of switchgrass for increasing water-holding capacity and fertility of sandy soil* Science of the Total Environment, 566, 387-397, 2016.

[8] LITTMANN, R. J. *U.S. Patent Application No. 15/092,079*, 2016.

[9] SALADINI, F., BETTI, G., FERRAGINA, E., BOURAOUI, F., CUPERTINO, S., CANITANO, G., & BIDOGLIO, G *Linking the water-energy-food nexus and sustainable development indicators for the Mediterranean region* Ecological Indicators, 91, 689-697, 2018

[10] HAK, T., JANOUŠKOVÁ, S., & MOLDAN, B. *Sustainable Development Goals: A need for relevant indicators* Ecological Indicators, 60, 565-573, 2016.

[11] DUNCAN, J. A. B., & PASCUCCI, S. *Circular solutions for linear problems: Principles for sustainable food futures.* Solutions, 7(4), 58-65, 2016

[12] MIGLIORINI, P., & WEZEL, A. *Converging and diverging principles and practices of organic agriculture regulations and agroecology. A review* Agronomy for sustainable development, 37(6), 63, 2017.

[13] STRUIK, P. C., & KUYPER, T. W. *Sustainable intensification in agriculture: the richer shade of green. A review* Agronomy for Sustainable Development, 37(5), 39, 2017.

[14] PRETTY, J. *Intensification for redesigned and sustainable agricultural systems.* Science, 362(6417), eaav0294, 2018.

[15] GUMMA, M. K., THENKABAIL, P. S., TELUGUNTLA, P. G., OLIPHANT, A., XIONG, J., GIRI, C., & WHITBREAD, A. M. *Agricultural cropland extent and areas of South Asia derived using Landsat satellite 30-m time-series big-data using*

random forest machine learning algorithms on the Google Earth Engine cloud GIScience & Remote Sensing, 1-21, 2019.

[16] Delgado, J., Short, N. M., Roberts, D. P., & Vandenberg, B. *Big Data Analysis for Sustainable Agriculture* Frontiers in Sustainable Food Systems, 3, 54, 2019.

[17] Bouis, H. E., & Saltzman, A. *Improving nutrition through biofortification: a review of evidence from HarvestPlus, 2003 through 2016* Global food security, 12, 49-58, 2017

[18] Togliatti, K., Archontoulis, S. V., Dietzel, R., Puntel, L., & VanLoocke, A. *How does inclusion of weather forecasting impact in-season crop model predictions?* Field Crops Research, 214, 261-272, 2017.

[19] Van Eeuwijk, F. A., Bustos-Korts, D., Millet, E. J., Boer, M. P., Kruijer, W., Thompson, A., & Muller, O *Modelling strategies for assessing and increasing the effectiveness of new phenotyping techniques in plant breeding. Plant science*, 282, 23-39. 2019.

[20] Thompson, L. M. *Weather and technology in the production of corn and soybeans*, 1963.

[21] Subudhi, A., Dash, M., & Sabut, S. *Automated segmentation and classification of brain stroke using expectation-maximization and random forest classifier* Biocybernetics and Biomedical Engineering, 40(1), 277-289. 2020.

[22] Bullock, D. W. *The Influence of State-Level Production Outcomes Upon US National Corn and Soybean Production: A Novel Application of Correlated Component Regression* No. 1187-2019-1851, 2017.

[23] Yost, M. A., Sudduth, K. A., Walthall, C. L., & Kitchen, N. R. *Public–private collaboration toward research, education and innovation opportunities in precision agriculture. Precision Agriculture, 20(1), 4-18.*

[24] Ma, Y *, (2019). Seed coating with beneficial microorganisms for precision agriculture. Biotechnology advances, 37(7), 107423.*

[25] Lowenberg-DeBoer, J., & Erickson, B. *(2019). Setting the record straight on precision agriculture adoption. Agronomy Journal*, 111(4), 1552-1569.

[26] Basha, S. M., Rajput, D. S., & Vandhan, V. (2018). Impact of gradient ascent and boosting algorithm in classification. *International Journal of Intelligent Engineering and Systems (IJIES)*, 11(1), 41-49.

[27] Basha, S. M., & Rajput, D. S. (2018, November). Evaluating the Importance of each Feature in Classification task. *In 2018 8th International Conference on Communication Systems and Network Technologies* (CSNT) (pp. 151-155). IEEE.

[28] Baliarsingh, S. K., Vipsita, S., Gandomi, A. H., Panda, A., Bakshi, S., & Ramasubbareddy, S. (2020). Analysis of high-dimensional genomic data using MapReduce based probabilistic neural network. *Computer methods and programs in biomedicine*, 195, 105625.

[29] Attili, V. R., Annaluri, S. R., Gali, S. R., & Somula, R. (2020). Behaviour and Emotions of Working Professionals Towards Online Learning Systems: Sentiment Analysis. *International Journal of Gaming and Computer-Mediated Simulations (IJGCMS)*, 12(2), 26-43.

[30] Ravindranath, V., Ramasamy, S., Somula, R., Sahoo, K. S., & Gandomi, A. H. (2020, July). Swarm Intelligence Based Feature Selection for Intrusion and Detection System in Cloud Infrastructure. *In 2020 IEEE Congress on Evolutionary Computation (CEC)* (pp. 1-6). IEEE.

Edited by: Rajkumar Rajasekaran
Received: Mar 17, 2020
Accepted: Sep 13, 2020


Scalable Computing: Practice and Experience, ISSN 1895-1767, http://www.scpe.org
© 2020 SCPE. Volume 21, Issues 4, pp. 601–610, DOI 10.12694:/scpe.v21i4.1718

NOISE DEDUCTION IN NOVEL PADDY DATA REPOSITORY USING FILTERING TECHNIQUES

MALATHI V. *AND GOPINATH M.P†

Abstract. Classification of paddy crop diseases in prior knowledge is the current challenging task to evolve the economic growth of the country. In image processing techniques, the initial process is to eliminate the noise present in the dataset. Removing the noise leads to improvements in the quality of the image. Noise can be removed by applying filtering techniques. In this paper, a novel data repository created from different paddy areas in Vellore, which includes the following diseases, namely Bacteria Leaf Blight, Blast, Leaf Spot, Leaf Holder, Hispa and Healthy leaves. In the initial process, three kinds of noises, namely Salt and Pepper noise, Speckle noise, and Poisson noises, were removed using noise filtering techniques, namely Median and Wiener filter. The interpretation made over the median and Wiener filtering techniques concerning noises, the performance of the methods measured using metrics namely PSNR (peak to signal to noise ration), MSE (mean square error), Maxerr (Maximum squared error), L2rat (ratio of squared error). It is observed that the PSNR value of the hybrid approach is 18.42dB, which produces less error rate as compared with the traditional approach. Results suggest that the methods used in this paper are suitable for processing noise.

Key words: Salt and pepper noise; Speckle noise; Poisson noise; Median filter; Wiener filter.

AMS subject classifications. 60G35

1. Introduction. Agriculture is the primary resource to increase the economic growth of our country; it is the foundation of our economic framework. Paddy is the essential crops cultivated throughout the world, during the production period the majority of the yield loss occurs due to the diseases listed below Brown Spot, Bacterial Leaf Blight, and Blast. Diagnosing the diseases in an earlier stage using the computational approach plays an essential role in today's world [1]. Nowadays, the whole world depends on digital data, so processing the digital image is the most welcoming research topic. Data sets collected by camera are usually contaminated by noise. Noise is an essential factor that influences the quality of the image, which is primarily produced in the processes of image acquisition. Eliminating noise from the raw image is still a challenging issues for researchers. Thus, noise removal is the most necessary and the first step to be taken before the images is processed. It is essential to apply an efficient noise reduction technique to compensate for such data corruption. Image denoising remains a challenge for researchers because noise removal introduces artifacts and causes blurring of the images. So This paper mainly focuses and describes Wiener and median filtering techniques for noise reduction.

The combination of the pixel value forms an images. During image acquisition, there is a chance of changing the pixel value due to the noise present in the environment [2]. Noise might emerge during image capturing and transmission procedure, it corrupts the significant pixel value in the image, yet additionally influences the particular visualization of the image. Therefore, noise reduction plays a vital role in image processing and computer vision [3]. In Section 3, noises, namely Salt and Pepper noise, Speckle noise, Gaussian noise, and Poisson noise, are discussed. The above-listed noise is present in the digital image, and it is a tedious process to eliminate the specific noises in prior knowledge. The noise can be reduced either by linear or non-linear filter method; in noise filtering techniques, the most challenging task is to filter the noise without any information loss. Various kinds of noise reduction techniques, namely linear filter, min filter, max filter, median filter, wiener filter, Gaussian filter, guided, BM3D (Block matching and 3D filtering), and adaptive fuzzy switching median filter remove noises present in the image [4]. In the median filter, the performance can be enhanced by

*School of Computer Science and Engineering, Vellore Institute of Technology, Katpadi, Vellore, India.
† Associate Professor, School of Computer Science and Engineering, Vellore Institute of Technology, Katpadi, Vellore, India (mpgopinath@vit.ac.in).

the researchers by utilizing the conventional median filter [5] [6] [7]. Dividing-conquering strategy improves the computational speed, and the complexity of the algorithm was achieved $O(N2)$ to $O(n(ln(n)))$ [8] [9]. In the proposed work, the novel data repository created and three noises, namely Salt and Pepper noise, Speckle noise, and Poisson noise, are introduced. Median and Weiner filtering techniques remove noises. The comparative study made with various kinds of noise concerning noise reduction techniques. Performance metrics, namely PSNR, MSE, Maxerr, L2rat used to evaluate the quality of the image.

The structure of the paper categorized as follows: Section 2 explains the related work; Section 3 describes the novel data repository and noise reduction techniques. Section 4 defines the strategies used to evaluate the quality of the images using various performance evaluation metrics. Section 5 report's the conclusion of the work.

2. Related Works. In this section, the recent contribution made by various researchers discussed below. Reza et al. [10] proposed a system to detect Jute Plant Leaf using Image Processing techniques and Machine learning techniques, the flow of the work categorize into two steps: initial stage involves pre-processing task, and the final stage is segmentation process. Bilateral filter techniques used to remove the noise present in the image and replaced with intensity value. Agarwal et al.[11] proposed denoising the images using various filtering techniques, namely special filter, wiener filter, and median filter. The comparative study conducted on these filtering techniques; at last, the author concluded that a wiener filter is a more suitable method to denoise the images. Orillo et al.[12] proposed a system using inverse laplacians; the issue of the monogenetic signal was handled efficiently by using this algorithm. Archana et al.[13] intended the detection of plant diseases, the essential investigating strategy for programmed recognizable proof is the sifting procedure of pre-processing technique. Henceforth, this Image filtering assumes a significant job to expel clamor from the image. Thus, this pre-processing strategy is the underlying stage to improve image quality. The comparison made over four different filters, namely Gaussian filter, median filter, mean filter, and Wiener by utilizing a standard data-set. The image nature of general outcomes shows that the examination of different separating strategy performed to upgrade quality. So, this paper gives the best beginning for specialists to the programmed discovery of rice plant sickness identification. The performance metrics evaluated using PSNR, SNR, and MSE. Through this work, among all the separating procedure, Weiner channel has preferred outcome over all the channels of most noteworthy PSNR qualities and SNR values. So these techniques used to pre-process dark-colored spot sickness in rice plants, which assists with dissecting further preparing.

Rani et al.[14] performed a comparative study of various filtering techniques. It was hard to state which filtering techniques are the best among all the filters. Exploring the outcomes by applying distinctive filter types to an image the accompanying resultants states: The Block Matching filter (BM3D) observed to be the stable method. The median filter is the best filtering technique for salt and pepper noise and Gaussian noise. Speckle noise works average for min and max filter, guided filter suits for Poisson noise. Salt and pepper noise can be removed effectively by using an Adaptive fuzzy median filter. At last, BM3D is a suitable method to eliminate all kinds of noises. Kaur [15] performed a comparative study over different filtering techniques by using the performance metrics. The performance of the filter varies, eventually concerning the noise present in the image. Salt and Pepper noise removed effectively using a median filter. Similarly, Poisson and speckle noise goes well with wiener filter. The essential results achieved recently by the authors in noise reduction techniques have mentioned in Table 2.1. The resultant analysis states that PSNR is the most frequently used noise measuring metrics and observed that the average PSNR value is around 23 dB for any kind of dataset, and a minimum of 14.45 can be achieved based on the kind of filtering techniques used.

3. Proposed Work.

3.1. Novel Data Repository.

3.1.1. Camera Specification. The images are captured with the help of high-resolution cameras, namely Canon EOS 1200D and FLIR E8 (Thermal camera). The specification of these cameras are mentioned below, Canon EOS 1200D 18MP, Digital SLR Camera (Black) with EF-S 18-55mm f/3.5-5.6, is II Lens and FLIR E8's crisp 76,800 (320 X 240) pixel infrared resolution, +2% accuracy of reading for ambient temperature 10°C to 35° C(50° to 95°F) and object temperature above 0°C (32°F), the field of view is 45° X 34°.

TABLE 2.1
Review for Noise reduction using filtering techniques

Year	author	methods	PSNR	SNR	SSIM	MSE	Dataset
			27.81	-	0.712	-	set 12 dataset(lena images)
2019	Fan et al	wiener filter	22.95	-	0.647	-	set 12 dataset(boat image)
			23.91	-	-	-	set 12 dataset(monarch image)
2018	Archana et al.	Wiener filter	38.2	14.98	-	42.75	Paddy leaf
		Linear filter	14.45	-	-	1330	
2018	Rani et al	Min filter	18.85	-	-	1075.52	
		Max filter	18.49	-	-	1156.6	MATLAB cameraman image
		Median filter	29.21	-	-	237.5	
2012	Zhu et al	Wiener filter	25.93	-	-	165.96	Lena image
		Median filter	32.118	26.105	-	-	

3.1.2. Image Collection. The images collected from the state of Tamil Nadu include the following regions; the Agri field (VIT School of Agricultural Innovations And Advanced Learning. (VAIAL), VIT, Vellore), Brahmapuram, sevur, Latheri, vaduthangal from Vellore district. The complete field survey has undergone concerning disease symptoms and climatic conditions; samples were collected and captured using two different high-resolution cameras, as mentioned in section 3.1.1. Later, the diseases classified with the help of plant pathologists from VAIAL, Vellore Institute of Technology, Vellore. In the region, as mentioned above, the paddy crop leaves are affected by two significant reasons, namely Diseases and Pest. The leaves turn to yellow; the brown spot appears over the leaves; insects consume the leaves as well as hold the leaves and lay their eggs. Insect nest is the major problem in crops; the only solution to destroy the insect nest is clipping off the leaves, but this process can cause the bacteria to enter quickly through open pores and affects the plants. Different kinds of diseases include bacteria leaf blight, blight, leaf spot, leaf holder, hispa, and healthy leaves are collected in the region, as mentioned earlier, are illustrated in Table 3.1. Sample images of the data repository shown in Fig 3.1.

TABLE 3.1
count of the images included in the repository

S.No	Diseases	Number of images captured using canon 1200D	Number of images captured using FLIR E8	Total
1	Bacteria leaf blight	208	341	549
2	Blight	–	206	206
3	Leaf spot	188	225	413
4	Leaf holder	54	69	123
5	Hispa	53	317	370
6	Healthy leaves	–	427	427
			Totally	2,088

3.2. Noise in Paddy images. The digital image categorized into binary, grayscale, and color images. In the binary image, the pixel value lies between zero or one (i.e., either black or white). In a grayscale image, pixel value lies between 0 to 255, and this implies every pixel right now appeared by eight bits, which actually of one byte. The color image generated by the combination of the red, green, and blue pixel values. Therefore 2563 different combination color values are obtained. During the image capturing process, due to the ultraviolet radiation and the dust present in the environment causes variation in the pixel value. Salt and pepper noise, Gaussian noise, speckle noise, and Poisson noise are the kinds of noises present in the image [15].

3.2.1. Salt and Pepper noise. Salt and pepper noise comes under the category of impulse; the resultant of this noise leads to white dot scattered over the image, and the pixel value is in dynamic range. The salt and

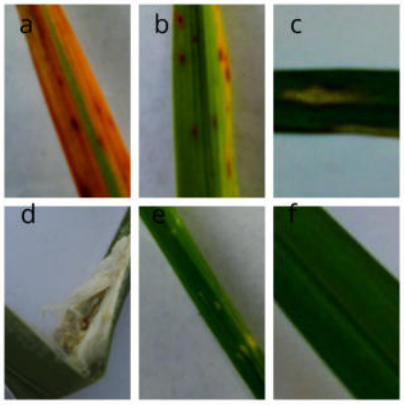


FIG. 3.1. *Sample image data repository a. Bacteria leaf blight; b. Leaf spot; c. Blast; d. Leaf holder; e. Hispa; f. Healthy leaf*

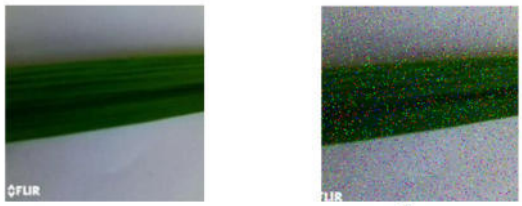


FIG. 3.2. *a. Input image of healthy leaves; b. images with salt and pepper noise*

pepper noise occurs due to the following issues like a malfunction in sensor, camera, information transmission, and storage allocation [16]. Fig 3.2 illustrates, salt and pepper noise added to the original image, i.e., salt noise induced by adding a brightness value of 255 pixels and pepper noise influenced by adding darkness with 0-pixel values.

3.2.2. Poisson Noise. Poisson, commonly known as shot photon noise, occurred when the count of photons detected by the senor isn't adequate to give distinguishable statistical data. Shot clamor might be overwhelmed when the limited number of particles that convey vitality is adequately little, so vulnerabilities because of the Poisson dispersion, which depicts the event of autonomous irregular occasions, are of centrality [17]. Fig 3.3 illustrates the induction of Poisson noise in the bacteria leaf blight.

3.2.3. Speckle Noise. Speckle noise is commonly known as Multiplicative noise. The noise was created by multiplying random value concerning pixel value and illustrated by using formula (3.1) mentioned below. Speckle noise observed in a radar system, even though it might show up in a remotely detected picture using lucid radiation [18]. Fig 3.4 illustrates the induction of speckle-noise over bacteria leaf blight and healthy leaves.

$$s = x + u * x \tag{3.1}$$

where s represents the speckle noise, x is the input image, u is the uniform noise image, m is the mean and v represent the variance.

3.3. Noise reduction techniques. The above-discussed noise can be removed by applying filtering techniques to the noisy data. The various noise filtering techniques are Linear filter, Median filter, Wiener filter, Gaussian filter, Kuan filter, Bilateral filter, Adaptive median filter, Adaptive wiener filter, Mean filter, Adaptive fuzzy, Guided filter, BM3D filter. In this section, two primary filtering techniques, namely Median and Wiener filter, were applied over the data-set, and performances measured. Wiener filter is optimized to identify the mean square error estimate of the original signal from a noisy measurement. Wiener filter is the frequently used

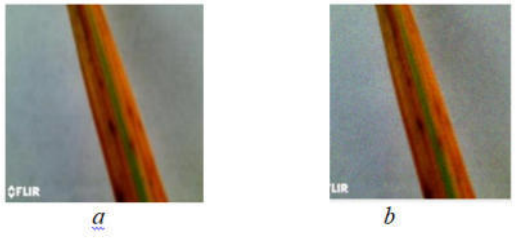


FIG. 3.3. *a. Input image of Bacteria leaf blight disease; b.Images with Poisson noise*

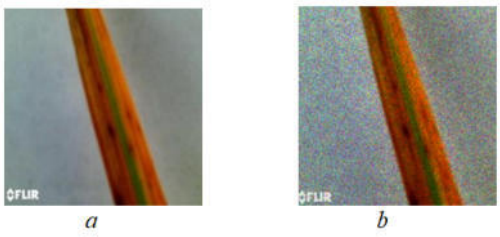


FIG. 3.4. *a. Input image of Bacteria leaf blight disease; b.Images with speckle noise*

technique to eliminate the blurriness present in the images. Wiener filtering is, in fact, the underlying premise for the restoration of other kinds of a blur; and being a least-mean-squares technique, it has roots in a spectrum of other engineering applications [20]. Mean filters usually remove the edges that have a high-frequency component, and this problem can overcome by using the median filter. The median filter protects the image information in horizontal, vertical, and diagonal directions. Computational complexity is low [21].

3.3.1. Median filter. Median filter falls under the non-linear filter type. In the median filtering technique, the smoothing method used to remove the noise present in the image. This filter furthermore brings down the intensity variety among one and the rest of a considerable number of pixels. The median is measured as follows, the pixel value arranged in ascending order concerning window size later, and the middle value replaced with the whole pixel value. The median filter measured using the formula (3.2) mentioned below [19].

$$m(x,y) = medianw(q,r), \qquad \text{where}(q,r) \in (x,y) \tag{3.2}$$

Pros of the median filter are, it is simple to implement, no need for generating new pixel value and Less sensitive, so the noise pixel value are removed effectively [22].

3.3.2. Wiener filter. Wiener filter is the kind of statistical approach; Wiener filter design is to cut down the measure of noise display in a signal by examining the noiseless signal. It diminishes the bulk of noise present in the noise by a Statistical approach. Wiener filter produces better results as compared to linear filter, but it consumes more computational cost as compared with other filters, as mentioned in the previous session. The application of the Wiener filter such as, linear identification, channel restoration, equalization, and system detection. The pros and limitation of wiener filter are it uses a broad window side to do the smoothing process, and it is more suitable to reduce speckle noise present in the image. The obscure edge reduced by cutting down the Wiener size. Limitations of the Wiener filter are, computational cost is more. It doesn't support well for nature noise since the restoration process doesn't achieve perfection. In the Wiener filter, detecting power spectra is problematic [22].

4. Results and Discussions.

4.1. Software and hardware requirement. The experiment was implemented in the following hardware; Intel(R) Core(TM) i5-8300H CPU @ 2.30GHz, 8GB RAM, 64-bit Operating System, x64 based processor, and software specification; Matlab R2019b 64-bit.

4.2. Performance evaluation metrics. The quality of the image measured using the performance evaluation metrics includes PSNR (peak signal-to-noise ratio), MSE (mean square error), Maxerr (maximum squared error), and L2rat (ratio of squared error).

4.2.1. PSNR. The term peak signal-to-noise ratio (PSNR) is an appearance for the ratio among the maximum probable rate (power) of a signal and the power of distorting noise that changes the quality of its illustration.

$$PSNR = 20\log 10\frac{2\alpha - 1}{\sqrt{(MSE)}} \tag{4.1}$$

$$MSE = \frac{\|x - y\|}{N} \tag{4.2}$$

4.2.2. MSE. MSE states the average error among raw images and the noisy image. The error represents the quantity of the raw image varies from the image

$$MSE = \frac{1}{qp}\sum_{0}^{q-1}\sum_{0}^{p-1} m(i,j) - n(i,j) \tag{4.3}$$

$$[PSNR, MSE, MAXERR, L2RAT] = measerr(X, XAPP) \tag{4.4}$$

Maxerr is the maximum absolute squared deviation of the data, X, from the approximation $XAPP$. $L2RAT$ is the ratio of the squared norms of the signal or image approximation, $XAPP$, to the input signal or image X.

4.3. Interpretation over filtering techniques. The experiment analysis carried out using a different kinds of approaches. Initially, the three kinds of noises, namely Salt and Pepper noise, Poisson noise, and Speckle noise was induced over the dataset using imnoise function, later two different filtering techniques like Median filter and Wiener filter are applied to remove the noise as illustrated in Fig 4.1. Observation states that, for Salt and Pepper noise, the PSNR and MSE median filter is less than the wiener noise. So, concluded that the median filter works better as compared with the Wiener filter, as the resultant value illustrated in Tables 4.1, 4.2 and 4.3.

TABLE 4.1
Performance evaluation metrics for salt and pepper noise, noises filtered using median and wiener filter.

Type of filter										
Classes	Median filter					Wiener filter				
	PSNR		MSE	MAXERR	L2RAT	PSNR		MSE	MAXERR	L2RAT
	Original	Denoise				Original	Denoise			
Bacteria leaf blight	37.2513	30.825	12.244	134	0.998	24.958	18.587	207.588	166	1.011
Blast	35.986	28.640	16.384	170	0.997	23.015	15.746	324.719	227	1.0156
Leaf spot	38.854	32.538	8.466	154	0.998	24.487	18.207	231.379	155	1.0070
Leaf holder	37.888	31.63	10.573	153	0.999	24.852	18.637	212.738	188	1.0092
Leaf holder	37.888	31.63	10.573	153	0.999	24.852	18.637	212.738	188	1.0092
Healthy	38.603	32.268	8.968	143	0.998	24.200	17.905	247.17	201	1.0078

TABLE 4.2
Performance evaluation metrics for speckle noise, noises filtered using median and Wiener filter.

Classes	Type of filter									
	Median filter					Wiener filter				
	PSNR		MSE	MAXERR	L2RAT	PSNR		MSE	MAXERR	L2RAT
	Original	Denoise				Original	Denoise			
Bacteria leaf blight	29.7453	23.318	68.952	88	0.998	29.543	23.125	72.237	79	1.0004
Blast	27.500	20.146	115.612	179	0.995	28.384	21.037	94.33	89.000	0.997
Leaf spot	26.694	20.387	139.198	155	1.0008	28.956	22.661	82.68	77	1.0036
Leaf holder	26.714	20.444	138.545	154	0.996	29.295	23.034	76.479	85	0.998
Hispa	26.92	20.422	132.153	149.00	0.999	29.589	23.088	71.475	70	0.9999
Healthy	26.802	20.453	135.784	139	0.995	29.461	23.125	73.613	75	0.998

TABLE 4.3
Performance evaluation metrics for Poisson noise, noises filtered using median and Wiener filter.

Classes	Type of filter									
	Median filter					Wiener filter				
	PSNR		MSE	MAXERR	L2RAT	PSNR		MSE	MAXERR	L2RAT
	Original	Denoise				Original	Denoise			
Bacteria leaf blight	34.594	28.159	22.576	127	0.996	37.298	30.8712	12.113	31	0.998
Blast	34.499	27.137	23.076	177	0.9941	38.756	31.408	8.658	37.000	0.9974
Leaf spot	35.421	29.094	18.659	159	0.996	37.501	31.185	11.558	30	0.998
Leaf holder	34.72	28.455	21.928	150	0.997	37.341	31.077	11.992	30	0.998
Hispa	34.797	28.291	21.543	143	0.998	37.512	31.005	11.529	25	0.997
Healthy	35.248	28.906	19.419	141	0.997	38.043	31.711	10.203	28	0.999

In the second approach since the kind of noises can't be easily detected in the data-set, so two different filtering techniques namely median and Wiener noise are applied over the data-set to remove salt and pepper noise, Poisson noise and wiener noise. The performance metrics illustrated in Table 4.4.

In the third approach, the hybrid technique followed, as illustrated in Fig 4.2, initially, introduce salt and pepper noise, Poisson noise, and speckle noise over the data-set. Later median filter techniques used to remove the noise. Then hybrid method (median and wiener filter) were used to remove the noise. The resultant value illustrated in Table 4.5. The observed value states that the error value is less and works better for the hybrid model.

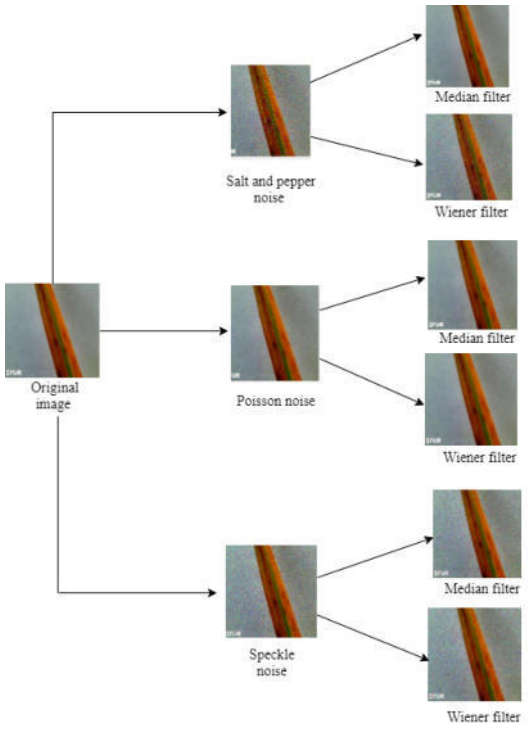


FIG. 4.1. *Induced noises removed using median and Wiener filter*

TABLE 4.4
Evaluation metrics for the given data repository using median and Wiener filter.

Type of filter										
Classes	Median filter					Wiener filter				
	PSNR		MSE	MAXERR	L2RAT	PSNR		MSE	MAXERR	L2RAT
	Original	Denoise				Original	Denoise			
Bacteria leaf blight	38.092	31.667	10.089	131	0.998	37.542	31.115	11.431	136	0.9984
Blast	36.47	29.127	14.657	170	0.9985	36.244	28.897	15.439	175	0.997
Leaf spot	39.24	32.523	7.745	144	0.9991	38.392	31.672	9.416	137	0.9985
Leaf holder	38.262	32.0036	9.702	147	0.9992	37.675	31.416	11.105	144	0.998
Hispa	38.2002	31.6987	9.842	139	0.991	37.655	31.152	11.156	140	0.998
Healthy	38.913	32.581	8.3517	142	0.9993	38.606	32.27	8.963	142	0.9986

5. Conclusion. In this work, image filtering techniques, namely Median filter and Wiener filter, were applied to remove the Salt and pepper, Speckle, and Poisson noises present in the given novel data repository. The interpretation made over various kinds of noise concerning noise reduction techniques. The quality of the image evaluated using performance metrics, namely PSNR (peak to signal to noise ration), MSE (mean square error), Maxerr (Maximum squared error), L2rat (ratio of squared error). The resultant value states that the median filter produces a better solution to eliminate noise as compared with a Wiener. In future work, we are planning to incorporate more filtering techniques over various kinds of crop leaves.

TABLE 4.5
Hybrid approach.

Classes	Type of filter									
	Median filter					Wiener filter				
	PSNR		MSE	MAXERR	L2RAT	PSNR		MSE	MAXERR	L2RAT
	Original	Denoise				Original	Denoise			
Bacteria leaf blight	26	20	142	139	0.988	24.912	18.426	209.82	127	0.984
Blast	27	19	125.48	182	0.989	24.27	16.881	242.90	179	0.986
Leaf spot	26.71	20.28	138.51	134	0.987	25.06	18.61	202.35	146	0.98
Leaf holder	26.54	20.24	144.03	157	0.989	24.72	18.41	218.86	149	0.987
Hispa	26.74	20.20	137.57	153	0.989	24.94	18.40	208.35	148	0.990
Healthy	26.60	20.22	142.21	148	0.988	24.68	18.29	221.10	149	0.985

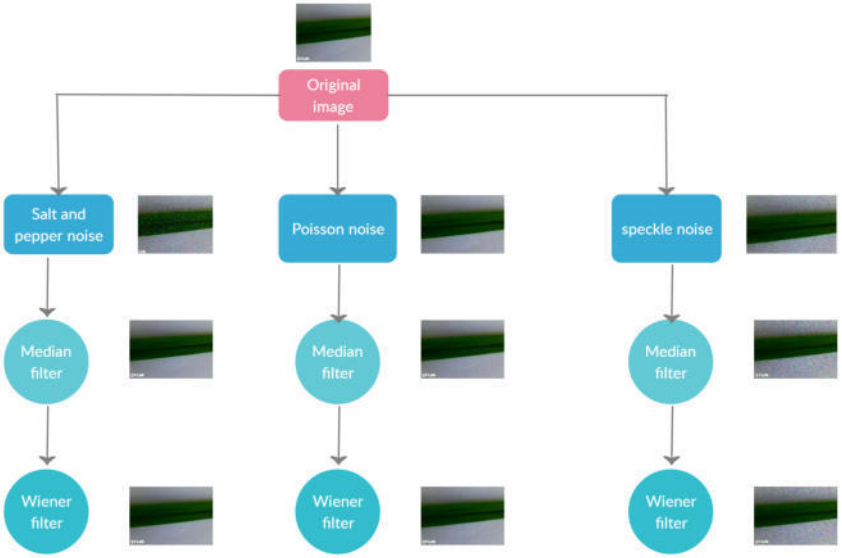


FIG. 4.2. *A hybrid approach to reduce noise from the image data-repository*

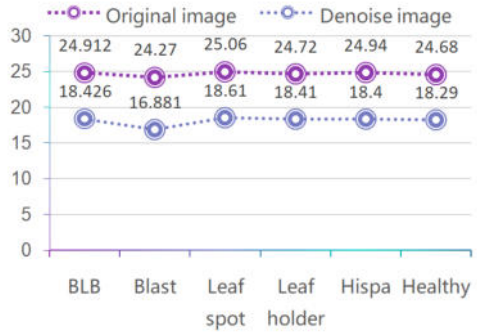


FIG. 4.3. *Hybrid approach*

Acknowledgments. This research work was supported by Dr. Aarthy S.L and Dr. Sujatha R from the school of Information Technology and Engineering, Vellore Institute of Technology, and Dr. Priyadharshini B plant pathologist from VAIAL, Vellore Institute of Technology, Vellore.

REFERENCES

[1] Guchait N , Bhakta I. , Phadikar S. And Majumder K., *Visual computing for blast and brown spot disease detection in rice leaves.* , In proceedings of the 2nd international conference on communication, devices, and computing, springer, singapore ,pp. 595-606, 2020.
[2] Rath A.K. and Meher J.K, *Disease detection in infected plant leaf by computational method*, Archives of phytopathology and plant protection, 1-11,2020.
[3] Chatterjee A., Roy S. And Das S.,*Feature selection using rough set theory from infected rice plant images*, In computational intelligence in pattern recognition, springer, singapore. Pp. 417-427, 2020.
[4] Brady, D. J., Hu, M., Wang, C., Yan, X., Fang, L., Zhu, Y. And Ma, Z , *Smart cameras*, Arxiv preprint arxiv:2002.04705, 2020.
[5] Guohong Liu, Wenming Guo,*Application of Improved Arthmetic of Median Filtering Denoising*,Computer engineering and applications, vol.46, no.10, pp.187-189,2010.
[6] Xiaokai Wang, Feng Li,, *Improved Adaptive Median Filtering* ,Computer Engineering And Applications, vol.46, no. 3, pp. 175-176, 2010.
[7] Chao Wang, Zhongfu Ye , *Salt-And-Pepper Noise Removal By Adaptive Median Filter And Tv* , In painting, Journal Of University Of Science And Technology Of China, Vol.38, No. 3, Pp. 282-287, 2008.
[8] Qinghua Huang, Heqin Zhou, Huanqing Feng, *A Fast And Effective Algorithm Of Pulse Noise Filtering For Imaging Data*, Computer Engineering And Applications, No. 17, Pp. 113- 114, 210, 2002.
[9] T S Huang, G T Tang , *A Fast Two- Dimensional Median Filtering Algorithm* ,IEEE Trans Acoustics, Speech, And Signal Processing, Vol.27, No. 1, Pp. 13- 18,1979.
[10] Reza Zn, Nuzhat F, Mahsa NA And Ali H , *Detecting Jute Plant Disease Using Image Processing And Machine Learning* , 3rd International Conference On Electrical Engineering And Information Communication Technology (ICEEICT),Pp. 16, 2016.
[11] Agarwal SK and Kumar P, *Denoising Of a Mixed Noise Color Image Using New Filter Technique* , Proc.-Int. Conf. Comput. Intell. Commun. Networks, Pp.324–32, 2015.
[12] Orillo JW, Dela Cruz J, Agapito L, Satimbre PJ And Valenzuela I, *Identification Of Diseases In Rice Plant (Oryza Sativa) Using Back Propagation Artificial Neural Network* ,International Conference On Humanoid, Nanotechnology, Information Technology, Communication And Control, Environment And Management, 2014.
[13] Archana k S, and Sahayadhas A*Comparison Of Various Filters For Noise Removal In Paddy Leaf Images* ,International Journal Of Engineering and Technology, 7 2.21, 372-374, 2018.
[14] Rani, k. S., And Rao, D. N , *A Comparative Study Of Various Noise Removal Techniques Using Filters* , Journal Of Engineering And Technology, 7(2), 47-52, 2018.
[15] Kaur, S , *Noise Types And Various Removal Techniques* ,International Journal Of Advanced Research In Electronics And Communication Engineering (Ijarece), 4(2), 226-230, 2015.
[16] Halder, A., Sengupta, S., Bhattacharya, P., And Sarkar, A , *Fast Adaptive Decision-Based Mean Filter For Removing Salt-And-Pepper Noise In Images* ,In Computational Intelligence In Pattern Recognition, Springer, Singapore. Pp. 783-793, 2020.
[17] Eckert, D., Vesal, S., Ritschl, L., Kappler, S., And Maier, A , *Deep Learning-Based Denoising Of Mammographic Images Using Physics-Driven Data Augmentation* ,In Bildverarbeitungfür Die Medizin 2020, Springer Vieweg, Wiesbaden.(Pp. 94-100), 2020.
[18] Cruz, M. L, *Full Image Reconstruction With Reduced Speckle Noise, From a Partially Illuminated Fresnel Hologram, Using a Structured Random Phase* ,Applied Optics, 58(8), 1917-1923, 2019.
[19] Das, S., Saikia, J., Das, S., And Goni, N, *A Comparative Study Of Different Noise Filtering Techniques In Digital Images*, International Journal Of Engineering Research And General Science, 3(5), 180-190. 2015.
[20] Diwakar M, Kumar M, *A Review On Ct Image Noise And Its Denoising* ,Biomed Signal Process Control 42:73–88, 2018.
[21] Jain P, Tyagi V , *A Survey Of Edge-Preserving Image Denoising Methods* ,Inf Syst Front 18(1):159–170, 2016.
[22] Zhu, Y., And Huang, C , *An Improved Median Filtering Algorithm For Image Noise Reduction* ,Physics Procedia, 25, 609-616, 2012.
[23] Fan, L., Zhang, F., Fan, H., And Zhang, C , *Brief Review Of Image Denoising Techniques* ,Visual Computing For Industry, Biomedicine, And Art, 2(1), 7, 2019.

Edited by: Rajkumar Rajasekaran
Received: Mar 27, 2020
Accepted: Aug 9, 2020


Scalable Computing: Practice and Experience, ISSN 1895-1767, http://www.scpe.org
© 2020 SCPE. Volume 21, Issues 4, pp. 611–623, DOI 10.12694:/scpe.v21i4.1759

REAL-TIME BIG DATA ANALYTICS FRAMEWORK WITH DATA BLENDING APPROACH FOR MULTIPLE DATA SOURCES IN SMART CITY APPLICATIONS

MANJUNATHA S*AND ANNAPPA B†

Abstract. Advancement in Information Communication Technology (ICT) and the Internet of Things (IoT) has to lead to the continuous generation of a large amount of data. Smart city projects are being implemented in various parts of the world where analysis of public data helps in providing a better quality of life. Data analytics plays a vital role in many such data-driven applications. Real-time analytics for finding valuable insights at the right time using smart city data is crucial in making appropriate decisions for city administration. It is essential to use multiple data sources as input for the analysis to achieve better and more accurate data-driven solutions. It helps in finding more accurate solutions and making appropriate decisions. Public safety is one of the major concerns in any smart city project in which real-time analytics is much useful in the early detection of valuable data patterns. It is crucial to find early predictions of crime-related incidents and generating emergency alerts for making appropriate decisions to provide security to the people and safety of the city infrastructure. This paper discusses the proposed real-time big data analytics framework with data blending approach using multiple data sources for smart city applications. Analytics using multiple data sources for a specific data-driven solution helps in finding more data patterns, which in turn increases the accuracy of analytics results. The data preprocessing phase is a challenging task in data analytics when data being ingested continuously in real-time into the analytics system. The proposed system helps in the preprocessing of real-time data with data blending of multiple data sources used in the analytics. The proposed framework is beneficial when data from multiple sources are ingested in real-time as input data and is also flexible to use any additional data source of interest. The experimental work carried out with the proposed framework using multiple data sources to find the crime-related insights in real-time helps the public safety solutions in the smart city. The experimental outcome shows that there is a significant increase in the number of identified useful data patterns as the number of data sources increases. A real-time based emergency alert system to help the public safety solution is implemented using a machine learning-based classification algorithm with the proposed framework. The experiment is carried out with different classification algorithms, and the results show that Naive Bayes classification performs better in generating emergency alerts.

Key words: Big Data, Data blending, Preprocessing, Real time analytics, Public safety, Smart city

AMS subject classifications. 68T05

1. Introduction.

Technological advancement in data analytics is changing the business process by enabling faster and better decisions based on real-time analytics. When data analysts can harness useful insights from data faster, it has a significant advantage in reducing costs, increasing efficiency, and profit. Extracting valuable insights from raw data in real-time is critical for many real-time applications. The demand for real-time analytics is high in recent days in various fields where data-driven solutions are being used. In most of the data-driven solutions, real-time processing of data for making timely decisions can enhance the quality of service, improve the accuracy of predictions, and help in making early decisions. It is challenging for the data analysts to process data from multiple sources in real-time for a specific analytical solution. The outcomes of the analytics are more effective and accurate when more data from appropriate data sources get processed for a specific analytical solution.

In smart cities, the data gets generated continuously in real-time from different applications, devices, and social media on a large scale. The data generated from various smart applications and smart devices are of different types and formats. The valuable insights derived from the data generated within the city helps effective management and administration of the city services. The data-driven solutions are widely used in some of the smart city applications such as smart traffic management, smart parking systems, smart environment,

*Department of Computer Science and Engineering, National Institute of Technology Karnataka Surathkal, India-575025 (manjunatha.msh@ieee.org).

†Department of Computer Science and Engineering, National Institute of Technology Karnataka Surathkal, India-575025 (annappa@ieee.org).

smart policing, smart healthcare, etc. A vast amount of user-generated content within the city are analyzed for finding useful insights to enhance the services and performance of smart city applications. In turn, finding valuable data patterns in real-time greatly help in improving the performance of smart city applications and quality of service. The advancement in digitization in recent days opens up possible creation of user-generated content from various sources — analytics on all available data as input to discover valuable data patterns results in finding accurate data-driven solutions. For example, smart policing applications for public safety; collect the user-generated contents from different social networking applications and any specific smart application designed for the same purpose. Real-time analysis of the data collected from these data sources helps in early predictions and monitoring of crime-related incidents within the city. It is a challenging task for analysts to use multiple data sources with different properties in a specific data-driven solution.

The proposed work is aiming to design a real-time big data analytics framework with a data blending approach for data preprocessing when multiple data sources as input. The motivation for this work is, when target data spread across multiple sources, analysts must use all possible data sources to find hidden patterns and discover valuable insights for more effective solutions. In smart policing applications for public safety in smart cities, data analysts are collecting the data within the city from different sources for finding crime-related data patterns that further used for crime detection and administrative decisions for crime prevention. In this scenario, many popular social media platforms used by the public and any specific applications offered by the police department are the major data sources of information. All these data generated within the city are analyzed for making better decisions or more accurate predictions for crime detection and prevention. The rapid growth of digitization in various fields created ample space for more and more new data sources, which are added regularly in the form of social media and smart applications. It is a challenging task for the data analysts to use additional data sources in their existing data-driven solutions with minimal cost and time. The proposed framework is an attempt to address the issue of blending data from multiple data sources for real-time data analytics in smart city applications for public safety.

In this work, a smart policing system for public safety in a smart city is considered where different data sources from social media and smart application data are collected and analyzed for generating emergency alerts using the proposed framework. The data blending approach proposed with the framework helps the analysts to add up any of the related data-sources of interest in the existing analytics framework. In the proposed mechanism, the analysts can use all identified data sources in real-time for making better analytics solutions without any additional delay. Section 2 of this article describes the importance of real-time analytics and data preprocessing in it, and related work carried out. Section 3 describes the design and implementation of the proposed framework for real-time analytics with a data blending mechanism for a smart policing system use case. Section 4 of this article is a discussion of experimental work along with results, and finally, Section 5 concludes and summarizes the proposed framework along with future work for further enhancement.

2. Backgound.

2.1. Real-time Data Analytics. Real-time analytics and streaming analytics have become more prevalent in big data applications, where timely decisions are more crucial and beneficial. It is a need in many big data applications to generate results in real-time for better performance. In Real-time analytics, data processed at the very moment it arrives into the system rather than processing at a later stage from data storage wherein it gets stored. Some applications generate data continuously in real-time, which affects the outcome of the analytical results. For example, the applications for environmental monitoring need to collect real-time data such as temperature, humidity readings continuously. Real-time analytics helps the analysts to glean essential insights quickly and find the data-driven solution instantly. The critical part in real-time big data analytics is extracting valuable information from the incoming data as and when it enters an existing big data infrastructure. The predictions or decision making in these applications are affected by both historical data stored and real-time data gets generated continuously. Real-time analytics technologies and processes must be capable of manage and analyze the data as and when it enters the database.

Big data analytics research resulted in many real-time and streaming analytics tools such as Apache Storm, Apache Spark, and Apache Flink. In Spark, data stream represented as a sequence of Resilient Distributed Datasets (RDDs) and in-memory computing feature enables it to compute data batches quicker than Hadoop. Apache Storm is another distributed computing framework for streaming data processing, but there are limited

streaming machine learning libraries are available. Apache Flink is brought as an alternative for Spark with its defining traits as real-time processing and low data latency. Spark processes chunks of data known as RDDs, whereas Flink can process rows after rows of data in real-time.

2.2. Smart city and Public safety. Urban development is a crucial issue for any government as the urban population is increasing around the world in recent days [1]. Smart cities are one of the frontline projects in most of the countries for urban development. While 'smart city' means different things to different people, one common thing everyone agrees on is that digital technologies are used in the smart city to improve the quality of the services within the city. The technological growth in digital and communication media incorporated in the smart city for better services within the city. Internet of Things (IoT) and Information Technology (IT) help to accomplish many smart applications in smart cities. It leads to generating a massive amount of data in distinctive formats. The advancement in big data technologies exploits to analyze the data generated within the city for enhanced services in the smart city. The smart applications used in smart cities such as smart traffic, smart environment, smart governance, smart agriculture, smart health-care generates a large amount of data, which can be used to extract useful insights to enhance the quality of the service. The different types of sensors, video surveillance cameras, and smart mobile applications used by smart city applications are the major source for generating data. The smart applications created for smart city services and many popular social media applications are cause for generating large amounts of user-generated content within the city, which helps in enhancing the quality of service within the city.

Smart policing solutions are widely used for public safety due to the technological adoption of the Internet of Things and Cloud [2]. Transport and traffic security, infrastructure security, emergency services for fire and medical, crisis management, and law enforcement are the most common solutions in smart city public safety services. Real-time information is crucial for better implementation of such applications to provide timely services. Real-time crime centers are established in some cities to keep the cities safe by monitoring the activities in real-time within the city. Intelligent analytics on real-time data generated within the city is the solution for smarter crime responses, monitoring, and prevention. Law enforcement agencies are switching towards predictive policing for their routine and investigation procedures. It involves advanced analytics techniques to predict what and where an incident likely to happen. Predictive policing in real-time can help in early monitoring of the crime and preventing it before it happens.

2.3. Data Preprocessing in Big Data Analytics. Data preprocessing is a crucial and significant phase within the data analytics process [3]. The raw data used as input into the analytics system is likely to be noisy, inconsistent, and imperfect. The data preprocessing phase is the set of techniques used for making raw data as analytics-ready in the data analytics process [4]. The preprocessing phase in real-time data analytics becomes challenging, where the raw data enters into the data collection system continuously. The critical part in data preprocessing includes mainly two concepts, such as data cleaning and feature engineering. Data preprocessing is essential for achieving better accuracy and performance in the analytical model. Most of the effort made in the preprocessing of big data is mainly focus on developing feature selection methods [5]. Noise reduction, instance reduction, and missing values imputations are the important preprocessing methods focused by data analysts. When the data is collected from various sources, combining this to form a consistent data is an important process in making the data ready for analysis. Data blending is a technique in preprocessing for combining data from multiple sources to create a common data set for decision-making [6]. It is one of the quick methods to extract common information from multiple data sources.

2.4. Related Work. Big data has been a progressive aspect of the industries due to data explosion, which inhabited all business categories from the past few years. The academic and industry research produced many applications using real-time big data analytics in the area of healthcare, fraud detection, smart grid, social media analytics, sensor data analytics, and many more. The majority of the work is on social media data analytics for knowledge discovery. Congosto et al. [7] proposed a cost-effective framework to perform micro-blogs analytics in twitter stream data. An event detection system is incorporated to detect important events in real-time from twitter data streams is proposed by Hasan et al. [8]. Similar work has done for city event detection for London city using twitter data streams by Zhou et al. [9]. An adaptive filtering algorithm is proposed by Fan et al. [10], to filter interesting tweets from the twitter stream concerning user interest

profiles. A medical emergency system is proposed by Rathore et al. [11], to find the intelligent decision by analyzing medical data collected from sensors attached to the human body. Charlie Catlett et al. [12] proposed a spatio-temporal crime forecasting model to detect high-risk crime regions using an auto-regressive model. Pina-Garcia et al. [13] proposed that data generated from different social media platforms can be integrated to enhance big data-driven models for crime prediction. Harnessing multi-source data about public sentiments and activities for informed design is proposed by Linlin You et al. [14] that addresses the process from data collection to data visualization. Zheng Xu et al. [15] proposed a framework for collecting and analyzing data from social media and surveillance cameras to describe public safety events.

Similarly, many works have been attempted for the safety of the city using real-time data. The real-time event detection system is designed to detect and classify the events for high way traffic data by Khazaei et al. [16]. An event detection system designed for real-time data analytics of IoT enabled communication system by Ali et al. [17]. A real-time monitoring system using social big data is proposed for disaster management by Choi and Bae [18]. A real-time road traffic monitoring system is proposed by Wang et al. [19], estimating online vacancies on the road using a traffic sensor data stream. Real-time data analytics for predictions are used in many data-driven applications. A prediction system designed using real-time news data sources to predict future terrorist incidents is proposed by Toure and Gangopadhyay [20]. A predictive model is proposed by Zhang and Yuan [21] for air quality monitoring by analysis of real-time meteorology data from Beijing city. Some of the work attempted for detecting, and monitoring of the crimes are forecasting crime trends in urban areas by Cesario et al. [22], spectral analysis of crimes in the city by Venturini and Baralis [23], Parvez et al. [24] and an intelligent solution for the smart city using real-time crime analysis by Ghosh et al. [25].

From the past few years, the research articles on streaming data analytics have been highlighted the need for the preprocessing mechanism of the data collected in the streaming manner for the analytics [26]. The different technological frameworks have been used for streaming data analytics. Fernando Puentes et al. [27] analyzed characteristics of different open source frameworks available for streaming data analytics. Finding proper preprocessing mechanisms for data in motion in real-time is essential in achieving better performance. Whenever analytics is performed immediately after data is collected, the preprocessing mechanism to be done as soon as the data enters the system. Data preprocessing is becoming a critical methodology for knowledge discovery in streaming data [5]. The authors identify the role of data preprocessing methodologies in the streaming analytics system for better performance. The critical preprocessing methodologies include data reduction, incremental learning, concept drift detection, and adaptation, and stream discretization algorithms. The data preprocessing with manual intervention is of no use for any better analytics system [28]. The authors use an adaptive preprocessing mechanism for prediction on real-time sensor data.

3. Design and Implementation.

3.1. Real-time Analytics Framework. Real-time Big data analytics is an iterative process. Any real-time analytics design is broadly based on one of the two important data processing architectures proposed by Lambda [29] and Kappa architecture [30]. Figure 3.1 shows the overview of Lambda architecture for real-time analytics. Lambda architecture is a data processing technique in a big data environment consisting of three layers, namely batch layer, serving layer, and speed layer. In this architecture, the data enters into the system is passed through both batch layer and speed layer. The batch layer is responsible for managing the master dataset and pre-compute the batch views, while the speed layer is responsible for calculating real-time views with real-time data. The serving layer task is to index and expose the pre-computed batch views for queries to be executed. A query to be executed can be answered through combined results of both batch and real-time views. Kappa architecture is a data processing architecture that is an alternate and simplification of lambda architecture. This architecture targets only on data as a stream, so batch layer as in lambda architecture is removed. It comprises of real-time layer and serving layer. Real-time input data is streamed through the real-time layer and results of which passed into the serving layer for queries to be executed.

In the proposed work, the real-time data from multiple sources are analyzed to discover useful insights for making real-time decisions and predictions. The data processed at the moment is stored for further use in predictive models in later stages. The framework is designed based on Lambda architecture. The data is ingested into the analytical system immediately after it gets generated at the particular source and preprocessed it to make it ready for further analytics. The data from identified sources streamed through the real-time layer

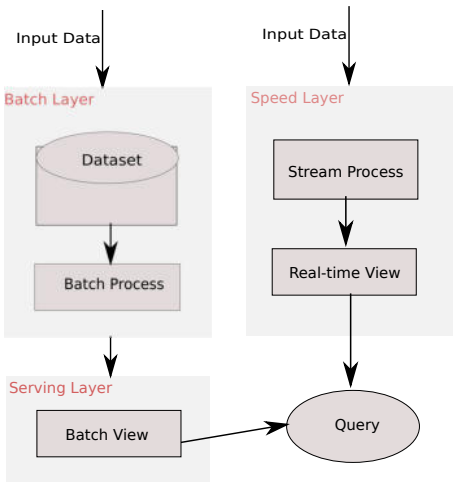


FIG. 3.1. *Real-time analytics architecture*

where it is processed and passed into the serving layer. The real-time queries to be executed using the real-time views of the serving layer. The data stored for further use is executed using the batch views along with the real-time views in the data-driven models.

3.2. Proposed Design. The proposed design for real-time analytics using multiple data sources is as shown in Figure 3.2. The real-time data from identified data sources are collected and processed for a specific

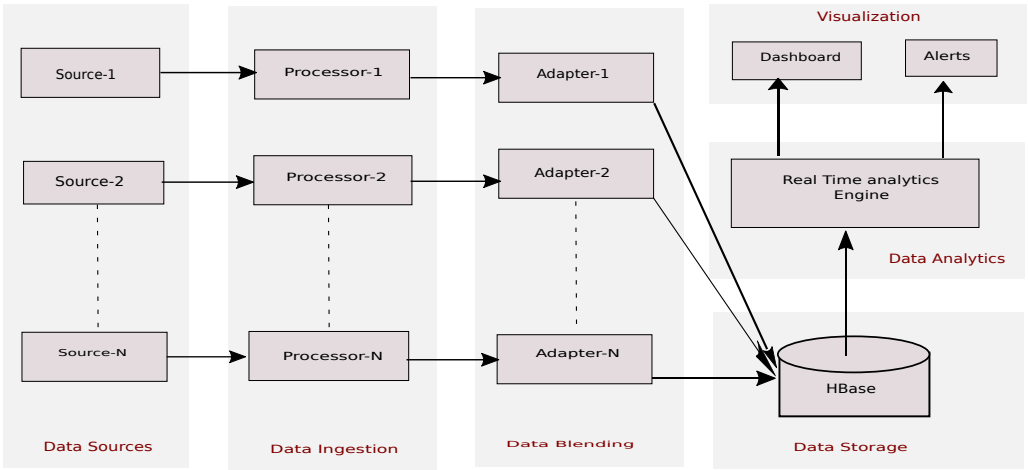


FIG. 3.2. *Proposed framework for Real time big data analytics*

data-driven solution. When the input data required for the analysis are identified at different sources, it is essential to use all available data in the process to increase the accuracy or performance of the data-driven solution. Here raw data from multiple sources in real-time are used as input data for the specific data-driven solution. The data is ingested from a particular source as soon as it is generated at the source. The data ingestion phase consists of different data ingestion processors for each input data source used. Each data ingestion processor is comprised of a real-time data ingestion mechanism for the specific data source. The processor also includes an initial stage of preprocessing mechanism for filtering of data of interest for the desired analytical solution. The data ingested and filtered at each source is passed through a data blending

mechanism. The purpose of the data blending mechanism is to integrate the data from different sources to a single common dataset for further analysis. The data blending phase consists of separate adapters for each source, which reads the input from respective data ingestion processors. Each adapter is a real-time task that can read the data immediately when it filtered out from the respective processor. Data blending is performed to extract the common data of interest from each source and append it to a single dataset. The blended data is used in the next stage for analysis to find meaningful patterns in real-time. The outcome of this helps in making data-driven decisions such as emergency alerts of crime incidents, identifying crime hotspots, and prediction of possible occurrences of crimes in the city.

4. Experimental Evaluation. Figure 4.1 illustrates the detailed framework and the flow of the real-time analytics process. Three different data sources identified as input data for experimental work, where data

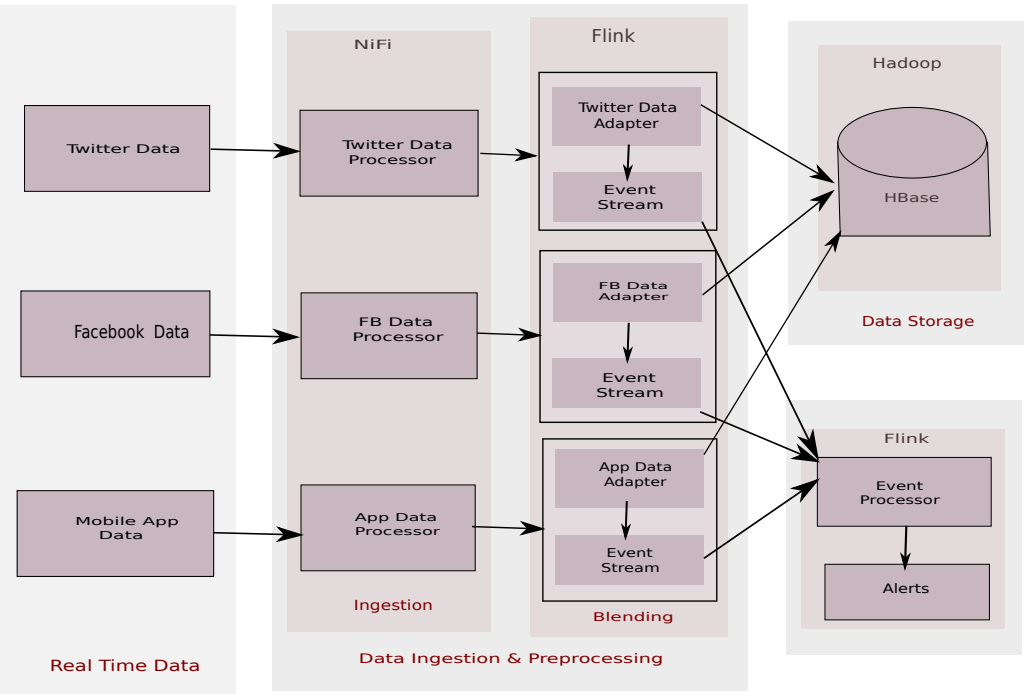


FIG. 4.1. *Data blending of real time data from multiple sources*

collected in real-time. The data from each source is ingested and filtered by respective ingestion processors and then passed into corresponding adapters for data blending mechanism. Each adapter is comprised of a mechanism to read the data from the respective processor whenever new data arrives at the processor. When the processor passes the new data to the adapter, a real-time job is executed to preprocess the data with a data blending mechanism and store it on the HBase table on top of Hadoop. Here HBase supports real-time read/write access to the data. The preprocessed data stored on the HBase table is a blended data from multiple sources that can be used in the further process for a real-time analytical solution to make desired data-driven decisions. A real-time emergency alert mechanism also introduced during the data blending mechanism to generate alerts on any emergency incidents. The contents of the ingested data from the processors are verified for any topics related to the emergency events. The input data stream is processed by the event processor to generate any emergency alerts.

The critical approach in the proposed work is the data blending mechanism for preprocessing the data. Here data from multiple sources prepared ready for further analytics process. Data preprocessing is a critical step in the analytics process as it takes the maximum time of the entire process. The quality of the analytical result purely depends on the quality of the data used. Preprocessing the input data with appropriate preprocessing mechanisms is necessary. In the proposed work, analytics to be performed in real-time where it is a challenging

task to preprocess the data as the data arrives continuously at data collection end. Preprocessing is to be done whenever new data ingested into the system. In the proposed mechanism, the data from multiple sources are used as input, whereas kinds of literature referred to are targeting the single source of data. When data from various data sources used in the analytics, each source may consist of data in different formats, structures. The proposed design mainly consists of three components like processors for data ingestion, adapters for data blending mechanism, and event processor to generate emergency alerts. Data ingestion processors are responsible for data collection in real-time and also the necessary filtering of expected data in real-time from the data sources. Adapters for data blending mechanism are to preprocess the data for making it ready for analytics and append into blended data. The purpose of an event processor is to analyze the incoming data streams sent from the adapters to detect any emergency incident in the city.

Data Ingestion Processors. For the experimental work, real-time data from Twitter, Facebook posts, and citizen complaint data from the mobile application are used as input data sources. For real-time data collection, separate data ingestion processors are written for each of the data sources, where each processor is performing the task of real-time data ingestion along with the initial stage preprocessing of data. The incoming data is filtered in the initial stage of preprocessing to extract only the data related to crime. For example, in the case of Twitter data, only the tweets related to the crime are considered as required data for our analytics process, and other tweets are discarded. The workflow of each processor is as shown in Figure 4.2. Each

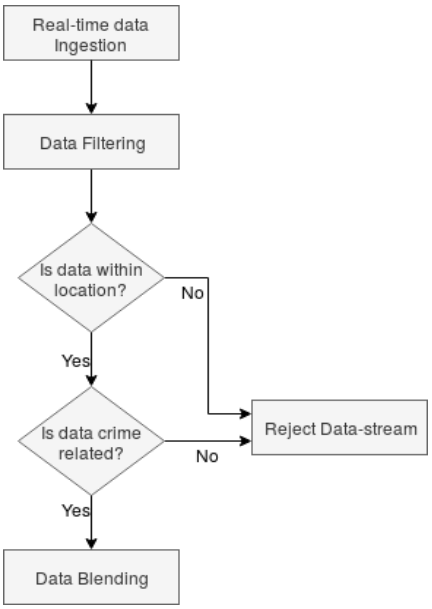


Fig. 4.2. *Processor for Crime data filtering*

processor is configured for the respective data source with the data ingestion mechanism of real-time data. The processors are responsible for real-time data ingestion into the analytical system as and when new data generated at the source. In data ingestion processors, data filtering is done to refine the data to select only the crime-related data of the specific city and discard any other unrelated data streams. If the values of location in the incoming data match with the city location values, then the data is considered for the further process; otherwise, data discarded directly. Further, the actual content of the accepted data is verified for having any information related to crime.

The incoming data streams filtered based on city location values and further verified for whether contents of the incoming data related to crime or not. A knowledge-base is created, which consists of crime-related words and phrases to compare with the incoming data to find out any crime-related information is present in the incoming data. For experimental work, 565 words and phrases which are related to different categories of crimes are used in the knowledge base with the help of Cambridge and Macmillan dictionaries. The contents

of the knowledge base are used to verify the crime-related information in the contents of the incoming data stream. If any matching information present in the incoming data, then the data stream is passed to the next stage of preprocessing. The outputs of the processors are passed through respective adapters for data blending mechanism.

Data blending mechanism. Real-time data ingested from each source by respective data ingestion processors are passed to respective adapters. Each adapter process the incoming data from the respective processor with the data blending mechanism. These adapters are the real-time jobs written using Apache Flink as the real-time processing tool. With its streaming architecture, Apache Flink helps to process the events in real-time with consistently high speed with low latency. In this experiment, all three data sources used for the analytics streamed from the data ingestion mechanism are in javascript object notation (JSON) format, but the structure of the data is different in each source. Data blending mechanism is the process of combining the data from the multiple sources into a single dataset. Data blending is a different mechanism than the data integration process. Data blending is about working with multiple data sources by preparing them and joining them together for a specific use case, whereas data integration typically stores as a single source in the data warehouse for a user to access.

The proposed data blending mechanism is implemented with adapters to process data streams from the respective data ingestion processors. The adapters are written as Flink jobs that can read new data from the respective processor as and when it arrives. A blending mechanism is a process of combining the data received from the different processors and store it on a particular data storage for further use. Here, we use HBase to store the blended data received from the adapters. We also added an emergency event process mechanism within the adapters. During the data blending mechanism, incoming data stream contents are observed for data patterns related to any emergency events. Such data streams are passed to an emergency event processor to generate emergency alerts.

The working of the Twitter data adapter is as shown in Algorithm 1. Here, an adapter can read the

Algorithm 1 Twitter data Adapter

1. Read datastream from Twitter data processor
2. Parse the datastream to select target fields (created-at, name, location, text)
3. Send the selected fields of datastream to event processor
4. Write the values of selected fields to new row in HBase table BLENDED_TABLE as
   valueof(source-id) <- 1
   valueof(created-at) <- Time
   valueof(name) <- User
   valueof(location) <- Location
   valueof(text) <- Contents
5. Repeat Step-1

twitter data ingested and filtered at the respective processor immediately once it is available. The adapter for the twitter data source is written as a Flink job, which reads each new input JSON file from the output of the twitter data ingestion processor. This JSON file is parsed to filter the target fields, which are the useful information to be stored on blended data for further analytics. In the JSON file from the twitter data source, the values from specific fields such as created-at, name, location, and text are considered for the analytics at the next stage. This information from each of the incoming data streams used to store on the HBase table. Another information source-id is stored as '1' for all the new appended rows from the twitter adapter. The source-id is to be used in the further process to find the identity of the data source. During the blending mechanism in the adapter, contents of the text in the tweet are observed to identify the target events for emergency alerts. Each incoming data is passed through an event processor to generate any emergency alerts.

The working of the adapters for the other data sources used in the experiment is also similar to the twitter data adapter. The structure of incoming data is different with different attribute names in each data source. The working of the facebook data adapter shown in Algorithm 2 is similar to the adapter for Twitter data, but the target fields selected are created-time, id, location, and message. The values of these attributes in the

Algorithm 2 Facebook data Adapter

1. Read datastream from Facebook data processor
2. Parse the datastream to select target fields (created-time, id, location, message)
3. Send the selected fields of datastream to event processor
4. Write the values of selected fields to new row in HBase table BLENDED_TABLE as
   valueof(source-id) <- 2
   valueof(created-time) <- Time
   valueof(id) <- User
   valueof(location) <- Location
   valueof(message) <- Contents
5. Repeat Step-1

input data are stored on the blended table on HBase. In this case, the source-id is stored as '2' for all new rows appended on blended data. During this process, the data stream with the selected attributes is passed through an event processor to detect any emergency events. Similarly, for the third data source used, an application data adapter is added where the attributes such as created-time, complaint-id, incident-location, and description are selected for further process. For this data source, source-id as '3' is assigned for all new rows to append on the blended table. Here the contents of 'message' in the input data are used for finding the patterns related to emergency events, as explained in the twitter data adapter and facebook data adapter. Similarly, one can add any other data source available for the analysis.

Event Processor. The purpose of the event processor is to process the event streams passed from the adapter to find any emergency incidents. The event processor consists of a machine learning-based classification model to generate emergency event alerts from the incoming data stream. A training model is developed by using information about different categories of crime incidents such as fire incidents, vehicle accidents, robbery, rape, murder, and gang-war. The initial training model is created using the data related to these six categories of crime incidents data. This training data set is updated regularly as the model is tested with new incoming data streams. The contents of the newly arrived data stream from any of the three data adapters are verified for any emergency incidents. When such incidents are detected during the process, an emergency alert gets generated to help in taking appropriate action by the law enforcement authorities.

As and when the new data stream is passed to the event processor, the content of the data is processed for selecting the topic feature by adopting Latent Dirichlet Allocation (LDA) [31]. Initially, non-English contents are filtered out by using a language detection library, and then stop words are filtered out from the contents. Latent Dirichlet Allocation (LDA) is used to train a topic model that can output the distribution of topics. Then, the classification model developed in the event processor is used to find any emergency incident. This model has experimented with the most popular classification algorithms to choose the better one for the most appropriate results.

In this work, the most commonly used classification algorithms in streaming data analytics such as the Naive Bayes (NB) classifier, Support Vector Machines (SVM) classifier, Logistic Regression (LR), and Random Forest (RF) algorithms are used. NB classifier is a probabilistic classification algorithm based on the application of Bayes's theorem [32]. The model assumes that the presence of a specific feature is unrelated to the presence of any other feature. SVM classifier [33] is based on separating hyperplane according to which new samples are classified. Logistic Regression (LR) is a linear classifier that measures the relationship between the dependent variable and independent variables by determining the probabilities using a logistic function [34]. Random Forest (RF) is based on the forest construction procedure where features as at nodes grow like branches of a tree, finally combining all trees form a Random Forest model [35]. To evaluate the performance of the model, frequently used three statistical metrics like accuracy, precision, and recall are used. Out of the four different classifiers used, the NB classifier gives the most accurate results. Hence this classifier is used to generate emergency alerts in the proposed system. A detailed comparison of the classifier is provided in the next section.

5. Result and Discussion. The performance of the four different classifiers used in the experiment for emergency events classification is as shown in Figure 5.1. The experiment targeted for emergency incidents

by considering the six different categories of crimes. The performance metrics are computed for each category of crime incidents. Then, the overall measure is calculated as the average of the per class measure. Here NB classifier achieves a higher accuracy of 73%, which is a 3% improvement over RF, 5% improvement over SVM, and 8% improvement over LR classifier.

The proposed data blending mechanism helps the analysts to collect more input data in real-time. Figure 5.2 shows the observations after the blending mechanism. Figure 5.2.(a) represents the comparison of the data appended on the blended table from each data source. The values for each data source used in the experiment are calculated using source-id in the blended table. For each source-id, the total number of data updated at an hourly basis is observed. The consolidated data appended at an hourly basis is considered as data from multiple sources. The x-axis represents each hour of execution of the experiment, and the y-axis represents the number of data rows appended on the blended table related to the respective source. Similarly, Figure 5.2.(b) shows the number of crime data of different categories from different sources in the observation at a particular period. When data used from multiple sources in the experiments, analysts can benefit from processing more data to achieve better analytical results.

The proposed mechanism is beneficial whenever any new data source is available for the analytics. If any real-time data source to be considered as an additional input in the existing experiment, then one can easily

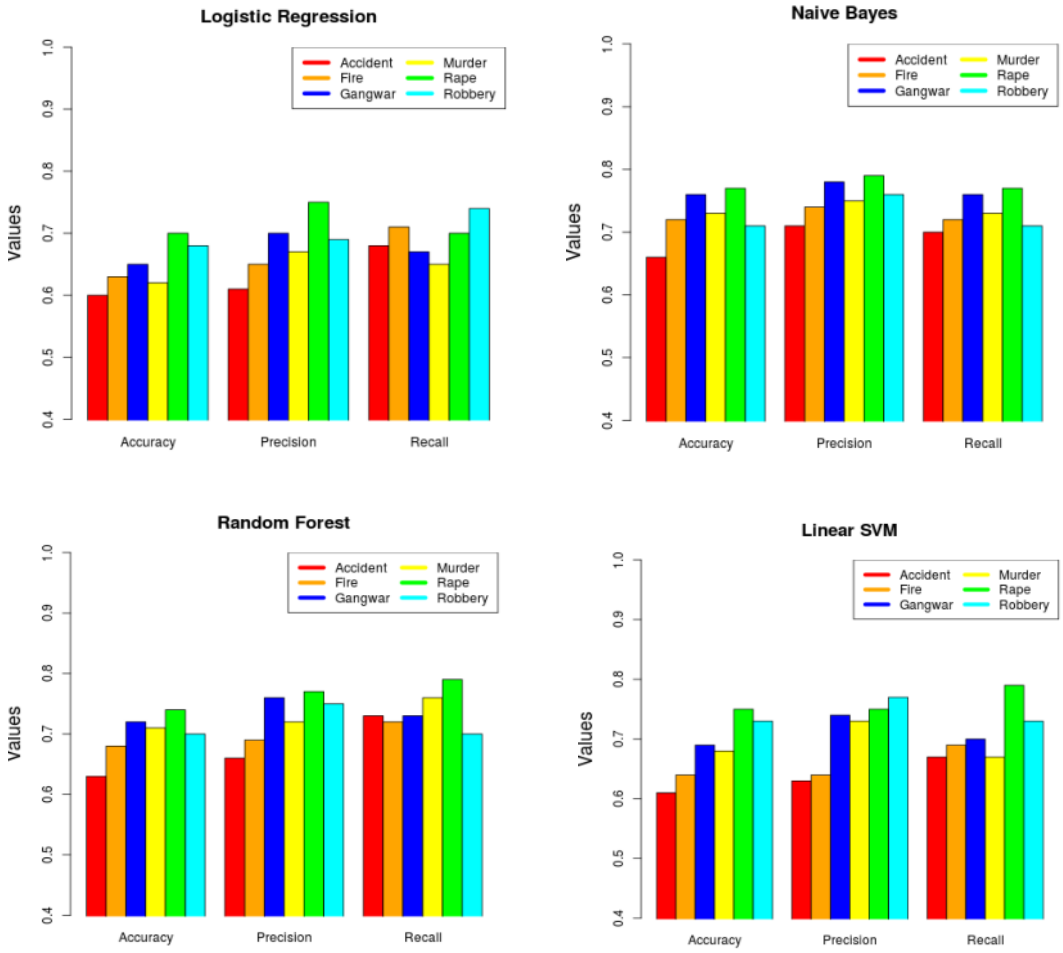
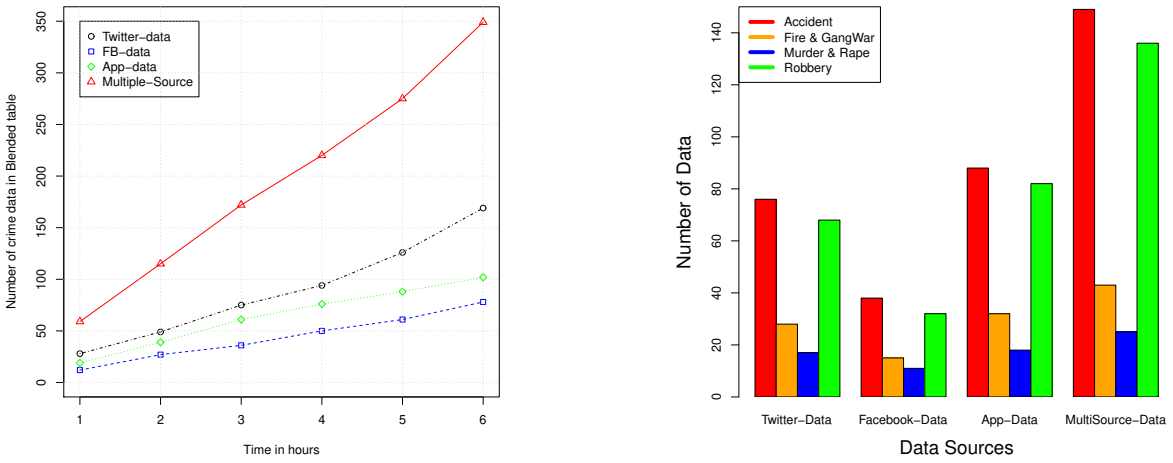


FIG. 5.1. *Performance Comparison of Classification Algorithms for Emergency Alerts*



(a) Hourly-data *(b) Source-wise data*

FIG. 5.2. *Data blending of real-time data from multiple sources*

consider it for analytics by adding a new processor and an adapter for the particular data source. The new processor to be added must consist of a mechanism for ingesting the identified data source with a preprocessing mechanism, as explained in the earlier sections. Also, an adapter to be added with a data blending mechanism, as discussed in the earlier section. Each of the new data sources considered gives additional input to the event processor for emergency alerts. More and more input data collection in real-time helps the analysts to increase the performance of the analytics outcome.

6. Conclusions and future work. Real-time data analytics is invaluable in many data-driven applications for quick response and actions. Public safety is one of the key services in smart city applications, where timely decisions and predictions are much beneficial for detecting and preventing crimes. In real-time data analytics, it is crucial to perform the entire analytics process as quickly as possible. In the data analytics process, the majority of the time spent for preprocessing the data to make it prepare as analytics-ready. In real-time analytics, whenever new data arrives in the data collection phase, it must be preprocessed and analyzed for a desired analytical solution without much delay in the entire process. Collecting the maximum data for the analysis helps in achieving better outcomes. Hence, it is essential to use multiple data sources for input data in analytics to find much better and accurate outcomes. The real-time analytics framework with the data blending approach proposed in this work is appropriate to preprocess the data from multiple sources in real-time. A real-time event processing mechanism is proposed for emergency alerts to any such incidents within the city. Analytical solutions such as predictions and data-driven decisions are possibly more accurate when all available data are used instead of a single data source. The proposed mechanism is much flexible to add any new data source to be used for the analytics with the existing experimental setup.

The future work includes adopting the proposed framework with more number of input data sources. The input data used from all the data sources in the proposed work are text data ingested in the JSON format, thus future work of real-time analytics targets to use different data sources with different types of data formats and structures. The proposed data blending mechanism can be incorporated with any other data-driven applications where one can use input data from multiple sources. The classification model developed for generating emergency event alerts can be improved further for achieving more accuracy. The classification model is developed by selecting the better performing algorithm by comparing the four popularly used classification algorithms in streaming data. The future work is to use a few more algorithms for any better performance with the proposed mechanism. Further research focused on using the proposed mechanism for real-time crime hotspots predictions for public safety in the smart city. It also intended to extend this experimental setup in

the crime predictions when real-time data used along with the historical data.

Acknowledgement. The authors would like to thank Ministry of Electronics and Information Technology (MeitY), Government of India, for their support in a part of the research.

REFERENCES

[1] S. Dirks, C. Gurdgiev, M. Keeling, *Smarter cities for smarter growth: How cities can optimize their systems for the talent-based economy*, in: IBM Institute for Business Value, May 2010.
[2] Market-Research-Report, *Public safety solution for smart city- global forecast to 2023*,Tech. rep., Information and Communication Technology Press Release, July 2018.
[3] D. Pyle, *Data Preparation for Data Mining*, 1st Edition, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1999.
[4] S. Garcia, S., J. Luengo, F. Herrera, *Data Preprocessing in Data Mining*,Springer Publishing Company, Incorporated, 2014.
[5] S. Garcia, S. Ramirez-Gallego, J. Luengo, J. M. Benitez, F. Herrera, *Big data preprocessing: methods and prospects*, Big Data Analytics 1 (1)(2016) 9. doi:10.1186/s41044-016-0014-0..
[6] Alteryx, *The definitive guide to data blending*, White Paper.
[7] M. Congosto, P. Basanta-Val, L. Sanchez-Fernandez, *T-hoarder: A framework to process twitter data streams*, Journal of Network and Computer Applications 83 (2017) 28 – 39. doi:https://doi.org/10.1016/j.jnca.2017.01.029.
[8] M. Hasan, M. A. Orgun, R. Schwitter, *Real-time event detection from the twitter data stream using the twitternews+ framework*, Information Processing and Management 56 (3) (2019) 1146 – 1165. doi:https://doi.org/10.1016/j.ipm.2018.03.001.
[9] Y. Zhou, S. De, K. Moessner, *Real world city event extraction from twitter data streams*, Procedia Computer Science 98 (2016) 443 – 448, the 7th International Conference on Emerging Ubiquitous Systems and Pervasive Networks (EUSPN 2016)/The 6th International Conference on Current and Future Trends of Information and Communication Technologies in Health-care (ICTH-2016)/Affiliated Workshops. doi:https://doi.org/10.1016/j.procs.2016.09.069..
[10] F. Fan, Y. Feng, L. Yao, D. Zhao, *Adaptive evolutionary filtering in real-time twitter stream*, in: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, CIKM '16, ACM, New York, NY, USA, 2016, pp. 1079–1088. doi:10.1145/2983323.2983760.
[11] M. M. Rathore, A. Ahmad, A. Paul, J. Wan, D. Zhang, *Real-time medical emergency response system: Exploiting iot and big data for public health*, Jornal of Medical System. 40 (12) (2016) 1–10. doi:10.1007/s10916-016-0647-6.
[12] Charlie Catlett, Eugenio Cesario, Domenico Talia, Andrea Vinci, *Spatio-temporal crime predictions in smart cities: A data-driven approach and experiments*, Pervasive and Mobile Computing, 53, 62-74, (2019), https://doi.org/10.1016/j.pmcj.2019.01.003.
[13] C.A. Pina-Garcia, L. Ramirez-Ramirez, *Exploring crime patterns in Mexico City*, Journal of Big Data, 6, 65, (2019), https://doi.org/10.1186/s40537-019-0228-x.
[14] L. You, B. Tunçer, H. Xing, *Harnessing Multi-Source Data about Public Sentiments and Activities for Informed Design*, IEEE Transactions on Knowledge and Data Engineering, 31, 2, 343-356, (2019),doi: 10.1109/TKDE.2018.2828431.
[15] Zheng Xu, Lin Mei, Zhihan Lv, Chuanping Hu, Xiangfeng Luo, Hui Zhang, Yunhuai Liu, *Multi-Modal Description of Public Safety Events Using Surveillance and Social Media*, IEEE Transactions on Big Data, 5, 4, 529-539, (2019),doi: 10.1109/TBDATA.2017.2656918.
[16] H. Khazaei, R. Veleda, M. Litoiu, A. Tizghadam, *Realtime big data analytics for event detection in highways*, in: 2016 IEEE 3rd World Forum on Internet of Things (WF-IoT), 2016, pp. 472–477. doi:10.1109/WF-IoT.2016.7845461.
[17] M. I. Ali, N. Ono, M. Kaysar, Z. U. Shamszaman, T.-L. Pham, F. Gao, K. Griffin, A. Mileo, *Real-time data analytics and event detection for iot-enabled communication systems*, Journal of Web Semantics 42 (2017) 19–37. doi:https://doi.org/10.1016/j.websem.2016.07.001.
[18] S. Choi, B. Bae, *The real-time monitoring system of social big data for dis- aster management*, in: J. J. J. H. Park, I. Stojmenovic, H. Y. Jeong, G. Yi (Eds.), Computer Science and its Applications, Springer Berlin Heidelberg, Berlin, Heidelberg, 2015, pp. 809–815.
[19] F. Wang, L. Hu, D. Zhou, R. Sun, J. Hu, K. Zhao, *Estimating on-line vacancies in real-time road traffic monitoring with traffic sensor data stream*, Ad Hoc Networks 35 (2015) 3–13, special Issue on Big Data Inspired Data Sensing, Processing and Networking Technologies. doi:https://doi.org/10.1016/j.adhoc.2015.07.003.
[20] I. Toure, A. Gangopadhyay, *Real time big data analytics for predicting terrorist incidents*, in: 2016 IEEE Symposium on Technologies for Homeland Security (HST), 2016, pp. 1–6. doi:10.1109/THS.2016.7568906.
[21] C. Zhang, D. Yuan, *Fast fine-grained air quality index level prediction using random forest algorithm on cluster computing of spark*, in: 2015 IEEE 12th Intl Conf on Ubiquitous Intelligence and Computing and 2015 IEEE 12th Intl Conf on Autonomic and Trusted Computing and 2015 IEEE 15th Intl Conf on Scalable Computing and Communications and Its Associated Workshops (UIC-ATC-ScalCom), 2015, pp. 929–934. doi:10.1109/UIC-ATC-ScalCom-CBDCom-IoP.2015.177..
[22] E. Cesario, C. Catlett, D. Talia, *Forecasting crimes using autoregressive models*, in: 2016 IEEE 14th Intl Conf on Dependable, Autonomic and Secure Computing, 14th Intl Conf on Pervasive Intelligence and Computing, 2nd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/CyberSciTech), 2016, pp. 795–802. doi:10.1109/DASC-PICom-DataCom-CyberSciTec.2016.138.
[23] L. Venturini, E. Baralis, *A spectral analysis of crimes in san francisco*, in: Proceedings of the 2Nd ACM SIGSPATIAL Workshop on Smart Cities and Urban Analytics, UrbanGIS '16, ACM, New York, NY, USA, 2016, pp. 4:1–4:4.

doi:10.1145/3007540.3007544.

[24] M. R. Parvez, T. Mosharraf, M. E. Ali, *A novel approach to identify spatio-temporal crime pattern in dhaka city*, in: Proceedings of the Eighth International Conference on Information and Communication Technologies and Development, ICTD'16, ACM, New York, NY, USA, 2016, pp. 41:1–41:4. doi:10.1145/2909609.2909624.

[25] D. Ghosh, S. A. Chun, B. Shafiq, N. R. Adam, *Big data-based smart city platform: Real-time crime analysis*, in: Proceedings of the 17th International Digital Government Research Conference on Digital Government Research, dg.o '16, ACM, New York, NY, USA, 2016, pp. 58–66. doi:10.1145/2912160.2912205.

[26] S. Ramirez-Gallego, B. Krawczyk, S. Garcia, M. Wozniak, F. Herrera, *BA survey on data preprocessing for data stream mining: Current status and future directions*, Neurocomputing 239 (2017) 39 –57. doi:https://doi.org/10.1016/j.neucom.2017.01.078. URL http://www.sciencedirect.com/science/article/pii/S0925231217302631.

[27] F. Puentes, M.D. Perez-Godoy, P. Gonzalez, M.J. Del Jesus, *An analysis of technological frameworks for data streams*, Progress in Artificial Intelligence(2020), https://doi.org/10.1007/s13748-020-00210-6.

[28] I. Zliobaite, B. Gabrys, *Adaptive preprocessing for streaming data*, IEEE Transactions on Knowledge and Data Engineering 26 (2) (2014) 309–321. doi:10.1109/TKDE.2012.147.

[29] N. Marz, J. Warren, *Big Data: Principles and Best Practices of Scalable Realtime Data Systems*, 1st Edition, Manning Publications Co., Greenwich, CT, USA, 2015.

[30] J. Kreps, *Questioning the Lamda Architecture*, O'reilly, July 2014.

[31] D. M. Blei, A. Y. Ng, M. I. Jordan, *Latent dirichlet allocation*, Journal of Machine Learning Research.

[32] G. H. John, P. Langley, *Estimating continuous distributions in bayesian classifiers*, in: Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence, UAI'95, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1995, pp. 338–345. URL http://dl.acm.org/citation.cfm?id=2074158.2074196.

[33] C. Cortes, V. Vapnik, *Support-vector networks*, Machine Learning 20 (3) (1995) 273–297. doi:10.1023/A:1022627411411.

[34] I. Witten, E. Frank, M. Hall, *Data Mining: Practical Machine Learning Tools and Techniques*, 3rd Edition, Elsevier, 2011.

[35] L. Breiman, *Random forests*, Machine Learning, 5–32, (2001), doi:https:600//doi.org/10.1023/A:1010933404324.

Edited by: Rajkumar Rajasekaran
Received: May 8, 2020
Accepted: Dec 7, 2020


Scalable Computing: Practice and Experience, ISSN 1895-1767, http://www.scpe.org
© 2020 SCPE. Volume 21, Issues 4, pp. 625–635, DOI 10.12694:/scpe.v21i4.1780

IDENTIFICATION OF TOMATO LEAF DISEASE DETECTION USING PRETRAINED DEEP CONVOLUTIONAL NEURAL NETWORK MODELS

ANANDHAKRISHNAN T*AND JAISAKTHI S.M †

Abstract. In this paper, we proposed a plant leaf disease identification model based on a Pretrained deep convolutional neural network (Deep CNN). The Deep CNN model is trained using an open dataset with 10 different classes of tomato leaves We observed that overall architectures which can increase the best performance of the model. The proposed model was trained using different training epochs, batch sizes and dropouts. The Xception has attained maximum accuracy compare with all other approaches. After an extensive simulation, the proposed model achieves classification accuracy better. This accuracy of the proposed work is greater than the accuracy of all other Pretrained approaches. The proposed model is also tested with respect to its consistency and reliability.The set of data used for this work was collected from the plant village dataset, including sick and healthy images. Models for detection of plant disease should predict the disease quickly and accurately in the early stage itself .so that a proper precautionary measures can be applied to avoid further spread of the diseases. So, to reduce the main issue about the leaf diseases, we can analyze distinct kinds of deep neural network architectures in this research. From the outcomes, Xception has a constantly improving more to enhance the accuracy by increasing the number of epochs, without any indications of overfitting and decrease in quality. And Xception also generated a fine 99.45% precision in less computing time.

Key words: Convolutional neural network, Architectures, Accuracy

AMS subject classifications. 68T05

1. Introduction. Deep CNN is a leading area of research in modern age and machine learning and has been already demonstrated and implemented successfully in different crop fields. Next shows the degree of machine learning techniques using specific layers of information processing to obtain and categorize features and evaluate patterns for supervised or unmonitored learning [20]. Sound processes, natural language and image vision, reinforcement were also tested [2]. It was also commonly used to identify objects and rank objects in multiple globe industries such as business,forestry,aerospace, etc. [15]. It has also been commonly used to identify objects and rank objects in multiple globe industries such as company, forestry, aerospace, etc. A number of CNN comparison analyzes with an increased number of margin layers were performed predominantly. Other works in which AlexNet, Google Inception V3, Inception V4, VGG network are shown [14]. Another job is the inner shift of the layer, trying to change the input details into a layer throughout practice. Furthermore, a range of optimization methods have been suggested to address issues properly, as well as to transfer teaching, CNN some technologies [4]. CNN has some methods of optimization that shown in. So that to improve batch standardization. Deep Learning throughout the classification of tomato plant disease image explains expertise in extending skillful picture handling study and implementation to the farming sector that has been shown. It is now possible to use CNN learning models to define Leaf disease and classify plant disease. Regulation of nutrition and plant safety are a major problem for the anticipated population increase in the world. In comparison, it is necessary to recognize plant diseases and to make appropriate comparison steps.In this research, in the assignment of defining and classifying crop disease a science evaluation of state-of the art profound learning designs is carried out [7]. Section 1 reviews related work in the agricultural sector. Section 2 describes the current art of Convolutional techniques and other equipment. Sections 3, 4 discusses the experimental set-up and findings. Section 5, 6 discusses the conclusion, as well as the methodology to achieve this task.

*Research Scholar, School of Computer Science and Engineering, VIT, Vellore, India (anandhakrishnan.t@vit.ac.in).

†Associate professor, School of Computer Science and Engineering, VIT, Vellore, India (jaisakthi.murugaiyan@vit.ac.in (Corresponding Author)).

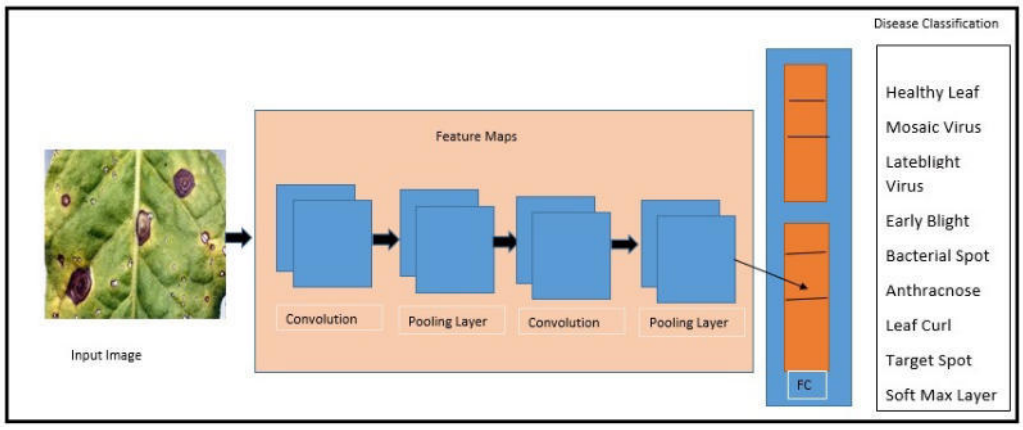


FIG. 1.1. *Deep convolutional neural network architecture*

2. Related Works. Many deep learning methods are used in crop fields and identification of images, including plant disease analysis and control of pests. Methodologies for deep learning and image processing have been expanded. Typical machine learning and deep learning approaches have been commonly adapted in the agricultural area. The paper [15] used a deep learning method in their work on developing a smartphone prognostic system for disease. Using data sources of 54,306 sample images of natural and contaminated plant leaves, they were using CNN to train their model. Sladojevic et al [3] addresses the plant leaves disease identification with 13 types of leafs with 4483 images and they used caffenet model for deeplearning the images for preprocessing they used Croppping the images and data augmentation technique is used and obtained the accuracy of 96.30% compare to the better results than SVM.

CNN were trained to use images to classify 14 species of crops and 26 diseases of the leaf..They assessed CNN's effectiveness for greenhouse and crop and disease classified problem. Two AlexNet and GoogleNet architectures have been implemented [4]. Their model was 99.35% valid. Even though their deep learning system achieved current art results, it was poorly performed when examined on image sets taken under numerous natural conditions. Similarly, [13] suggested that CNN use leaf images to identify plant diseases and validate the model. Their system was able to recognize from healthy leaves 14 various types of plant disease. Sladojevic etal Addresses the plant leaves disease identification with 13 types of leafs with 4483 images and they used caffenet model for deeplearning the images for preprocessing they used Cropping the images and data augmentation technique is used and obtained the accuracy of 96.30% compare to the better results than SVM. Mohanty et al. [14]. It focus on the plant leaves disease identification with 38 types of leafs with 54483 images and they used Alexnet model for deeplearning the images for preprocessing they resized to 224*224. No data augmentation technique is used and obtained the accuracy of 99.30% compare to the better results than other approaches.

But added, plants can be differentiated from their natural environment.On their investigation analysis, they accomplished an average 96.30% validity Probably used deep learning architectures to categorize plant species [6]. In their work, they presents a technique that can use colored images to produce plants and species. In their work, they were using CNN, which was tested on a total of 10,413 images of 22 species and crops.The CNN structure had such a problem in classifying certain plant species, and this is assumed to be limited to a small number of training datasets for those species [3]. The next method, called Deep Fruits, was introduced for tomato image detection in agriculture. They describe the CNN approach to tomato detection in their work using image data.Their intention was to construct an correct, reliable system of fruit identification and recognition,An important element of agricultural components for yield estimation and automatically generated processing.They trained their system and were able in the identification to accomplish an improvement of 0.838 accuracy and recall from the previous work. They trained to identify some fruits, taking four hours to annotate and train the new model per fruit all across the entire process [16]. Supervised learning techniques were applied same in classes on crop disease. The Artificial Neural Network [2] categorized the image of the potato leaf as sustainable or inconvenient. The results show that backpropogation could effectively detect blackspots Either

FIG. 2.1. *Sample Tomato leaf diseases in real field conditions*

explore the disease and detect the disease with a consistency of 92%. Four combinations of neural networks have been used to differentiate between wheat stripe rust and wheat leaf rust and grape downy mildew based on extraction techniques. Results revealed that plant disease identity and diagnosis could be achieved efficiently using image processing-based Neural networks. In contrast [19] proposes image analysis methodology for the detection of tomato scab disease. The sick images are collected from various vegetable fields and stored for improvement. In order to gain target regions for disease spots, the image segmentation is performed. Ultimately a specified region assessment disease spots is focused on a gray image processing approach

2.1. Materials and methods. Deep computer vision and image identification learning is currently progressing. The Deep-CNN standard comprises of a softmax or input and output layer, a categorization layer and hidden multiple layer. In general, CNN's hidden layers consist of convolution layers, pool layers, fc layers, and sometimes softmax layers. Lenet-5 architecture follows most CNN implementations. A number of CNN architectures were designed, by contrast [3]. During this work, A study comparing of the current neural network of convolution and its tuning to define and identify tomato plant disease using PlantVillage images is carried out. PlantsVillage contains 14528 images with open and free dataset, with 10 diseases for one crop plant. VGG 16, Xception V4, ResNet50, Alexnet, Lenet are the architectures evaluated. Quick and accurate Desired models for the detection of plant disease are desired so that specific primitive measures can be used soon in the categorization of leaf disease [22].

2.2. BenchMark Dataset. Image processing modules have been evaluated and practiced on sample tomato leaf images to classify and identify image disease that the CNN model had not seen before PlantVillage's data set [3] was used for this study openly and freely. Plant dataset has 14528 images for one crop plant with 10 diseases. For the first time, the images for the VGG network, ResNet and Alexnet architectures were resized to 224 * 224. The images are expanded to 299 * 299 pixels on both sides for both the Xception V4 architecture. Data standardization is performed by splitting all pixel values. Increase the aim or category variable to be used for the studied models very first, two parts of the data are available. First, the specific training data and then the highly classified data with a percentage ratio of 80 percent and 20 percent. The new range of the basically split current ratio is inspired by Mohanty et al work [21]. That test set is used to examine and formulate new models. The data is again divided into two training data and the validation data remains 80 percent and 20 percent and to assess if the model is overfitted in our approach, respectively. The training class included 14528 samples, validation samples, and a sample test set of 3632.

2.3. Progressive art of deep learning.

VGG net model. Deep teaching design has comparable methods, VGG net is one of the CNN systems designed for the ILSVRC-2014 assignment by [17]. The system may have reached a top-5 error level of 7.5 percent on verification, leading to an rise in some study job. As seen in [15] the template is generally used for its modesty. Figure 2.1 shows that overall model loss and accuracy with only 3 convolutionary layers on top of each other, the CNN system is increasingly consistency-sized. Max performs volume reduction (down sampling) for pooling. As seen in their work [16], our own initial softmax layer was overpassed And substituted

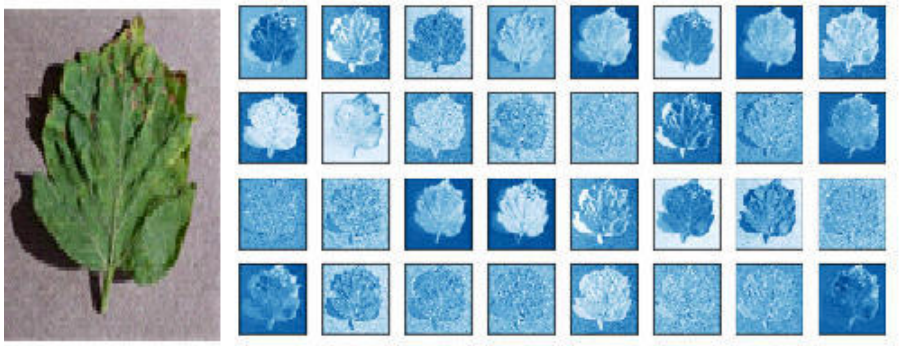


FIG. 2.2. *Visualization of convolution Filters*

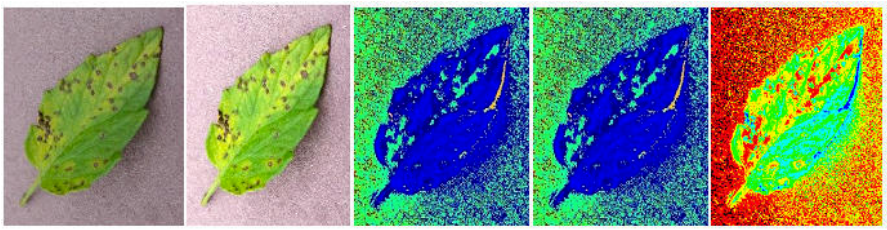


FIG. 2.3. *Action visualization learned weights by all layers*

by two fully embedded layers each of which has 4096 nodes and a classifier softmax. Our group's number 38 marks. A pre-trained ImageNet weight system was also used. Furthermore, on the sample set, the CNN system evaluated cross-entropy loss and accuracy.

ResNet. Very prominent in CNN [9]. ResNet design launched the ResNet system, the basis for the 2015 ILSVRC and 2015 classification contest.The suggested system of job took second position in the ImageNet classification with only a failure rate of 3.57 percent. The inability to understand different differential components for learning identification and degradation issues in different fields. Figure 2.2 shows that overall loss and accuracy of the model. This is an architecture that was incorporated as a network-in-network (NIN) into many full residual units. Such excess units are a set of basic components which are used to build networks.The result is a collection of basic residual unit types. ResNet architecture has resulted in a collection of basic residual unit types [5]. Convolution, pooling consists of residual units. A ResNet system has been created with 50, 101 and 50,101. Finally, a tailored crop illness detection layer of CNN was developed.

Xception V4. The later participation of GoogleNet architecture referring to the version amount was linked to Xception with few layers [8]. Stated architecture of Inception V3 also offers improvements to only the Inception module to increase the classification accuracy of ImageNet. In order to give primary importance to Inception V4 [16] strengthened the design.This architecture combines architectural design of activation with residual connections. Their objective is to combine training in network start-ups. The beginning-up module consists of a pooled layer stacked with convolution layers. The convolutions are 1 range, 3 range and 5 range in different sizes. The use of a bottle neck layer that is a bottle neck layer is another essential feature of the starting module. The bottleneck layer enables to reduce the computational demands that allow the required output to be produced. Within the unit, pooling layer is used in terms of size reduction [10]. The fusion of such parts as shown in requires a concatenation filter. Figure 2.3 shows that overall loss and accuracy of model. Xception v4 removes the concatenation stage with existing associations of the Inception architecture filter [13]. Finetuning of Inception V4 was carried out using ImageNet pre-trained weights. Furthermore, Avg pooling layer (88), dropout and softmax were used to transpose and describe a fresh template on the surface.

TABLE 2.1
Hyperparameters Settings

S.No	Hyperparameter	Settings
FD2	10^{-10}	26
FD4	10^{-12}	30
FD6	10^{-12}	30

TABLE 3.1
Hyperparameters Settings

S.No	Hyperparameter	Settings
1	Number of cnn layers	5
2	Number of neurons	500,100,7
3	Number of echos	30
4	Batch size	64
5	Activation functions	Relu
6	Optimizers	SGD
7	Learning rates	0.1
8	Momentum	0.9
9	Number of folds in cross-validation	10
10	Weight decay	0.004

2.4. Fine-tuning of models. Fine-tuning is an intrinsic notion of teaching to transfer the process of learning. Transfer teaching is an ML method, suggesting that acquiring expertise during practice is used in one type of issue to training in another associated assignment or field . The first few levels are taught in profound teaching to define the characteristics of the task. Most teaching involves fine-tuned teaching tests, Because the classes are much more faster than scratch [15]. Compared to scratch-trained designs they are also much more accurate. The CNN models have been carefully adjusted to verify and classify 10 crop disease types with previously trained designs to improve the leaf dataset in the training.The leafdataset provides about 1.2 million images and 1000 class entries.On the other hand, the Plantdataset contains 54,306 images and 38 classes. The Plantdataset is inadequate to train deep networks, although using the pre-trained weights of ImageNet. Fine tuning was achieved on both the plant dataset without any increase in information on CNN Inception v4, VGG16, ResNet. Fine tuning was achieved on the plant dataset without any increase in data from CNN Inception v4, VGG16, ResNet. The models were also developed and equipped with ImageNet pre-trained weights. By comparison, a new softmax layer on top layer truncated in the top layer [20]. Using the optimization SGD algorithm and the original learning frequency was 0.001, the system was also finely tuned.

2.5. Batch Normalization. Normalization is a method of deep learning that allows the inner change covariates to decrease layer problems [12]. When the yield of one layer is the source of the next layers during Deep Neural Networks practice. As the parameters of past layers shift during the learning phase of the CNN network, the dispersion of information information to layers differs dramatically over time. By having low learning prices, this method decreases instruction and makes it much easier to train those designs with regression swamping.When normalizing the CNN-Batch enables minimize the difficulties presented by the inner change of covariates. Changing the mean or width of input through one minibatch normalizes each layer's output, enabling higher learning prices to be used. All CNN tests are related to ReLU's batch normalization and activation feature.

3. DNN Settings. The DNN parameter settings consist of a number of all series of specific elements, that are presented in various architectures. DNN architectures shows pretrained architecture of a CNN, with its main elements, such as the input layer, convolutional layer, pooling layer, batch normalization and a process of flattening, where the information is entered into a set of dense layers, representing the result obtained in the output layer.

To continue a decent comparison between the experiments, an sample workflow attempt to standardize the hyper-parameters across the experiments was also made, using the following hyper-parameters, which are described in Table 3.1. DNN has unique significance and advanced in many research areas. Stochastic Gradient

TABLE 3.2
Machine Specifications

Hardware/software	Characteristics
Memory	64 Gb
Processor	Intel Core i7-7700
Graphics	GeForce GTX 1070 X 8 Gb
Operating system	ubuntu, 64 bits

Descent (SGD) has popular optimization algorithm and has been used in various architectures , and has proved to be an selective system between accuracy and efficiency [2]. The SGD is simple and effective, and it requires a tuning of the model hyper-parameters, particularly the initial learning rate, which is used in the optimization since it determines that the how fast the weights are adjusted in order to get a local or global to reduce the min loss function. The momentum helps to accelerate SGD in the suitable direction and reduces the overfitting [1]. In addition, the regularization is a very important technique to prevent the overfitting. The most common type of regularization is L2 Regularization, where the combination with SGD results in weight decay, in which each update the weights are scaled by a factor lightly smaller than one [18]. Each experiment runs a total of 30 epochs, where each epoch is the number of the training iteration. The choice was made due to the results of Mohanty et al work proposa because of its consistently converging after the first step down in the learning rate. Finally, all the CNNs are trained with the batch size of 32.Training these pretrained CNN architectures is extremely computational cost is very high intensive. Therefore, all the experiments are carried out on a workstation, presenting the details summarized in Table 3.2 [11]. The training process was conducted by Tensor flow using python with Deep Learning (DL) which provides a framework to design and implement CNNs, where applications and graphics help to visualize network activations and monitor the progress of network training. Meanwhile, the statistical analysis of each architecture was carried out with the Anaconda navigator–Spyder 3.2.

4. Results.

4.1. Experiment setup. The benchmark system used in our analysis is a GPU TiTan K40c workstation. The library of OpenCV, Keras, Theano, and CuDNN is used for implementation.

4.2. Training. From each DL- experiment, the model assessment uses accuracy metric and categorical cross-entropy loss (loss). The output is graphically designed with consistency and loss for each model. An overall experiment is computed using the test dataset to measure the loss score and accuracy and is used to determine the model performance. Table 4.1 introduces the measured results. For a total of 10 epochs and 30 epochs, each experiment runs. Where the epoch is the amount of iterations of training. The 10 and 30 iterations learning real choice was made to check which suitable model could mix with few iterations and which one is suffering from the problem of moral degradation. All popular deep learning networks have normalized the hyper-parameters. Stochastic Gradient Descent (SGD) is being used to train all network models, SGD runs better and diverges easily. They trained the networks with the batch size of 16 due to GPU memory limitations. Its learning rate for all networks was set at 0.001. They used 1e-6 weight from Decay and 0.9 momentum from Nesterov. Both experiments use batch standardized technique and ReLU activation function [13]. There is no data increase for all networks.

4.3. Results of the experiments. This experimental research has conducted an evaluation of the suitability of the distributed deep cnn to identify crop disease using pictures. Our focus was on the refining layers VGG 16, Inception V4, ResNet with 50, 101 and 152. Fine tuning and training of deep learning architectures as described in Section 2.1. Figures 4.1-4.8 explain the experiment's results. After good tuning, each document shows the precision and entropy loss of each DL design, all designs with 10 epochs except for VGG 16 were more than 90 percent accurate. By comparison, accuracy outcomes were achieved even after the 30th training cycle with systemically decreased log-loss. ResNet and designs continuously conduct better than VGG 16 and Inception V4. In addition, as viewed in The Deeper Models, they combined readily with stronger sample results as described in Table 4.1. Perform properly with fewer ResNet 50 and ResNet 101. On the other side, as shown in Figs. 4.1-4.8, ResNet 152 works badly with less iterations 4 and 5. However, with an enhanced amount of

TABLE 4.1
Results of Training and Testing and Execution time of the Proposed DCNN

CNN	Layers	Parameters	Training Loss	Validation Loss	time/sec
Alexnet	16	28.2M	0.0056	0.24	3Sec
Lenet	50	98.6M	0.009	0.56	2sec
Resnet	101	14.6M	0.0011	0.32	2.5sec
VGG 16	152	28.5M	0.99	0.0156	1.2sec
Proposed work	121	48.5M	0.17	0.24	2.9sec

TABLE 4.2
Hyperparameters Settings

Accuracy	Our proposed work	Alexnet	Lenet	Resnet	VGG16
%	99.45	90.1	88.3	98.40	90.1

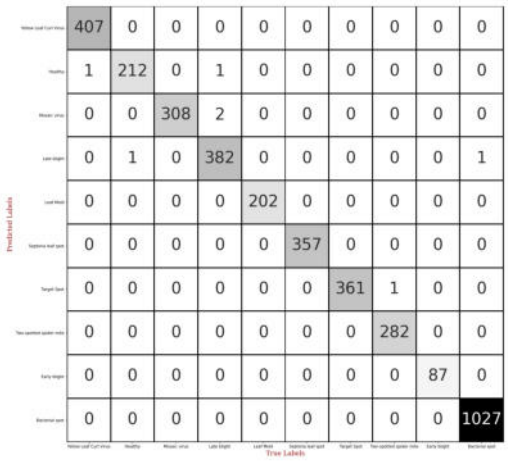


FIG. 4.1. *Results of the Proposed Confusion matrix*

TABLE 4.3
Classification accuracy of proposed CNN

Measures	Yellow	Healthy	Mosaic	Late Blight	Leaf mold	Septoria	Target
Accuracy	67	99	73	70	82	71	73
Precision	39	21	19	20	21	20	19
Sensitivity	98	90	99	85	79	89	78
Specificity	75	39	82	72	62	79	47

iterations as shown in Fig. 4.7, ResNet 152 continues to raise its precision and reduces its log-loss. Overall, Xception 121 performed well with the highest accuracy and low loss while VGG 16 performed badly with the lowest accuracy.

4.4. Discussions. The overall objective of this proposed work is to find the leaf disease in the images and to create an automatic classification of tomato leaf disease classification. Fast, accurate and detection of this disease can find out the diagnosis of disease. In this proposed Deep CNN, with TensorFlow and Keras model.Deep CNN we have analyzed from the Plant kaggle dataset it has 14528 images in that 1337 images are used validation and 3632 testing for one crop plant with 10 diseases. For the first time, the images for the VGG network, ResNet and Alexnet, Lenet, Xception architectures were resized to 224 * 224. The images

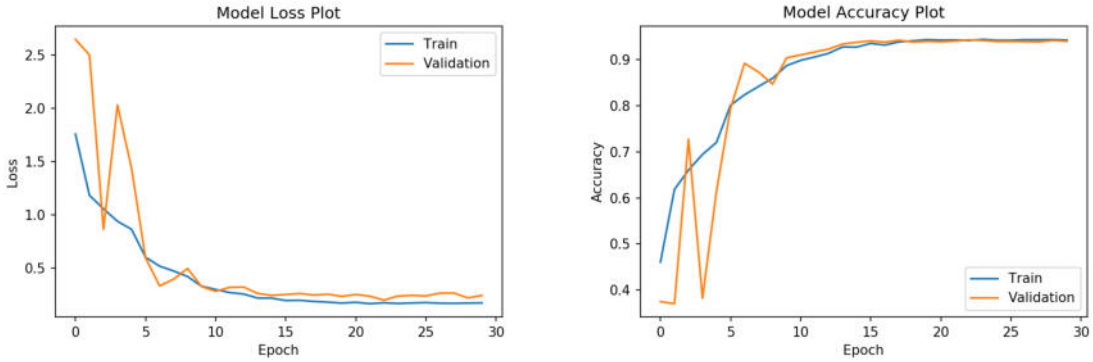


FIG. 4.2. *Results of ALEXNET Model: left Loss, right Accuracy*

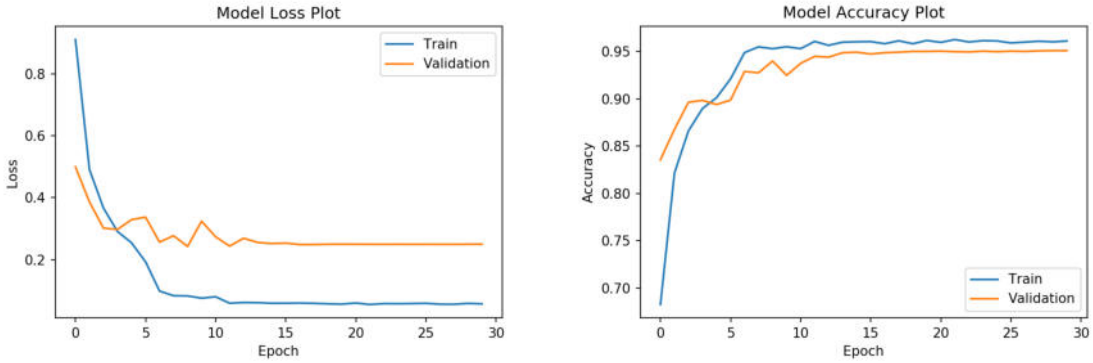


FIG. 4.3. *Results of LENET Model: left Loss, right Accuracy*

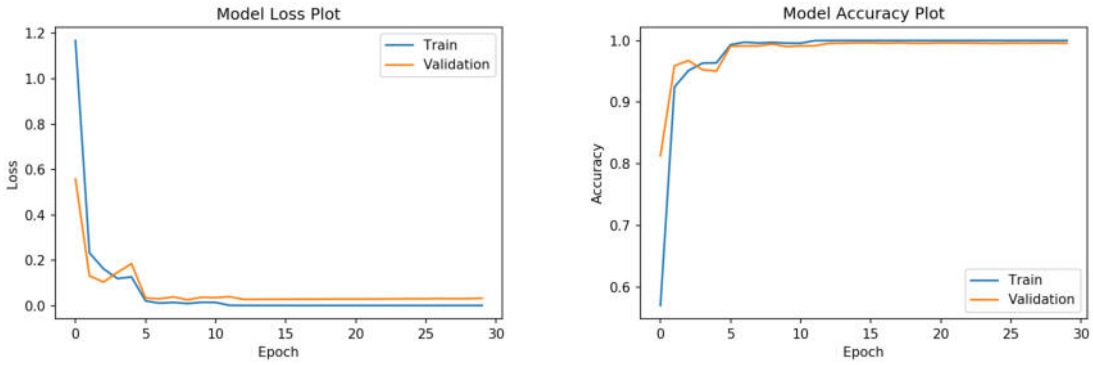


FIG. 4.4. *Results of VGG16 Model: left Loss, right Accuracy*

are enlarged to 299 * 299 pixels on both sides for both the Xception V4 architecture. Data standardization is performed by splitting all pixel values. Increase the aim or category variable to be used for the studied models very first, two parts of the data are available. First, the specific training data and then the highly classified data with a percentage ratio of 80 percent and 20 percent. The experimental results proved that the proposed method can reach a very high accuracy of 99.45% in the validation set and 95.03% in the test set with epochs equal to 30. The authors developed a classification of images from the kaggle plant village dataset, they used

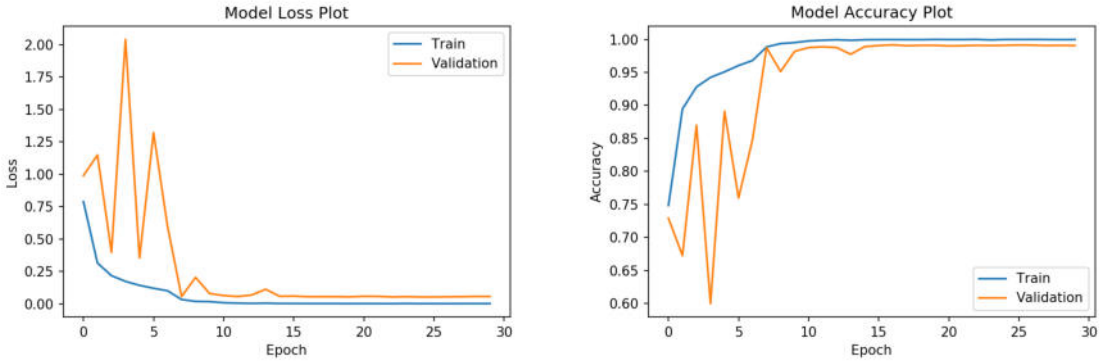


FIG. 4.5. *Results of Resnet Model: left Loss, right Accuracy*

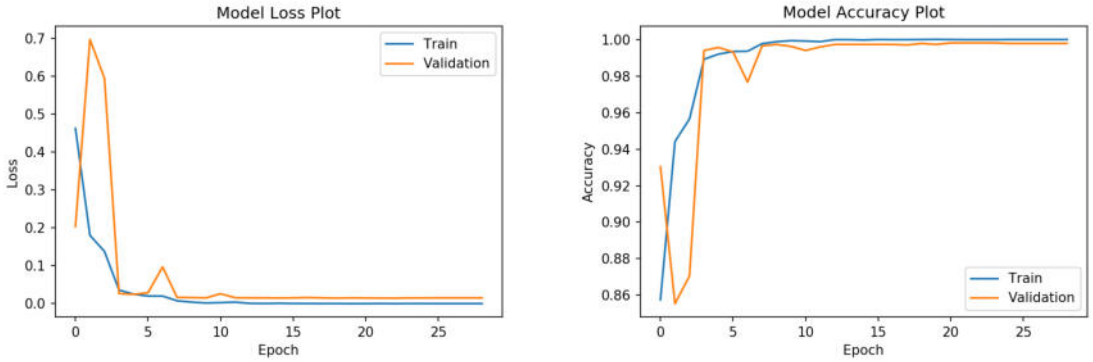


FIG. 4.6. *Results of Xception Model: left Loss, right Accuracy*

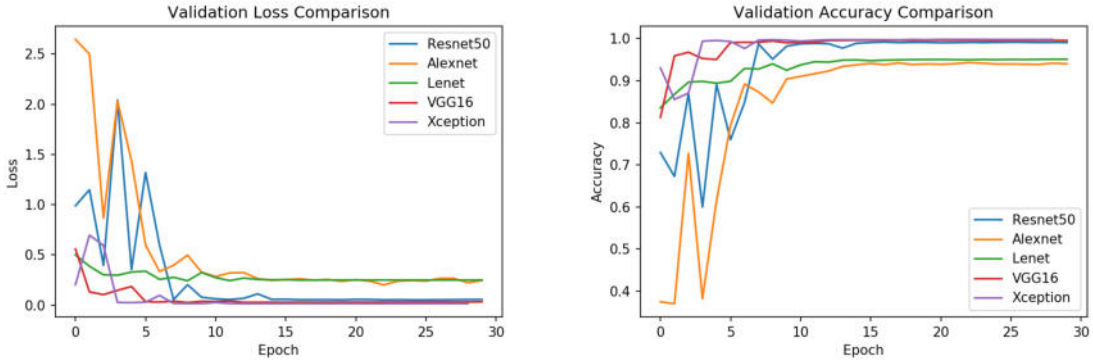


FIG. 4.7. *Left: Results of Loss Comparison; Right: Results of Accuracy Comparison*

convolution, pooling and full connection layers in the model created with the VGGNET architecture. In their research, the test phase of the educated model, class validation was obtained at 95.62%. In our work we have trained a CNN based on the ResNet50 architecture to classify leaf images of we have obtained a accuracy of 91% on the validation dataset. Therefore we have analyse differnt architectures of alexnet, lenet, VGGNET, resnet50 all of them Xception has attained a maximum accuracy of 99.45% of all rest of architectures.

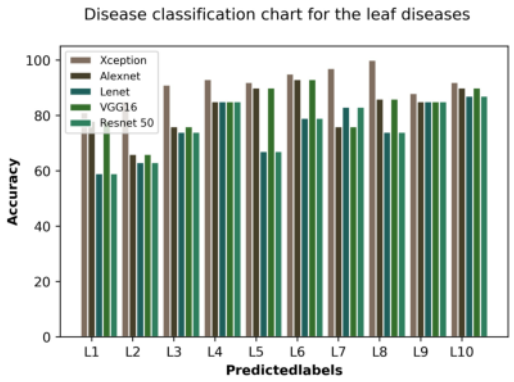


FIG. 4.8. *Disease Classification Plot for all Diseases*

TABLE 4.4
Parameters setting for proposed DCNN

Class	Images for training	Images for validation	Images for Testing
YLC Virus	1787	128	408
Healthy	787	120	213
Mosaic virus	1283	131	308
Late blight	1524	152	385
Leaf Mold	750	140	202
Septoria leaf spot	1414	185	357
Target Spot	1315	106	361
Two-spottedspider mite	1121	129	283
Early blight	286	56	87
Bacterial spot	4329	190	1028
Total	14528	1337	3632

5. Conclusion. From the above task, the refinement and analysis of the progressive deep convolutional neural network is performed for the identification of image-based plant disease. The architectures analyzed are VGG 16, Xception V4, ResNet 50, Alexnet, Lenet layers.From the analysis, Xception continues to yield consistent rises Throughout Precision with increasing number of epochs, with no signs of performance depletion and overfitting. In addition, Xception needs fewer numerical parameters and reasonable computing time to obtain the best outcomes in classified events. Xception's test accuracy for the 30th epoch is 99.45% percent, defeating the entire of all the architectures. Therefore, Xception v4 is a good model for image-based disease detection of plants. Even though the architecture's performance is good, increased research is needed to improve computational time.

REFERENCES

[1] S. CHETLUR, C. WOOLLEY, P. VANDERMERSCH, J. COHEN, J. TRAN, B. CATANZARO, AND E. SHELHAMER, *cudnn: Efficient primitives for deep learning*, arXiv preprint arXiv:1410.0759, (2014).
[2] L. DENG AND D. YU, *Deep learning: methods and applications*, Foundations and trends in signal processing, 7 (2014), pp. 197–387.
[3] K. HE, X. ZHANG, S. REN, AND J. SUN, *Deep residual learning for image recognition*, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.
[4] G. HUANG, Z. LIU, L. VAN DER MAATEN, AND K. Q. WEINBERGER, *Densely connected convolutional networks*, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 4700–4708.
[5] D. HUGHES, M. SALATHÉ, ET AL., *An open access repository of images on plant health to enable the development of mobile disease diagnostics*, arXiv preprint arXiv:1511.08060, (2015).
[6] S. IOFFE AND C. SZEGEDY, *Batch normalization: Accelerating deep network training by reducing internal covariate shift*, arXiv preprint arXiv:1502.03167, (2015).
[7] D. S. KERMANY, M. GOLDBAUM, W. CAI, C. C. VALENTIM, H. LIANG, S. L. BAXTER, A. MCKEOWN, G. YANG, X. WU,

F. Yan, et al., *Identifying medical diagnoses and treatable diseases by image-based deep learning*, Cell, 172 (2018), pp. 1122–1131.
[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, *Imagenet classification with deep convolutional neural networks*, Communications of the ACM, 60 (2017), pp. 84–90.
[9] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, *Gradient-based learning applied to document recognition*, Proceedings of the IEEE, 86 (1998), pp. 2278–2324.
[10] M. Dyrmann, H. Karstoft, and H.S. Midtiby, *Plant species classification using deep convolutional neural network*, Biosystems Engineering, 151 (2016), pp. 72–80,
[11] S. Prasanna Mohanty, D. Hughes, and M. Salathe, *Using deep learning for image-based plant disease detection*, arXiv, (2016), pp. arXiv–1604.
[12] A. S. Rajput, S. Shukla, and S. Thakur, *Soybean leaf diseases detection and classification using recent image processing techniques*.
[13] A. K. Reyes, J. C. Caicedo, and J. E. Camargo, *Fine-tuning deep convolutional networks for plant recognition.*, CLEF (Working Notes), 1391 (2015), pp. 467–475.
[14] I. Sa, Z. Ge, F. Dayoub, B. Upcroft, T. Perez, and C. McCool, *Deepfruits: A fruit detection system using deep neural networks*, Sensors, 16 (2016), p. 1222.
[15] D. Samanta, P. P. Chaudhury, and A. Ghosh, *Scab diseases detection of potato using image processing*, International Journal of Computer Trends and Technology, 3 (2012).
[16] S. Sladojevic, M. Arsenovic, A. Anderla, D. Culibrk, and D. Stefanovic, *Deep neural networks based recognition of plant diseases by leaf image classification*, Computational intelligence and neuroscience, 2016 (2016).
[17] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, *Inception-v4, inception-resnet and the impact of residual connections on learning*, arXiv preprint arXiv:1602.07261, (2016).
[18] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, *Rethinking the inception architecture for computer vision*, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 2818–2826.
[19] S. Prasanna Mohanty, D. Hughes, David and M. Salathe,, *Using Deep Learning for Image-Based Plant Disease Detection*, arXiv (2016), arXiv–1604.
[20] E. C. Too, L. Yujian, S. Njuki, and L. Yingchun, *A comparative study of fine-tuning deep learning models for plant disease identification*, Computers and Electronics in Agriculture, 161 (2019), pp. 272–279.
[21] H. Wang, G. Li, Z. Ma, and X. Li, *Application of neural networks to image recognition of plant diseases*, in 2012 International Conference on Systems and Informatics (ICSAI2012), IEEE, 2012, pp. 2159–2164.
[22] D. Yu, W. Xiong, J. Droppo, A. Stolcke, G. Ye, J. Li, and G. Zweig, *Deep convolutional neural networks with layer-wise context expansion and attention.*, in Interspeech, 2016, pp. 17–21.

Edited by: Rajkumar Rajasekaran
Received: Apr 4, 2020
Accepted: Dec 7, 2020

Scalable Computing: Practice and Experience, ISSN 1895-1767, http://www.scpe.org
© 2020 SCPE. Volume 21, Issues 4, pp. 637–648, DOI 10.12694:/scpe.v21i4.1787

A NEW CLUSTERING ROUTING PROTOCOL FOR HOMOGENEOUS WIRELESS SENSOR NETWORKS POWERED BY RENEWABLE ENERGY SOURCES*

CHIRINE BASSIL† HUSSEIN EL GHOR‡ JAWAD KHALIFE§ AND NIZAR HAMADEH¶

Abstract. The technology of wireless sensor networks (WSNs) is in constant development and it made great progress in many applications. One of the most popular problems in WSNs is the limited energy storage power at every sensor node. This paper aims to propose and develop a new distributed clustering algorithm for energy harvesting wireless sensor networks denoted by DEH-WSN (Energy Harvesting for Distributed Clustering Wireless Sensor Networks Protocol) that relies on matching between clustering and energy harvesting in a distributed topology. DEH-WSN uses initial and residual energy capacity of the nodes to choose cluster heads. Simulation results prove that the proposed method increases network lifetime and the effective throughput.

Key words: WSN, Routing, Clustering, DEH-WSN, Network Lifetime, Throughput.

AMS subject classifications. 68M10

1. Introduction.

Wireless sensor networks (WSNs) are one of the fastest-growing technologies; it is enabled by the rapid advances in micro-electro-mechanical-system (MEMS), computer networks, and wireless communication technology. Sensors are developed by the integration of sensing and wireless communication. They consist of low-power small sensor nodes that are able to sense, process and interact over unreliable short-range radio connections. These sensor nodes are deployed in the target area to observe physical or environmental conditions like temperature, sound, vibration, or pressure. The essential subsystems of sensor nodes are used to acquire data for local processing, and for sharing information by wireless communication.

WSNs have attracted enormous attention in diverse areas such as disaster warning systems, environment monitoring, safety, intruder detection, and others [1, 2, 3]. WSNs are characterized by their potential applications in various fields [4], especially for disaster warning systems, ecological monitoring, Healthcare, intrusion detection, fire detection, to name but few.

Sensor nodes can fail due to energy starvation since they rely on batteries with limited energy capacity. Hence, energy harvesting system that harvests energy from renewable energy sources (solar, wind, vibrations, etc) is deemed as a promising solution to overcome the shortage of limited battery capacity by transforming energy-efficient solutions to a distributed Energy Harvesting approach [5]. The energy harvested is used in multiple conditions to implement the power systems such as no energy storage, energy storage without battery, and rechargeable battery [6]. However, since the energy consumption of sensor nodes is much higher than the charging rate, these nodes need to repose for some time to recharge, but this drives a modification in the network's topology. Therefore, new clustering techniques and cluster head (CH) selection are proposed to to maximize the network lifetime.

In this paper, we overcome the energy starvation in the sensors energy source by following a distributed energy Harvesting protocol that relies on a battery that harvests energy based on a renewable energy source. Moreover, the node that has a fewer delay time compared to its neighbors has higher possibilities to be a Cluster Head and it's election is based on their energy level and the energy harvested. After creating a cluster

*This work was supported by the Laboratory of Embedded and Networked Systems, Lebanese University.

†Laboratory of Embedded and Networked Systems, Lebanese University, Faculty of Technology, B.P. 813, Saida, Lebanon.

‡Laboratory of Embedded and Networked Systems, Lebanese University, Faculty of Technology, B.P. 813, Saida, Lebanon. (hussein.elghor@ul.eu.lb).

§Laboratory of Embedded and Networked Systems, Lebanese University, Beirut, Lebanon. (jkhalife@ul.eu.lb).

¶Laboratory of Embedded and Networked Systems, Lebanese University, Faculty of Technology, B.P. 813, Saida, Lebanon (nizar.hemadeh@ul.eu.lb).

and selecting a cluster head, each node from its cluster sends packets to the Base Station. To this end, we investigate the aggregation protocol for EH-WSN algorithms, which must run as dynamically as possible to prevent energy harvesting sensors in the block mode from joining the network since they increase the workload and lower the productivity. This protocol aims are to resolve the energy constraints to extend the lifetime of the WSN, to raise the transmission rate and to decrease the network workload.

The rest of this paper is summarized as follows: Section 2 present some related work to enhance the energy consumption by using clustering-based protocols. An overview of the distributed energy efficient protocol is described in section 3. The process of our protocol and its steps including the simulation and an execution are stated in Section 4, and Section 5 presents a summary of the study, including impact, limitations and future work.

2. Related Work. In the past decade, several researchers proposed a considerable number of algorithms that use clustering-based protocols to improve energy consumption. In this section, we introduce and describe the most relevant clustering protocols.

LEACH [7], the Low-Energy Adaptive Clustering Hierarchy is dynamic, where the clusters are randomly distributed. That means that every node can perform a Cluster Head (CH) function with a different probability. The position of cluster heads rotates between the various nodes to limit the breakdown due to energy starvation in any of the nodes. Whilst nodes are connected to the CHs with the smallest energy demanded to reach it. And in turn, the current cluster heads send the data in order through time slots allocation. Another widely used algorithm, Energy-efficient distributed clustering algorithm HEED was proposed in [8], the cluster heads are chosen according to the remaining energy combination and their connection costs.

EECS [9], EEUC [10], EDUC [11] and EADUC [12] are coverage-aware and energy-efficient algorithms. Consequently, they focus on effective size of clusters factor, which is the distance from cluster heads to the base station. Such algorithms improve the distribution of energy throughout the network and extend the network lifetime.

A Centralized Balance Clustering (CBC) [13] protocol is implemented in 3 steps. Initially, it computes the number of clusters due to the network conditions. Then, CH is selected for each block. In the final step, it is scheduled to send data while still avoiding any collision.

Later, a Hybrid Unequal Clustering with Layering Protocol (HUCL) was proposed in [14] that extends the network lifetime. This protocol is used to solve the problem of clustering overhead in case of dynamic clustering. HUCL first presents a simple compression algorithm to reduce the excess in data transmission and then proposes a mixture of static and dynamic clustering that greatly reduces the clustering overhead when compared to other dynamic aggregation techniques.

Amgoth et al. in [15] proposed an algorithm named ERA (Energy-Aware Routing Algorithm) that forms clusters according to the level of energy in the CHs. ERA organizes Clusters at different levels to be able to build the virtual backbone of the routing data.

The study in [16] suggested Energy and coverage-aware distributed clustering ECDC method that introduces coverage importance measures for the region and the whole coverage of points. ECDC increases the lifetime of the network by affecting these measures in calculating the waiting time and finding forwarding data the way to the sink.

Energy-Harvesting Stable Election Protocol (EH-SEP) [17] is based on the SEP algorithm introduced in [18]. EH-SEP is energy harvested clustering protocol. The probability of newer nodes is greater than that of older nodes, and hence the remaining energy is dedicated to the cluster heads.

CRBS algorithm (Clustering Routing Algorithm Based on Solar Energy Harvesting) [19] is another algorithm that uses both of the soft and hard thresholds to connect nodes to the network in the next round in case some nodes die. The main advantage of this protocol is that it increases the number of alive nodes and enhances the stability.

Later in [20], authors propose a novel NEEC protocol (Novel Energy Efficient Clustering), which is implemented in a centralized and distributed manner. NEEC uses hybridization such as static and dynamic clustering. In NEEC, the packets repaired in the Base Station are enhanced by supporting the consumed energy in the WSN.

S-LEACH presented in [21], is one of the most relevant studies conducted in WSN, which is based on energy

harvesting sensor nodes from solar energy and they relies on a battery as a backup power source. In S-LEACH, the BS chooses the CHs and then these CHs select a new one.

In [22], authors improved the previous methods by proposing a hybrid unequal energy-efficient clustering that prolongs the network lifetime. A novel clustering strategy is used based on arrangement of nodes in a network. Under this strategy, we can determine whether the information of the neighbors' nodes should be used or not. Hence, overhead is considerably reduced.

A clustering algorithm for heterogeneous WSNs based on a solar energy supply was presented in [23]. The CH is selected based on the self-replenishment state and the remaining energy of the nodes. Authors demonstrated that the proposed algorithm can effectively increase the network lifetime in addition to improving the network efficiency and stability while still balancing the energy consumption compared with previous algorithms.

Later in [24], authors presented a new hybrid and unequal multi-hop clustering protocol that aims to prolong the network lifetime. CHs are selected only by comparing the status of each node to its neighboring nodes. For each node, authors studied the following factors: the residual energy, the distance from this node to the base station, the number of neighboring nodes, and the placement of the layer node.

In [25], authors proposed an "energy-efficient clustering routing protocol" based on the deployment of each high-QoS node with an inter-cluster routing mechanism. For this basis, authors defined several formulas that is based on twofold coverage for information integrity, validity, and redundancy. Authors demonstrated that the deployment strategy of the proposed protocol has higher information integrity and validity, as well as lower redundancy.

For a complete survey on the clustering routing techniques in WSN with energy-efficient considerations, you can refer to the survey of energy-aware cluster head selection techniques in WSN in [26].

for reducing energy consumption

3. Distributed Protocol Overview. Li et al. in [27] proposed a distributed energy-efficient clustering algorithm. The main idea behind this Clustering Algorithm is to reduce energy consumption then increase the scalability and lifetime of the network. Following the thoughts of LEACH [27], [28], this protocol is an energy-efficient protocol for heterogeneous WSNs and it becomes homogenous after many rounds.

The field is divided into different clusters. Each cluster has a CH and some sensor nodes. CH receives from a cluster the information from the sensor nodes, and then sends it to the BS. To consider the node as Cluster Head CH, a probability function is defined to compute the residual energy and the average energy of networks. This probability function computes the ratio between the remaining energy in each node and the network's average energy. The nodes with high computation have a better chance to be elected as a CH.

There are two phases in the distributed protocol:

- Setup phase: the clusters are created and the cluster heads (CHs) are selected.
- Steady phase: the data from non-cluster heads are transmitted to the sink.

The proposed algorithm is divided into four steps:

1. Initialization Phase: the possibility of being CH changes according to the capabilities of the nodes. The desired number of CH is selected according to their location. When the range of CH is defined, CH sends a membership request message to all the nodes in its range, then request to reply with their current energy status. All nodes with high residual energy and processing power will be identified and they are made to sleep, they become the backup nodes. In case the nodes are not in the range of CH, they join the cluster by sending a message to the nearest cluster member.
2. Steady State Phase: the cluster members send the sensed data to the CH in the allotted time using TDMA schedule, and the non-cluster members to the cluster head through the intermediate cluster member.
3. Final Phase: CH will aggregate the data from all the nodes in its cluster, and then it will transmit this data to the base station.
4. Cluster Reconfiguration Phase: CH will activate the backup node if the CH residual energy reaches to the threshold value, then it will make the backup node as new CH, and transmit the new CH information to all other nodes and CH, and the old CH will become the general node in the last phase.

TABLE 4.1
Description of the parameters used in equation 4.1.

Parameters	Description
$E_{rem}(i,r)$	Remaining energy in node i during round r
$E_{eh}(i,r-1)$	Energy harvested by node i during previous round $r-1$
$E_{max}(i)$	highest energy storage capacity in node i

TABLE 4.2
Description of the parameters used in equation 4.1.

Parameters	Description
μ_i	Energy harvesting rate of node i during the round $r-1$
$P_{(h,min)}(r-1)$	Probable minimum energy harvesting rate for all nodes during the round $r-1$
$P_{(h,max)}(r-1)$	Probable maximum energy harvesting rate for all nodes during the round $r-1$

4. The Proposed Algorithm. We presented and evaluated a new routing protocol that relies on matching between clustering and energy harvesting in a distributed manner, which aims to maximize network lifetime and to become unlimited, by using energy harvesting instead of energy efficient Clustering protocols. At First, we introduced the model network, and then we explain the proposed algorithm.

4.1. Network Model. Energy scavenging in WSN is provided with Energy Harvesting nodes and a base station with unlimited network supplies. The data is taken by the sensor nodes and is sent to the base station. Some assumptions are made about the sensor nodes and the network model:

- N energy harvesting sensor nodes are distributed randomly in a $(N \times N)$ field.
- Nodes are aware of their location.
- The transmission power is detected according to the distance.
- CH can reach the base station in one hop or multiple hops.

4.2. Energy Harvesting Model. Energy harvesting (or energy scavenging) sensor nodes harvest the energy from the environment. The harvested energy uses a storage capacity defined by$E^{eh}_{i,r-1}$. However, the batteries have a limited amount of energy and they require a periodic charging replacement. In addition, renewable energy is not constant and it changes over time.

Based on the exponentially weighted moving average (EWMA) we used a forecast model for modeling energy harvested from sunlight [29, 30, 31].

Equation 4.1 calculates the amount of energy model for an energy harvesting node i:

$$E_{rem}(i,r) = min(E_{max}(i), E_{rem}(i,r-1) + E_{eh}(i,r-1)) \tag{4.1}$$

Table 4.1 shows a description of the parameters used in equation 4.1.

Equation 4.2 calculates the amount of energy harvesting:

$$E_{eh}(i,r-1) = \mu_i \Delta t \tag{4.2}$$

Equation 4.3 calculates the energy harvesting rate:

$$\mu_i = rand(P_{(h,min)}(r-1), P_{(h,max)}(r-1)) \tag{4.3}$$

Table 4.2 shows a description of the parameters used in equations 4.2 and 4.3.

The Energy harvested in any node has a low and high threshold. Each node will be automatically blocked if the node's energy is below the low threshold and will not participate in the current round, or has the ability to

TABLE 4.3
Description of the parameters used in equation 4.4.

Parameters	Description
D	The delay in node i
ϵ	A very small number, if the parameter is zero, ϵ does not affect the equation
$\|NL(i)\|$	The neighbors' number of node i
$\alpha(i)$	The number of times that node i is elected as CH.
T_2	The time needed for the next phase
V_r	A number ranging between 0.1 and 0.2

transfer data when the power state has not reached the required level. That means that the node will continue to recharge the battery. When the energy capacity in the battery of the blocked node becomes above the low threshold, the node will switch to active mode. In other words, the node will be able join the network at the next setup stage and will begin sending and receiving data in the next round.

4.3. DEH-WSN Protocol (Energy Harvesting for Distributed Clustering Wireless Sensor Networks). In this section, we introduce a new protocol, named DEH-WSN (Energy Harvesting for Distributed Clustering wireless sensor networks). We consider that the sensor nodes know the information about their location depending on the frequency power.

In DEH-WSN, the selection of the cluster head is based on the following factors: (i) constant probability value, (ii) initial energy level, (iii) processing power and (iv) the amount of harvested energy. Moreover, some nodes are selected as cluster head based on their location.

This protocol is performed in a distributed manner and distributed into two rounds, where each one implemented in two phases: Setup State and Steady data transmission State. In the setup phase, the cluster heads are selected and the normal clusters are formed according to the algorithm discussed later, then a matching between energy harvested and cluster heads are performed for moving to the next step. For the data transmission phase, the network schedule is divided into multiple rounds. In each one, cluster heads receive sensed data from the cluster member's nodes and collect data before transferring them to the base station.

4.3.1. Steady Phase:. The implementation of this phase requires four steps presented as follows:

Step 1: Calculation of delay time:. The selection of cluster heads depends on the energy level and capacity of the harvested energy. According to equation 4.4, the nodes calculate delay time, which help to choose the appropriate cluster head.

$$D(i,r) = \frac{E_{max}(i)}{E_{rem}(i,r)} \times X \times Y \times d_{i,BS} \times Z \times V_r \times T_2 \tag{4.4}$$

where:

$$X = \frac{1}{max(E_{eh}(i,r),\epsilon)}, \qquad Y = \frac{1}{max(|NL(i)|,\epsilon)} \qquad Z = max(\alpha(i),\epsilon)$$

The parameters of equation 4.4 are described in Table 4.3.

Step 2: Selection of a cluster head. Selection of cluster heads obeys the following rules:

- All nodes must wait to finish the delay time.
- The node that has smaller delay time has more possibility to be selected as a CH.
- If the node doesn't receive a message from the nearest neighbors, it declares itself as a CH.
- If the node receives CH massage, it has no possibility to be cluster head at all.
- In case any two nodes have similar delay times, the selected node should have a smaller ID.

Table 4.4
Description of the parameters used in above equatios

Parameters	Description
$E_{TX}^{CH_i,CH_j}(i,l,d)$	The energy needed to send data from CH_i to CH_j.
$E_{TX}^{CH_i,next\_hop_j}(j,l,d)$	The energy needed to send data from CH_j to the next phase
$E_{RX}(j,1)$	The energy needed to receive data in node j
$M(j)$	The number of member nodes in CH_j
$R(j)$	The number of CHs, where node j acts as relay node and receive the CHs data

Step 3: Cluster formation. For cluster formation, we followed the following rules:

- A selected CH node forwards a message including the energy level to the non-cluster-head.
- The non-cluster-head follows the cluster head among the lowest energy needed to transmit data to CH.
- A cluster head schedule nodes according to Time Division Multiple Access (TDMA) [32]. The further nodes must forward data as soon as possible and the further CH calculates the average energy.
- CH calculates the distance thresholds, where $d_{(c)}$ is the closest, and $d_{(f)}$ is the furthest nodes. The nodes that have a distance lower than $d_{(c)}$ lay in the first layer and the nodes that have a distance more than $d_{(f)}$ are laid in the second layer.

Step 4: Route Construction. Member nodes have the chance to turn to the sleep mode. Tthe first layer nodes plus nodes with distances smaller than d_0, from base station transmit the RR message (Route Request) into the network. The second layer nodes that receives the message must update their routing tables. After that, they will begin to transmit RR message to upper layers. The cluster heads found in the first layer must directly send data to the base stations. The cost to transfer data to BS is computed thanks to equation 4.5:

$$cost_i = E_{TX}^{CH_i,BS}(i,l,d) \tag{4.5}$$

In addition, the evaluation parameter to transfer data to a base station, or to the central Cluster-Head is computed as follows (equation 4.6):

$$cost_0 = \begin{cases} T_{c1} & if \;\; E_{rem}(i,r) \geq E_{TX}^{CH_i,CH_j}(i,l,d) \;\; and \;\; E_{rem}(j,r) \geq T_{c2} \\ Inf & Otherwise \end{cases} \tag{4.6}$$

where:

$$T_{c1} = E_{TX}^{CH_i,CH_j}(i,l,d) + E_{TX}^{CH_j,next\_hop_j}(j,l,d) + E_{RX}(j,l) \tag{4.7}$$

$$T_{c2} = E_{TX}^{CH_j,next\_hop_j}(j,l,d) + (E_{RX}(j,1) \times (M(j) + R(j) + 1)) \tag{4.8}$$

Table 4.4 shows a description of the parameters used in the above equations.

Hence, CH is chosen as the cheapest relay cost; thereafter CH_i sends a Route-Reply-message to the selected CH. Routing to the base station is made according to the route exposure phase without disruption during routing. Just if during intra-cluster routing a cluster head is not held in a layer.

Algorithm 1 states the pseudo code of the Setup phase of the proposed DEH-WSN algorithm.

4.4. Data Transmission Phase. Data transmission is performed thanks to the Carrier Sense Multiple Access (CSMA) method [30]. We use the Time Division Multiple Access (TDMA) where any cluster has only one node to send packets. First, nodes send all data to the cluster head. Then, Cluster head sends the packets to BS. By default, nodes that are in the first layer can send directly to BS.

Algorithm 2 shows the pseudo code of the Data Transmission Phase of the proposed DEH-WSN algorithm.

Algorithm 1 Setup phase of DEH-WSN

BEGIN
if $S[i].state$="CHP" **then** ▷ $S[i]$ is the sensor node and "CHP" is the Cluster Head P, i.e. the period that the network is operational
    exit
else$S[i].state = node$
while $CT < TimePh1$ **do** ▷ CT is the calculation time of a cluster and $TimePh1$ is the time of the first phase
    $V_r = rand(0.1, 0.2)$ ▷ V_r ranges between 0.1 & 0.2.
    Compute the Delay Time $D(i, r)$ by equation 4.4
$T = TimePh1 + D(i, r)$ ▷ T is the delay of a node
while $CT < TimePh2$ **do** ▷ $TimePh2$ is the Time of the second phase
    if $CT > T$ **then**
        $S[i].state$ = 'CH'
        Send Msg_Head
        Receive Msg_Head from CH
        Store in List_Head $CHL[]$ along with distance ▷ $CHL[]$ is defined as the List of Cluster Head
    else if List_Head is received from any neighbor **then**
        $S[i].state$ = 'CM' ▷ CM is defined as the Cluster Member
        Store '[j]' in List_Head $CHL[]$ along with distance
while $CT < TimePh3$ **do** ▷ $TimePh3$ is the Time of the third phase
    if $S[i].state$ = 'CM' **then**
        Choose the nearest $CHS[j]$ from $CHL[]$ list ▷ $CHS[]$ is defined as the List of Sensor Node
        $S[i].head = S[j]$
        End JC Msg to $S[j]$ ▷ JC: Join Cluster
    else if $S[i].state = CH$ **then**
        Receive JC Msg from CM
        Save in $CM[]$ List
        Compute the Avg Energy of Cluster & weak members
        Transmit TDMA to $CM[]$
        $S[i].state$ = 'CHP'
while $CT < TimePh4$ **do** ▷ $TimePh4$ is the Time of the fourth phase
    if $S[i].state$ = 'CHP' **then**
        while $CT < Tr$ **do** ▷ Tr is the Time Route Msg
            Wait and receive Route_Msg from CHs
            Broadcast Route_Msg
            Save CHs data in Relay_CH List[]
    Select the CH of the next hop from Relay_CH List[] by formulas 4.5 and 4.6
    while $CT < Trr$) **do** ▷ Tr is the Time Route Replay
        Wait and receive Route_Replay from CHs in upper Layer
        if receive Route_Replay **then**
            Store the information of the CHs in upper Layer
        broadcast Route_Replay to next hop in lower Layer
END

Algorithm 2 Data Transmission Phase of DEH-WSN

```
BEGIN
while CT < TimePh1 do
    if S[i].state =' CM' then
        while CT < transmition time from TDMA do
            In case the Sensor is relay for CM farther by TDMA
            Receive DP from CM and aggregation DP                ▷ DP is the Data Packet
        Transmit DP to BS or next hop from CM by TDMA
    else if [i].state = CHP then
        while CT < end of TDMA do
            Receive DP from CM and aggregation DP
        while CT < TimePh2 do
            if [i].state = CHP then
                while CT < base Layering transmission time do
                    In case Sensor is the relay for CH in the upper Layer
                    Receive DP from CH and aggregation DP
        Transmit DP to BS or next hop
END
```

TABLE 5.1
Scenarios used for the simulation.

Scenarios	Base Station	Nodes Number	Network Space
Scenario 1	(1500,500)	100	1km × 1 Km
Scenario 2	(500,500)	50	1km × 1 Km

5. Simulation Results. We assess the performance of the DEH-WSN protocol via simulations using MATLAB. We compared our proposed protocol (DEH-WSN) with respect to other NEEC [20] and HUCL [14].

Our study evaluates the stability in the network, the First Node Death (FND), the Half Node Death (HND), the number of alive nodes, the average energy during simulations and the throughput.

We have simulated our WSN in a sensing field of (1km × 1km). In these experiments, two scenarios have been evaluated, as shown in Table 5.1. Other parameters are shown in Table 5.2.

The clusters closer to BS should be smaller, the cluster should have more strength to route higher-level packets to the base stations. In our simulation, we assume that $P_{opt} = 0.1$ and $C = 0.4$, and we set $RL_{max} = 550$ m where $RL_{max} = 1.25 \times 550$ (second and third layers) and $RL_{max} = 1.75 \times 500$ (fourth layer).

In the first scenario, 100 sensors nodes are distributed randomly in the area, A sink node is at the location of coordinates ($x = 1500$, $y = 500$). In the second scenario, 50 sensor nodes are randomly distributed in the same simulation area with a sink node is at the location of coordinates ($x = 500$, $y = 500$). According to this data information, we presented a comparison between these two scenarios for the two experiments.

Based on LEACH protocol, both energy consumption per bit are used, the free space E_{fs} (d^n power loss) and the multipath fading E_{amp} (d^n power loss), they rely on the distance between the sender and receiver. Where we use the free space model with $n = 2$, if the distance is less than a threshold d_0. Otherwise, the multipath model is used where $n = 4$.

5.1. Experiment 1: Number of alive nodes per time. The first experiment is about the number of alive nodes per time for the three different protocols. Figures 5.1 illustrate the number of alive nodes during simulations, seeking to calculate the average number of alive nodes shown in Table 5.3, the results demonstrate that DEH-WSN has a great performance since it raises the number of alive nodes 25% against HUCL protocol.

TABLE 5.2
Simulation Parameters.

Parameters	Symbols	Value
Energy depletion of the node's electronics circuit to transmit or receive the signal.	E_{elec}	60 nJ/bit
Energy depletion of the booster to deliver at a shorter distance (Free Space)	E_{fs}	10 $nJ/bit/m^2$
Energy depletion of the booster to deliver at a longer distance (Transmit amplifier)	E_{amp}	0.0015 $pJ/bit/m^4$
Energy for data aggregation cost expended in Cluster-Head per signal	E_{DA}	5 $nJ/bit/signal$
First energy of normal nodes	E_o	0.5 J
Data packet size	-	5000 $byte$
Packet header size	-	25 $byte$
Control message size	-	50 $byte$
Lower threshold for energy harvesting	$E_{Th_{down}}$	0.1 J
Upper threshold for energy harvesting	$E_{Th_{up}}$	1 J
Probability rate of CH	P_{opt}	0.1

TABLE 5.3
Rate of the average number of alive nodes.

Protocol	Scenario 1	Scenario 2
DEH-WSN	80.6351	41.7261
NEEC [20]	80.6	40.9
HUCL [14]	60.5	33.5

5.2. Experiment 2: Number of Data Packets per time. The second experiment, evaluate the number of packets transmitted per time (throughput) for the three differences protocols. Figures 5.2 show the numbers of each packet established during the two simulations, the results demonstrate that DEH-WSN has throughput more improved and capable to transfer larger packets versus other protocols.

Tables 5.4 and 5.5 shows the performance evaluation of protocols and the rate parameters FND and HND in the two scenarios. The performance results of the protocol demonstrate that DEH-WSN is more effective for improving FND and HND and for transfering more packets against other protocols. Stability is increased by changing the possibility of nodes to be CH according to the energy status of nodes and the amount of harvested energy. CH consumes more energy consumption balanced between nodes during the simulation. The results prove the efficiency DEH-WSN in increasing of HND against other protocol during the simulation.

6. Conclusion. We presented and evaluated a new routing protocol that relies on matching between clustering and energy harvesting in a distributed manner, which aims to maximize network lifetime and to become unlimited, by using energy harvesting instead of energy efficient Clustering protocols. Starting with these aims, and motivated by the various studies of WSN algorithm, we introduced the model network and the energy harvesting model to match between distributed Clustering and energy harvesting. Based on this approach, we proposed our new Distributed energy Harvesting Clustering algorithm "DEH-WSN".

Moreover, in DEH-WSN, we use an estimation scheme to solve the average residual energy for the network that is recharged thanks to a uniform energy harvesting system. Our approach can be applied to the design of several types of wireless sensor network protocols that require network stability, since DEH-WSN can

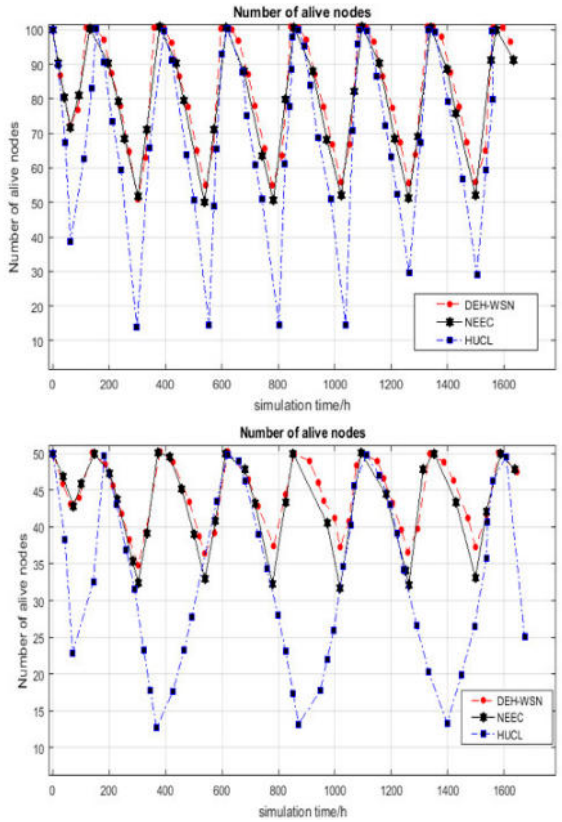


FIG. 5.1. *Number of Alive nodes per time for Scenarios 1 (top) and 2 (bottom)*

TABLE 5.4
Average of FND and HND for scenario 1.

Scenario 1	FND(100 Nodes)		HND(50 Nodes)	
	Time	Packets	Time	Packets
DEH-WSN	23 min	290.6	31 h: 23 min	2.6049e+04
NEEC	30 min	254.9	21 h: 24 min	27163.3
HUCL	24 min	124.2	8 h:30 min	3229.4

TABLE 5.5
Average of FND and HND for scenario 2.

Scenario 2	FND(100 Nodes)		HND(50 Nodes)	
	Time	Packets	Time	Packets
DEH-WSN	1 h:51 min	N	N	N
NEEC	1 h:48 min	876.5	N	N
HUCL	1 h:12 min	599.8	8 h:30 min	3431.6

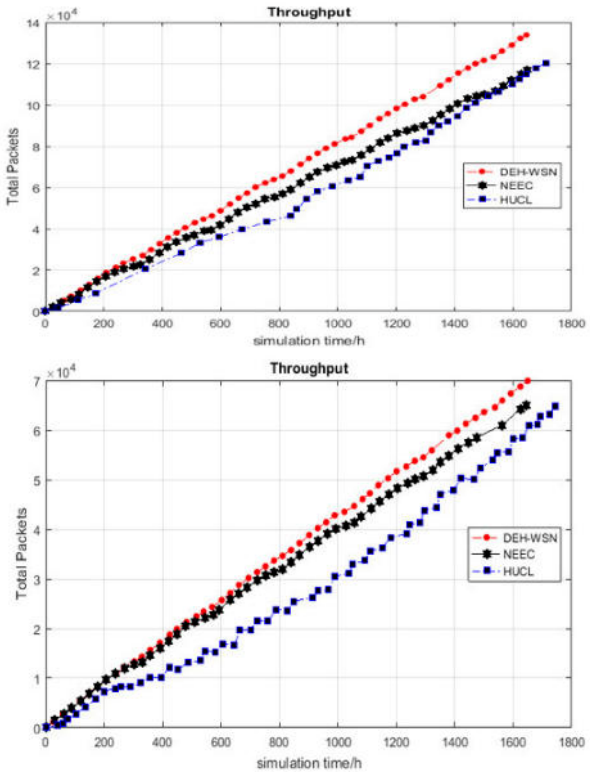


FIG. 5.2. *Number of packets made per time for Scenario 1 (top) and 2 (bottom)*

significantly improve the First Node Death (FND), the Half Node Death (HND), the number of alive nodes, the average energy, and the throughput. Furthermore, we presented two scenarios to evaluate this algorithm against other protocols. According to the simulation results, we demonstrate that the proposed algorithm balances the energy consumption, increases the number of available nodes and increases the number of repair packets in the BS. In future work, we can take different energy consumption for cluster head selection method to a heterogeneous wireless sensor network.

Acknowledgments. The author thanks the anonymous authors whose work largely constitutes this sample file. He also thanks the INFO-TeX mailing list for the valuable indirect assistance he received.

REFERENCES

[1] IAN F. AKYILDIZ AND MEHMET CAN VURAN. *Wireless sensor networks, volume 4, John Wiley & Sons*, 2010.
[2] IAN F. AKYILDIZ, WEILIAN SU, YOGESH SANKARA SUBRAMANIAM, AND ERDAL CAYIRCI. *Wireless sensor networks: A survey*, Computer networks, 38 (2002): pp. 393–-422.
[3] JENNIFER YICK, BISWANATH MUKHERJEE, AND DIPAK GHOSAL. *Wireless sensor network survey*, Computer networks, 52 (2008): pp. 2292–-2330.
[4] ALI A, KHELIL A, SHAIKH F, SURI N. *Efficient predictive monitoring of wireless sensor networks*, International Journal of Autonomous and Adaptive Communications Systems 5 (2012): pp. 233–254.
[5] WU Y , LIU W. *Routing protocol based on genetic algorithm for energy harvesting-wireless sensor networks*, IET Wireless Sens Syst (2013);3(2): pp. 112–-118.
[6] MAHER REBAI, MATTHIEU LE BERRE, HICHEM SNOUSSI, FAICEL HNAIEN, AND LYES KHOUKHI. *Sensor deployment optimization methods to achieve both coverage and connectivity in wireless sensor networks*, Computers & Operations Research, 59 (2015) pp. 11-–21.
[7] HEINZELMAN WR , CHANDRAKASAN A , BALAKRISHNAN H. *Energy-efficient communication protocol for wireless microsensor networks*, In Proceedings of the 33rd annual Hawaii international conference on system sciences (HICSS), Big Island, Hawaii, USA (2000), pp. 3005–3014.

[8] Younis Ossama, Fahmy Sonia. *Heed: a hybrid, energy-efficient, distributed clustering approach for ad-hoc sensor networks*, IEEE Trans Mobile Comput (2004) 3(4) pp. 660–-669.
[9] Ye Mao, Li Chengfa, Chen Guihai, Wu Jie. *EECS: an energy efficient clustering scheme in wireless sensor networks*, Ad Hoc Sensor Wirel Netw (2006) pp. 99-–119.
[10] Li Chengfa, Chen Guihai, Ye Mao, Wu Jie. *An uneven cluster-based routing protocol for wireless sensor networks*, Chin J Comput (2007) ,30(1) pp. 27-–36.
[11] Yu Jiguo, Qi Yingying, Wang Guanghui. *EDUC: an energy-driven unequal clustering protocol for heterogeneous wireless sensor networks*, J Control Theory Appl (2011), 9(1) pp.133–-139.
[12] Yu Jiguo, Qi Yingying, Wang Guanghui. *EADUC: an energy-aware distributed unequal clustering protocol for wireless sensor networks*, Int J Distrib Sensor Netw (2011), 7(1), pp. 1–8.
[13] Chen J , Li Z , Kuo YH. *A centralized balance clustering routing protocol for wireless sensor network*, Wireless Pers Commun (2013) ,72(1) pp. 623-–634.
[14] Malathi L , Gnanamurthy RK , Chandrasekaran K. *Energy efficient data collection through hybrid unequal clustering for wireless sensor networks*, Comput Electr Eng (2015) ,48 pp. 358–-370.
[15] Amgoth T , Jana PK. *Energy-aware routing algorithm for wireless sensor networks*, Comput Electr Eng (2015) ,41, pp. 357–-367.
[16] Gu X , Yu J , Yu D , Wang G , Lv Y. *ECDC: an energy and coverage-aware distributed clustering protocol for wireless sensor networks*, Comput Electr Eng (2014), 40(2), pp. 384-–398.
[17] Xu X , Xiao M , Yan W. *Clustering routing algorithm for heterogeneous WSN with energy harvesting*, Appl Mech Mater (2015) 73(3), pp. 734–-739.
[18] Xiao M , Zhang X , Dong Y. *An effective routing protocol for energy harvesting wireless sensor networks*, In: IEEE wireless communications and network conference (WCNC), (2013), pp. 2080–-2084.
[19] Yukun Y , Zhilong Y , Guan W. *Clustering routing algorithm of self-energized wireless sensor networks based on solar energy harvesting*, J China Univ Posts Telecommun (2015), 22(4), pp. 66-–73.
[20] S.M. Bozorgi, A.S. Rostami, A.A.R. Hosseinabadi, V.E. Balas. *A new clustering protocol for energy harvesting-wireless sensor networks*, Computers and Electrical Engineering (2017), pp. 1–15.
[21] Li J , Liu D. *DPSO-based clustering routing algorithm for energy harvesting wireless sensor networks*, In: International conference on wireless communications & signal processing (WCSP), (2015), pp. 1–-5.
[22] Seyed Mostafa Bozorgi, Amir Bidgoli. *HEEC: a hybrid unequal energy efficient clustering for wireless sensor networks*, Wireless Networks, (2018), pp. 1–22.
[23] Chong Han, Qing Lin, Jian Guo, Lijuan Sun, Zhuo Tao. *A Clustering Algorithm for Heterogeneous Wireless Sensor Networks Based on Solar Energy Supply* Electronics (2018), pp. 1–22.
[24] T. Han, S.M. Bozorgi, A.V. Orang, A.A.R. Hosseinabadi, A.K. Sangaiah and Mu-Yen Chen. *A Hybrid Unequal Clustering Based on Density with Energy Conservation in Wireless Nodes*, Sustainability (2019), 11, 746, pp. 1–26.
[25] Kaida Xu, Zhidong Zhao, Yi Luo, Guohua Hui, and Liqin Hu. *An Energy-Efficient Clustering Routing Protocol Based on a High-QoS Node Deployment with an Inter-Cluster Routing Mechanism in WSNs*, Sensors (2019), 19, 2752, pp. 1–23.
[26] J. John, P. Rodrigues. *A survey of energy-aware cluster head selection techniques in wireless sensor network*, Evolutionary Intelligence (2019), 19, 2752, pp. 1–13.
[27] Li Qing, Qingxin Zhu, and Mingwen Wang. *Design of a distributed energy efficient clustering algorithm for heterogeneous wireless sensor networks*, Computer communications, 29(12), (2006), pp. 2230-–2237.
[28] Wendi Beth Heinzelman. *Application-specific protocol architectures for wireless networks*, PhD thesis, Massachusetts Institute of Technology, (2000).
[29] Kansal A , Hsu J , Zahedi S , Srivastava MB. *ower management in energy harvesting sensor networks*, ACM Trans Embedded Comput Syst (2007) ,6(4), pp.1-–32.
[30] Yegnanarayanan V , Balas VE , Chitr G. *On certain graph domination numbers and applications*, Int J Adv Intell Paradigms (2014), 6(2), pp. 122–-135.
[31] Li J , Liu D. *DPSO-based clustering routing algorithm for energy harvesting wireless sensor networks*, In: International conference on wireless communications & signal processing (WCSP), (2015), pp. 1–-5.
[32] Olusanya, Olamide and Ogunseye, Abiodun. *The comparison of time division multiple access (TDMA) and wideband-code division multiple access system based on their modulation techniques*, International Journal of Computer Engineering (2014), 5(1), pp. 1–8.

Edited by: Dana Petcu
Received: Jul 7, 2020
Accepted: Dec 6, 2020

Scalable Computing: Practice and Experience, ISSN 1895-1767, http://www.scpe.org
© 2020 SCPE. Volume 21, Issues 4, pp. 649–660, DOI 10.12694:/scpe.v21i4.1799

SCIENTIFIC APPLICATIONS IN THE CLOUD: RESOURCE OPTIMISATION BASED ON METAHEURISTICS

ANAS MOKHTARI*, MOSTAFA AZIZI† AND MOHAMMED GABLI‡

Abstract. The advent of emerging technologies such as 5G and Internet of Things (IoT) will generate a colossal amount of data that should be processed by the cloud computing. Thereby, cloud resources optimisation represents significant benefits in different levels: cost reduction for the user, saving energy consumed by cloud data centres, etc. Cloud resource optimisation is a very complex task due to its NP-hard characteristic. In this case, use of metaheuristic approaches is more rational. But the quality of metaheuristic solutions changes by changing the problem. In this paper we have dealt with the problem of determining the configuration of resources in order to minimise the payment cost and the duration of the scientific applications execution. For that, we proposed a mathematical model and three metaheuristic approaches, namely the Genetic Algorithm (GA), hybridisation of the Genetic Algorithm with Local Search (GA-LS) and the Simulated Annealing (SA). The comparison between them showed that the simulated annealing finds more optimal solutions than those proposed by the genetic algorithm and the GA-LS hybridisation.

Key words: Cloud computing, Resources Management, Optimisation, Metaheuristic, Artificial Intelligence

AMS subject classifications. 68M14, 68T01, 90C10

1. Introduction.

The cloud computing is a distributed computer system based on an emergent technologies like the virtualisation. By comparing it with the conventional distributed computing, the latter has the objective of providing a collaborative sharing of resources to which users are linked, while the cloud computing has the objective of providing services or applications with ensuring of the scaling, the transparency (vis-a-vis the physical implementation of the cloud), the security, the supervision and the management.

There are three main types of cloud models:

IaaS (Infrastructure as a Service) offers a low level computing resources in the form of Virtual Machines (VM). EC2 [1] and Azure [3] are two IaaS examples provided respectively by Amazon and Microsoft.

PaaS (Platform as a Service) offers ready to use software platforms on which users develop and deploy their applications. Heroku [6] and Google App Engine [4] provide this type of cloud.

SaaS (Service as a Service) offers ready-to-use applications. It's the highest level in the cloud. G Suite [5] (Google) and Office 365 [7] (Microsoft) are SaaS models.

There are numerous applications that require High Performance Computing (HPC). Weather forecast, chemical process modeling, and the physics simulations are examples of such applications. Since the available computing resources in the cloud are very efficient, users of the HPC applications showed interest in running their intensive applications in the cloud environment.

Cloud computing face several challenges and problems that are subject to scientific research. The excessive consumption of the electric energy and the performance degradation of the application execution because of underestimation of reserved resources are examples of problems that cloud systems face.

This work addresses the problem of performance and cost of applications execution in the cloud. It's a double contradictory objectives problem. The first one is to maximise the resources to be used to have a good performance by reducing the execution time. The second one is to minimise the resources to be used to reduce the payment cost.

There are several works in the literature which have dealt with this type of problem. The complexity of the cloud resource optimisation problem has led researchers to use metaheuristics. For instance, in our

*MATSI Lab., ESTO, University Mohammed Ist, Oujda, Morocco (a.mokhtari@ump.ac.ma).
†MATSI Lab., ESTO, University Mohammed Ist, Oujda, Morocco (azizi.mos@ump.ac.ma).
‡LARI Lab., FSO, University Mohammed Ist, Oujda, Morocco (medgabli@ump.ac.ma).

TABLE 2.1
Definition of variables

Variable	Definition
P	Set of packages types offered by a cloud provider during a set of time periods
C_M	Set of consumer requirements as the maximum cost
T_M	Maximum time for execution
D_S	Disk storage
M_C	Memory capacity
G_f	Processing demand in Gflop
p	A package type from the set P ($p \in P$)
c_p	The cost of purchasing the package p for one period of time
d_p	Disk storage computing resource for the package p
m_p	Memory capacity for the package p
g_p	Processing power for the package p
N_M	Maximum limit of packages that a consumer can purchase at a period of time

previous works [20, 21, 22], we considered weighted objective functions in our model and chose the approach of genetic algorithms for dealing with this issue. The authors of [25] investigated the meta-heuristic resource allocation techniques used in the IaaS cloud computing environment. Jena et al. [26] proposed a hybridisation of modified Particle swarm optimization (MPSO) and improved Q-learning algorithm to load balancing of tasks on the cloud environment. Coutinho et al. [10] implemented an ILP model and then used Greedy Randomised Adaptive Search Procedure (GRASP) to solve the resource optimisation problem. Except that the proposed metaheuristics has been compared with a deterministic algorithm. Due to difficulty of this problem, which is classified as NP-hard [10], most of these works have tried to solve it using metaheuristic approaches. The effectiveness of these approaches in terms of the solutions found varies from one problem to another.

In this work, we used three metaheuristic algorithms: Genetic Algorithm (GA), Simulated Annealing (SA) and hybridisation of GA and Local Search method (GA-LS). Our goal is executing demanding HPC applications in the IaaS cloud.

The remaining of this paper is organised as follows: in the *Sect.* 2, we describe the mathematical model used. In the *Sect.* 3, we explain the three proposed resolution metaheuristics. In the *Sect.* 4, We compare between these approaches by experiments, then analyse the obtained results. We conclude in *Sect.* 5.

2. Mathematical model. In the infrastructure cloud (IaaS), resources are the virtual CPUs (vCPU), the memory, the storage drives, etc. of the VMs. Each resource type has the same characteristics as the equivalent physical resources (like processor frequency for the vCPU).

Cloud infrastructure service providers make available to the users a variety of VM choices. Each type has a specific computing power and usage price per hour.

The mathematical model that we present is based on the variables cited in the *Table* 2.1.

Our optimisation problem is composed of two objective functions: (i) minimise the total cost and (ii) minimise the execution time. We transformed them to a unique objective function by their sum with coefficients assigned to each of these two functions (weighted sum). So, we have [10]:

$$\text{(CC ILP)} \qquad min(\alpha f_1 + (1-\alpha) f_2) \tag{2.1}$$

with

$$f_1 = \sum_{p \in P} \sum_{i=1}^{N_M} \sum_{t \in T} c_p x_{pit} \tag{2.2}$$

and

$$f_2 = t_m \tag{2.3}$$

subject to

$$\sum_{p\in P}\sum_{i=1}^{N_M}\sum_{t\in T} c_p x_{pit} \leq C_M \tag{2.4}$$

$$\sum_{p\in P}\sum_{i=1}^{N_M} d_p x_{pit} \geq D_s x_{p'i't}, \qquad \forall t \in T, \forall p' \in P, \forall i' \in \{1, ..., N_M\} \tag{2.5}$$

$$\sum_{p\in P}\sum_{i=1}^{N_M} m_p x_{pit} \geq M_C x_{p'i't}, \qquad \forall t \in T, \forall p' \in P, \forall i' \in \{1, ..., N_M\} \tag{2.6}$$

$$\sum_{p\in P}\sum_{i=1}^{N_M}\sum_{t\in T} g_p x_{pit} \geq G_f \tag{2.7}$$

$$\sum_{p\in P}\sum_{i=1}^{N_M} x_{pit} \leq N_M, \qquad \forall t \in T \tag{2.8}$$

$$t_m \geq t x_{pit}, \qquad \forall t \in T, \forall p \in P, \forall i \in \{1, ..., N_M\} \tag{2.9}$$

$$x_{pit+1} \leq x_{pit}, \qquad \forall t \in T, \forall p \in P, \forall i \in \{1, ..., N_M\} \tag{2.10}$$

$$x_{pi+1t} \leq x_{pit}, \qquad \forall t \in T, \forall p \in P, \forall i \in \{1, ..., N_M - 1\} \tag{2.11}$$

$$x_{pit} \in \{0, 1\}, \qquad \forall t \in T, \forall p \in P, \forall i \in \{1, ..., N_M\} \tag{2.12}$$

$$t_m \in \mathbb{Z} \tag{2.13}$$

The terms α and $(1 - \alpha)$ represent the weights of the two objectives. To automate the choice of the weights and ensure a fair treatment between the two objective functions [12], we used dynamic weights to meet the condition (2.14):

$$|\alpha(t) f_1 - (1 - \alpha(t)) f_2| \prec \varepsilon, \tag{2.14}$$

where ε is a positive number in the vicinity of 0, t is a time-step, and $\alpha(t)$ and $(1 - \alpha(t))$ are the dynamic weights. $\alpha(t)$ is calculated by the formula (2.15):

$$\alpha(t) = \frac{|f_2(x_{t-1})|}{|f_1(x_{t-1})| + |f_2(x_{t-1})|} \tag{2.15}$$

where x_{t-1} is the best solution of the iteration $(t - 1)$ of the metaheuristic. For more details, see our previous work [20].

TABLE 3.1
The SA variables definitions.

Variable	Definition
S	Solution that represents a combination of packages.
Δ	The difference between the value of the optimisation function of the current solution and that under the evaluation.
T	System temperature.

3. Metaheuristic resolution approaches.

3.1. Simulated Annealing. The origin of this method come from the experiments done by Metropolis et al. [19] to simulate the stochastic evolution of such physical system. In the context of minimisation of the objective function f, the process of simulated annealing is illustrated in the Algorithm 5. *Table* 3.1 defines the used variables in this algorithm.

Algorithm 1 Simulated Annealing

```
1: Generate an initial solution S ← S_0
2: Initiate the temperature T ← T_0
3: repeat
4:     for a predetermined number of iterations do
5:         Generate a solution S_0 in the vicinity of S
6:         Calculate Δ = f(S_0) − f(S)
7:         if Δ < 0 then
8:             S ← S_0
9:         else
10:            S ← S_0 with the probability e^(−Δ/T)
11:        end if
12:    end for
13:    Decrease the temperature T
14: until The stop criterion is satisfactory
15: Return to the best configuration found
```

The proper functioning of the Simulated Annealing algorithm depends on the configuration space and the temperature decrease function. For example, Dréo et al. [11] and Kirkpatrick et al. [18] propose methods to define this function.

3.2. Genetic Algorithm. The Genetic Algorithm is an optimisation technique that mimic the natural evolution. The first work on the GA was developed by John Holland in 1962 [16]. The popularity of the GA returns to the work of David Golberg [14]. In this context, we call *chromosome* x any proposed solution by the GA. This chromosome is composed of a set of values called *genes*. The Algorithm 6 presents the second approach. *Table* 3.2 describes used variables.

The fitness function permits to evaluate each chromosome by assigning it a value that depends on the quality of this solution (chromosome). The crossover operation randomly chooses two chromosomes and a position of crossing. Then, we exchange the bits which are very close to the position of crossing. The mutation operation is to select a gene and replace its value by another randomly chosen from a given set.

3.3. Hybridisation of GA and Local Search method. The metaheuristic methods of optimisation are classified into two categories: approaches which aim at diversification and those which aim at intensification in the research space. The main difference between these two categories [13] is that in intensification, research focuses on the evaluation of neighbours of elite solutions, while diversification encourages the research process to evaluate regions not visited and to generate solutions that differ significantly from the solutions seen previously.

Algorithm 2 Genetic Algorithm

1: Initialise $t \leftarrow 0$. Randomly generate an initial population of individuals $P(0)$
2: Put $f(x,t) = \alpha(t)f_1(x) + (1-\alpha(t))f_2(x)$ and $\alpha(0) = 0.5$
3: **for** A predetermined number of iterations **do**
4: Put $t \leftarrow t+1$. Apply on each solution (chromosome) of the current population an evaluation by the fitness function f
5: Apply the selection operator on the current population $P(t)$ to produce a new population $P'(t)$
6: Apply the crossover operator on $P'(t)$ with a probability P_c. A new population $P''(t)$ is created
7: Apply the mutation operator on $P''(t)$ with a probability P_m. The result population is noted $P(t+1)$
8: Update the weight $\alpha(t)$ by using the dynamic method described by the formula (2.15)
9: **end for**

TABLE 3.2
Definitions of the genetic algorithm variables

Variable	Definition
$P(t)$	Population composed of several candidate solutions used in the iteration t.
x	A solution that represents a combination of packages.
P_c	Crossover probability.
P_m	Mutation probability.

GA are based on diversification, while the Local Search (LS) algorithm intensifies its search in the vicinity of a solution.

Evolutionary algorithms, including GA, suffer from the inability to intensify research enough. As a result, they cannot effectively achieve high quality candidate solutions [17]. A significant improvement in the performance of the GA for combinatorial problems can be ensured by the application of LS on the solution found by the GA. We talk about *hybridisation between GA and LS* (GA-LS).

The third approach is presented by the Algorithm 7. We are considering the improvement of the solution proposed by the GA using LS as a method of *intensification* [8]. The first part of this algorithm (lines 1 to 10) represents the GA. It consists in initialising a population of candidate solutions, then repetitively carrying out the operations of evaluation, selection, crossing and mutation. In the second part (lines 11 to 18), which represents the LS algorithm, the best solution found by the GA, noted s, is selected to become the initial configuration of LS.

4. Comparison of the Results.

4.1. Description of data. One of the most important parts of a comparison between metaheuristics is the testbed on which it is made [23]. It is preferable to develop a new test bed (program) to implement the metaheuristics and compare between them. In addition, so that the execution time of each algorithm is comparable to the others, the best approach to use consists in implementing these algorithms by the same programming language, compiling them in the same machine with the same parameters (flags) of compilation and run them in the same machine [23].

We implemented our approaches in *Java* language, then compiled the code by the *javac* compiler (OpenJDK), version 11.0.6, on a machine characterised by an Intel processor ® Core™ i7-2670QM which contains eight cores clocked at 2.20 GHz. We tested our programs on the same computer using the Java virtual machine (OpenJDK JRE), version 11.0.6.

We have applied these algorithms on four instances of applications with different sizes: an application which deals with the problem of manipulation of the biological sequence (*raxml*) [24], a typical analysis application for the CMS experience (*CMS-1500*) [9] and two applications for solving the QAP[1] problem by the separation and

[1]Quadratic Assignment Problem

Algorithm 3 Hybridisation algorithm between the genetic algorithm and the local search (GA-LS)

Set $t \leftarrow 0$
Randomly generate an initial population of individuals $P(0)$
Put $f(x,t) = \alpha(t)f_1(x) + (1-\alpha(t))f_2(x)$ and $\alpha(0) = 0.5$
for A predetermined number of iterations **do**
    Put $t \leftarrow t+1$. Apply on each solution (chromosome) of the current population an evaluation by the fitness function f
    Apply the selection operator on the current population $P(t)$ To produce a new population $P'(t)$
    Apply the crossover operator on $P'(t)$ with a probability P_c. A new population $P''(t)$ is created
    Apply the mutation operator on $P''(t)$ with a probability P_m. The result population is noted $P(t+1)$
    Update the weight $\alpha(t)$ by using the dynamic method described by the formula (2.15)
end for
Choose the solution $s \in S$ found by the genetic algorithm
repeat
    Choose s' in the vicinity of s ($s' \in V(s)$)
    $\Delta = f(s) - f(s')$
    if $\Delta > 0$ **then**
        $s \leftarrow s'$
    end if
until The stop criterion is satisfactory

TABLE 4.1
Characteristics of the applications used in the experiment [10]

Application	Memory (GB)	Storage (GB)	GFLOP	Time (hour)	Max packages	Cost ($)
raxml	3	2	3,317,760	10	20	50
cms-1500	2,250	30	324,000,000	10	20	75
nug28-cbb	528	528	541,765,325	10	20	120
nug30-cbb	918	918	967,438,080	15	30	250

evaluation algorithm (*nug28-cbb* and *nug30-cbb*) [15]. The characteristics of these applications are summarised in the *Table* 4.1.

In order to have realistic results, we applied our simulation to the offers of two cloud computing providers: Amazon EC2 [1] and Google Compute Engine [2] (*cf. Table* 4.2).

For each application, we performed the metaheuristics several times to ensure the accuracy of the results obtained. Details are presented in *Table* 4.3.

To give metaheuristic approaches more chance of finding good solutions, we have extended the execution time for larger application problems (between 1 and 20 seconds, *cf. Table* 4.3-left). In addition, in order to verify the influence of this duration on the quality of the results obtained, we launched other tests with a long execution time (*cf. Table* 4.3-right).

Since the LS depends on the solution found after the execution of the GA, and in order to be able to compare between the results of the GA and GA-LS hybridisation approaches, we have allocated to the LS part an execution time equivalent to approximately 8% of the execution time of the GA alone:

$$\text{Duration(GA-LS)} \approx 1.08 \times \text{duration(GA)} \Rightarrow \text{Duration(GA-LS)} \approx \text{duration(GA)}$$

Generally, the parameters values of each algorithm must be determined by its designer because their change influences the performance of metaheuristics [23]. For the case of our approaches, we fixed the parameters

TABLE 4.2
Cloud Provider's VM Instance Pricing

Cloud provider	Instance type	vCPU	Memory (GB)	Storage (GB)	Price (/hour)
Amazon EC2	c4.large	2	3.75	200	$0.128
	c4.xlarge	4	7.5	400	$0.255
	c4.2xlarge	8	15	800	$0.509
	c4.4xlarge	16	30	1,600	$1.018
	c4.8xlarge	36	60	2,500	$1.938
Google Compute Engine	n1-highcpu-2	2	1.80	100	$0.0764
	n1-highcpu-4	4	3.60	200	$0.1529
	n1-highcpu-8	8	7.20	400	$0.3058
	n1-highcpu-16	16	14.40	800	$0.6116
	n1-highcpu-32	32	28.80	1,600	$1.2233
	n1-highcpu-64	64	57.60	2,500	$2.4077

TABLE 4.3
Parameters used for the execution of the three approaches: GA, GA-LS hybridisation, and SA: for a no long duration (left) and for a long duration (right)

Application	Execution time (s)	Number of trials
raxml	1	10
cms-1500	5	10
nug28-cbb	10	10
nug30-cbb	20	10

Application	Execution time (s)	Number of trials
raxml	180	2
cms-1500	180	2
nug28-cbb	180	2
nug30-cbb	180	2

according to the application concerned. The details are presented in *Table* 4.4 for GA and GA-LS, and *Table* 4.5 for SA.

4.2. Results analysis. One of the most important points in the comparison between metaheuristics is the *solution quality*. The measurement of this quality is relative and it depends on the treated application. For long-term planning, the acceptable difference between the found and the optimal solutions is smaller than that of short-term planning applications [23]. For that, we must compare between the solutions found by metaheuristics and not determine if they reach a threshold of solution quality [23].

To compare the solutions found by metaheuristics, we must base ourselves on a metric. In our case, we have the possibility of using one of the following two metrics [23]:

- Deviation from best-known solutions for a problem;
- Deviation between the algorithms being compared: This method has the advantage of making the comparison between the algorithms very explicit. However, these comparisons lack any sense of the actual error of solutions.

In our knowledge, there is no best-known solutions for the case of our problem. So, we used the second metric.

The results of our experiment are detailed in the *Table* 4.6. We have limited ourselves in this table to the best solution found in the ten trials carried out by application.

In the *Application* column, we put the name of the problem instance to be executed with the name of the cloud on which this execution is planned. For example, the first line (*raxml_am*) concerns the execution of raxml instance in the Amazon EC2 (am) cloud, while for the fifth line (*raxml_go*), we are talking about the same application in the Google Compute Engine (go) cloud.

The details of the solution for each metaheuristic are represented in five columns: the payment cost f_1

TABLE 4.4
Parameters used for the GA

Problem instance	**Population size**	**Crossover probability (P_c)**	**Mutation probability (P_m)**
raxml	40	0.5	0.01
cms-1500	40	0.5	0.01
nug28-cbb	40	0.5	0.01
nug30-cbb	60	0.5	0.01

TABLE 4.5
Parameters used for the SA

Problem instance	**Annealing rate (λ)**	**Initial temperature**
raxml	0.9	100
cms-1500	0.9	100
nug28-cbb	0.9	100
nug30-cbb	0.9	100

(in dollars) of the found solution, the duration f_2 (in hours) necessary for the execution of the application, the value of the weighted objective function f, f_{avg} to check the robustness of this solution by calculating the average of the values of f for the ten trials carried out ($f_{avg} = \sum_{i=1}^{10} \frac{f_{trial_i}}{10}$), and the fifth column contains the execution time of the metaheuristic (in seconds) which allowed to have this result.

From *Fig.* 4.1, we can easily notice that, for all the tested applications, the best minimisation of the objective function f is ensured by the SA, followed by the GA-LS hybridisation algorithm. The optimal cost of f found by the SA is 53.34% (nug30-cbb_go) to 92.3% (raxml_go) less than that found by the GA, and 29.27% (nug28-cbb_go) to 90.57% (raxml_go) less than the GA-LS.

The fact that the GA-LS optimises the objective function better than the GA is expected since the GA-LS diverts the weakness of the GA in terms of intensification.

Since the function f is the result of the weighted sum of the two functions f_1 and f_2 ($f = \alpha f_1 + (1-\alpha) f_2$), we also analysed and compared these two objectives for the three metaheuristic approaches. The histograms in *Fig.* 4.2 and *Fig.* 4.3 represent, respectively, the values of f_1 and f_2.

Simulated Annealing gives us, for all the tested applications, cheaper solutions (f_1) and faster execution (f_2) than those proposed by the GA and the GA-LS. Solutions of the GA-LS hybridisation algorithm are more expensive than those found by the GA (case of nug28-cbb_go and nug30-cbb_go in *Fig.* 4.2), but are always faster in execution (*cf. Fig.* 4.3).

Since the aforementioned results represent the best solutions found among the ten trials that we carried out, this brings us back to verifying that these solutions do not represent particular cases (*noise*). The robustness of the GA is presented in *Fig.* 4.4. The red (dark) bar represents the value f and the red+blue (dark+light) bar represents f_{avg}. The best values of f are 20.5% to 46.35% smaller than the average f_{avg} of the solutions found by the ten trials. Similarly, f of the GA-LS (*cf. Fig.* 4.5) is 20.6% to 49.79% lower than f_{avg}. This variation is more reduced in the case of the SA (*cf Fig.* 4.6) since it is between 0.01% and 29.72%. The long-term execution of the three algorithms (180 seconds) did not considerably improve the quality of the found solutions (details in the *Table* 4.6). In this case too, the SA gives in all the tested cases more optimal solutions than those of the GA and the GA-LS.

5. Conclusion. In this paper, we treated the problem of multi-objective optimisation of the cloud computing resources, for the execution of the intensive computing applications. We took into account the decrease in the budget by reducing the allocation price of computing resources (first objective) and the increase in performance by minimising the execution time (second objective). Then, we implemented three problem solving

TABLE 4.6

Results obtained by the GA, the GA-LS hybridisation and the SA. The first half of the results (8 lines) represent the solutions found by our algorithms executed in durations between 1 and 20 seconds. The second half represents the tests of these algorithms in long durations (180 seconds). The values of f_1, f_2 and f represent the best solution found among the 10 tests launched. The columns of f_1 represent the estimated cost of payment in dollars. The columns of f_2 are the execution times of the application in question, estimated in hours. f is the fitness function (objective function) of the solution found. To ensure the robustness of these solutions, we added the columns of f_{moy} which represent the average of the values of the objective functions of the 10 tests carried out ($f_{moy} = \sum_{i=1}^{10} \frac{f_{test_i}}{10}$).

Application	**Genetic Algorithm**					**GA-LS Hybridisation**					**Simulated Annealing**				
	f_1	f_2	f	f_{avg}	Duration of execution (s)	f_1	f_2	f	f_{avg}	Duration of execution (s)	f_1	f_2	f	f_{avg}	Duration of execution (s)
raxml_am	45.5523	4	7.1108	13.2552	1	5.0931	3	3.2692	6.5115	1.08	0.6368	1	0.7781	0.8486	1
CMS-1500_am	60.8255	8	12.1523	15.2860	5	56.7662	3	7.5903	9.5729	5.36	52.5599	2	3.8533	5.4826	5
nug28-cbb_am	99.7775	4	7.5860	13.9686	10	91.9959	3	6.3321	9.7177	10.7	89.1166	2	3.9122	5.4203	10
nug30-cbb_am	167.3066	7	13.4763	22.8306	20	166.3851	4	10.5603	14.6710	21.5	157.5061	3	5.8878	7.2271	20
raxml_go	28.3623	6	9.0497	12.3524	1	28.2121	3	7.3921	9.0969	1.61	0.5351	1	0.6972	0.7684	1
CMS-1500_go	65.3562	5	9.3652	15.4298	5	53.7482	3	6.6703	9.0542	5.368	46.2074	2	3.8340	5.2608	5
nug28-cbb_go	77.7018	7	12.3838	16.9321	10	79.4214	3	8.1654	10.2845	10.745	77.2029	3	5.7755	5.7760	10
nug30-cbb_go	141.8158	7	12.5855	22.7710	20	154.4203	4	10.2320	13.7650	22.004	137.6620	3	5.8720	7.2010	20
raxml_am	44.8237	6	11.0931	11.0931	180	1.0222	1	1.0039	1.8485	193.404	0.6368	1	0.7781	0.7781	180
CMS-1500_am	73.7056	6	11.4818	12.2660	180	54.1620	2	6.2233	8.1126	193.479	54.6756	1	1.9640	1.9640	180
nug28-cbb_am	92.8217	8	14.8691	16.6128	180	95.6304	2	9.5825	9.8297	193.634	89.4684	2	3.9125	3.9131	180
nug30-cbb_am	178.2715	12	20.6893	22.2003	180	174.4713	3	11.9611	13.6688	194.847	156.4287	3	5.8870	5.8875	180
raxml_go	45.6386	5	9.6928	9.6928	180	10.0532	2	3.1836	3.3291	192.218	0.5351	1	0.6972	0.6972	180
CMS-1500_go	62.9901	8	14.1969	15.1925	180	46.4005	2	7.0035	7.5797	193.569	46.0908	2	3.8336	3.8337	180
nug28-cbb_go	102.7015	9	16.9870	17.1702	180	77.5516	3	9.0945	9.0977	195.795	77.2404	2	3.8990	3.8990	180
nug30-cbb_go	150.7867	8	15.3137	17.4729	180	151.8142	2	9.6737	12.5265	20.0483	137.7398	3	5.8721	5.8721	180

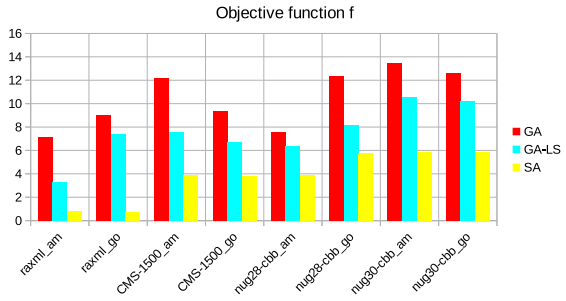


FIG. 4.1. *Comparison of the objective function values found by the GA, the GA-LS hybridisation and the SA.*

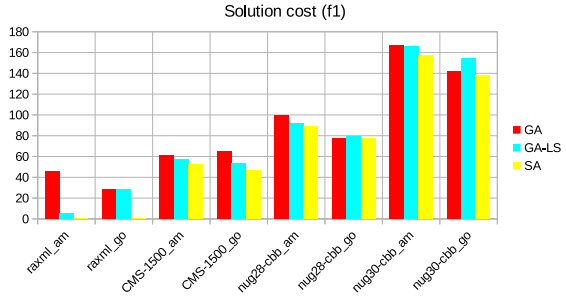


FIG. 4.2. *Comparison of payment costs of solutions found by the GA, the GA-LS hybridisation and the SA.*

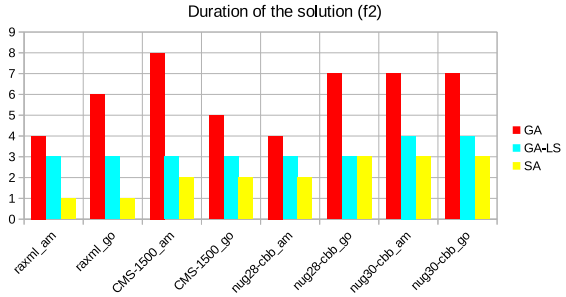


FIG. 4.3. *Comparison of the durations of the solutions found by the GA, the GA-LS hybridisation and the SA.*

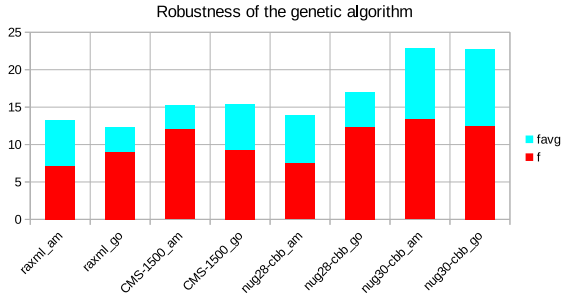


FIG. 4.4. *Robustness of the GA verified by the comparison between the best solution found (f) and the average of the solutions of the ten tests carried out (f_{moy}).*

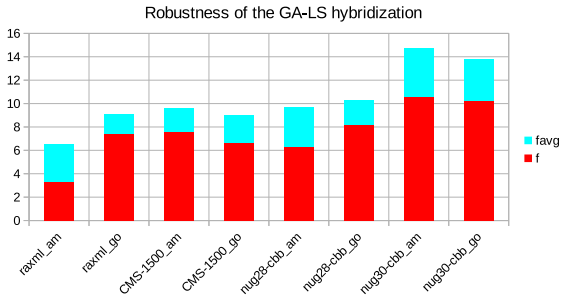


FIG. 4.5. *Robustness of the GA-LS hybridisation verified by the comparison between the best solution found (f) and the average of the solutions of the ten tests carried out (f_{avg}).*

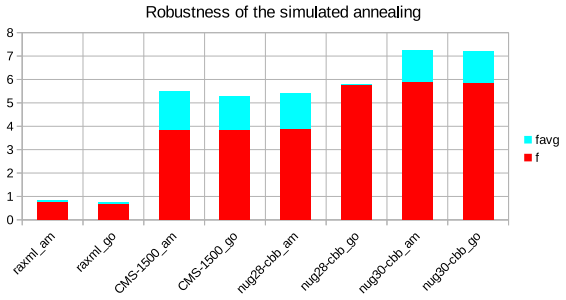


FIG. 4.6. *Robustness of the SA verified by the comparison between the best solution found (f) and the average of the solutions of the ten tests carried out (f_{avg}).*

metaheuristics, namely the Genetic Algorithm, the hybridisation between the Genetic Algorithm and Local Search, and the Simulated Annealing. To compare between these approaches, we launched simulations on different sizes applications. The obtained results showed that the Simulated Annealing finds more optimal solutions than those proposed by the Genetic Algorithm and the hybridisation between the Genetic Algorithm and Local Search.

In the future, we plan to extend our work by comparing the Simulated Annealing with other metaheuristics, and test them in a different context such as the multi-cloud.

REFERENCES

[1] *https://aws.amazon.com/fr/ec2/*, Oct. 2017.
[2] *https://cloud.google.com/compute/*, Oct. 2017.
[3] *https://azure.microsoft.com/fr-fr/*, Feb. 2020.
[4] *https://cloud.google.com/appengine/*, Feb. 2020.
[5] *https://gsuite.google.fr/intl/fr/*, Feb. 2020.
[6] *https://www.heroku.com/*, Feb. 2020.
[7] *https://www.office.com/*, Feb. 2020.
[8] C. BLUM AND A. ROLI, *Metaheuristics in combinatorial optimization: Overview and conceptual comparison*, ACM computing surveys (CSUR), 35 (2003), pp. 268–308.
[9] C. COLLABORATION, S. CHATRCHYAN, G. HMAYAKYAN, V. KHACHATRYAN, A. SIRUNYAN, W. ADAM, T. BAUER, T. BERGAUER, H. BERGAUER, M. DRAGICEVIC, ET AL., *The cms experiment at the cern lhc*, 2008.
[10] R. D. C. COUTINHO, L. M. DRUMMOND, AND Y. FROTA, *Optimization of a cloud resource management problem from a consumer perspective*, in European Conference on Parallel Processing, Springer, 2013, pp. 218–227.
[11] J. DRÉO, A. PÉTROWSKI, P. SIARRY, AND E. TAILLARD, *Métaheuristiques pour l'optimisation difficile*, 2003.
[12] M. GABLI, E. M. JAARA, AND E. B. MERMRI, *A genetic algorithm approach for an equitable treatment of objective functions in multi-objective optimization problems.*, IAENG International Journal of Computer Science, 41 (2014).
[13] F. GLOVER AND M. LAGUNA, *Tabu search kluwer academic*, Boston, Texas, (1997).

[14] D. Goldberg, *Genetic algorithms in search, optimization, and machine learning, addison-wesley, reading, ma, 1989*, NN Schraudolph and J, 3 (1989).
[15] A. D. Goncalves, L. M. Drummond, A. A. Pessoa, and P. Hahn, *Improving lower bounds for the quadratic assignment problem by applying a distributed dual ascent algorithm*, arXiv preprint arXiv:1304.0267, (2013).
[16] J. H. Holland, *Outline for a logical theory of adaptive systems*, Journal of the ACM (JACM), 9 (1962), pp. 297–314.
[17] H. H. Hoos and T. Stützle, *Stochastic local search: Foundations and applications*, Elsevier, 2004.
[18] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, *Optimization by simulated annealing*, science, 220 (1983), pp. 671–680.
[19] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, *Equation of state calculations by fast computing machines*, The journal of chemical physics, 21 (1953), pp. 1087–1092.
[20] A. Mokhtari, M. Azizi, and M. Gabli, *Optimizing management of cloud resources towards best performance for applications execution*, in 2017 First International Conference on Embedded & Distributed Systems (EDiS), IEEE, 2017, pp. 1–5.
[21] ———, *Multi-cloud resources optimization for users applications execution*, in International Conference Europe Middle East & North Africa Information Systems and Technologies to Support Learning, Springer, 2018, pp. 588–593.
[22] ———, *A fuzzy dynamic approach to manage optimally the cloud resources*, in International Conference on Innovative Research in Applied Science, Engineering and Technology, IEEE, 2020.
[23] J. Silberholz and B. Golden, *Comparison of metaheuristics*, in Handbook of metaheuristics, Springer, 2010, pp. 625–640.
[24] A. Stamatakis, *Raxml-vi-hpc: maximum likelihood-based phylogenetic analyses with thousands of taxa and mixed models*, Bioinformatics, 22 (2006), pp. 2688–2690.
[25] S.H.H. Madni, M.S.A. Latiff, Y. Coulibaly and S.I.M. Abdulhamid, *An appraisal of meta-heuristic resource allocation techniques for IaaS cloud*, Indian Journal of Science and Technology, 2016, 9(4), 1-14.
[26] U.K. Jena, P.K. Das and M.R.Kabat, *Hybridization of meta-heuristic algorithm for load balancing in cloud computing environment*, Journal of King Saud University-Computer and Information Sciences, 2020.

Edited by: Dana Petcu
Received: Aug 14, 2020
Accepted: Dec 6, 2020


Scalable Computing: Practice and Experience, ISSN 1895-1767, http://www.scpe.org
© 2020 SCPE. Volume 21, Issues 4, pp. 661–672, DOI 10.12694:/scpe.v21i4.1805

A DYNAMIC PREDICTION FOR ELASTIC RESOURCE ALLOCATION IN HYBRID CLOUD ENVIRONMENT

VIPUL CHUDASAMA, MADHURI BHAVSAR*

Abstract. Cloud applications heavily use resources and generate more traffic specifically during specific events. In order to achieve quality in service provisioning, the elasticity of resources is a major requirement. With the use of a hybrid cloud model, organizations combine the private and public cloud services to deploy applications for the elasticity of resources. For elasticity, a traditional adaptive policy implements threshold-based auto-scaling approaches that are adaptive and simple to follow. However, during a high dynamic and unpredictable workload, such a static threshold policy may not be effective. An efficient auto-scaling technique that predicts the system load is highly necessary. Balancing a dynamism of load through the best auto-scale policy is still a challenging issue. In this paper, we suggest an algorithm using Deep learning and queuing theory concepts that proactively indicate an appropriate number of future computing resources for short term resource demand. Experiment results show that the proposed model predicts SLA violation with higher accuracy 5% than the baseline model. The suggested model enhances the elasticity of resources with performance metrics.

Key words: Cloud Computing, Autoscaling, Elasticity, SLA violation, Hybrid cloud

AMS subject classifications. 68M14

1. Introduction. Cloud computing technology enables users to access data through a virtual environment. Data centers are nowadays offering high-performance cloud services to the users as per demand. Accessibility of data is quick and cost-effective due to the elasticity and pay per use model [1] . The majority of the utility services, including logistics to the health-care support, are using resources provided by the cloud. In Smart Cities, the Internet of Things (IoT) enabled applications deployed in the cloud to use computing processes and resources such as CPU, software, and hardware devices [3]. One of the features of the cloud is on-demand services in which resources are provided as per demand, which increases customer satisfaction. In the purest form, cloud computing provides solutions by storing and accessing data or programs as service for users.

In many organizations, data resides in local infrastructure like small clusters. Organizations develop policies to access data for their employees. Due to the expansion and other factors, demand for resource virtualization is increasing day by day. A hybrid cloud is an affordable solution to deal with a burst in requirements for specific events [2]. In the cloud bursting model, an application that is running in local infrastructure and bursts to a public cloud for more resources. This type of Hybrid model has the advantages of cost reduction and scalability with data sensitivity [4].

In such scenarios, the elasticity of services to satisfy the need for resources increases user satisfaction. In order to achieve elasticity of the services and promised QoS, there is a need for resource management in Cloud Data centers. As per average demand, CPU utilization is 15 % to 20 % in normal state and follows a linear relationship in peak demand [5]. In such dynamics, requirements for resources are market-driven. So the main goal is to provide Elastic services with a dynamic policy which adds or removes storage and computing resources to enhance the application performance. Major Cloud providers (AWS,Google) provide elasticity features based on some metrics (CPU utilization, Memory). The use of elasticity through such auto scale mechanisms can satisfy the peak demand for application and guarantees QoS requirement.

Dynamic provision of resources is one of the complex tasks in distributed systems. Distributed systems such web servers, big-data cluster and grid require efficient resource management to provide the elastic services. Fig. 1.1 represents a scenario of web servers in data centers which provide cloud service. The data center

*Department of Computer Science and Engineering, Nirma University, Ahmedabad, India (vipul.chudasama@nirmauni.ac.in, madhuri.bhavsar@nirmauni.ac.in)

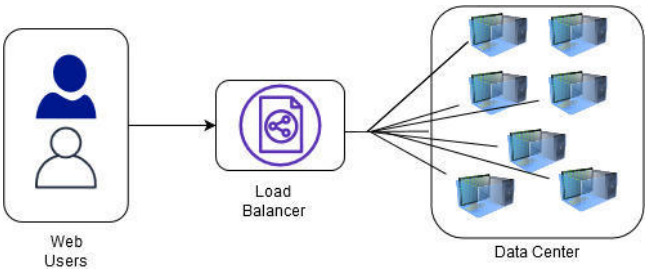


FIG. 1.1. *Elastic Cloud service*

manages the allocation and deallocation of servers through auto-scale to optimize resources. Flexible allocation and deallocation of resources are achieved with an auto-scale mechanism. An auto-scale mechanism is deployed on every workload in a cloud to maintain the resources in a balanced state. The balanced state of resources will help providers frame the policies to scale down idle resources and save energy consumption. The role of cloud providers with such optimizations in their operational environment helps in Green computing while maintaining Service Level Agreement (SLA) with end users [6]. Such an auto-scale mechanism is further divided in two classes: (i) Reactive approach, where static rules trigger to match the requirement of resources; (ii) Proactive approach, which is used to forecast the workload to meet the resource demand. In both classes, highly variable workload patterns are used to get future resource utilization demands, which is a challenging task. Such foreseen future demand helps allocate applications to the system that will improve the utilization with the help of time series analysis [15].

In recent works, autoregressive models such as ARIMA, AR, ARMA are proposed to predict metrics (system load) from monitoring service [8], which follows linear patterns. In Linear models, a dependent variable that is regressed on a number of independent variables while some non-linear features cannot be interpreted. However, non-linear patterns of resource usage are also addressed in some proposals to forecast future demand[9][10]. The neural network-based regression models are used to capture the non-linearity of resource usage. Our contribution in this work is to propose a hybrid model with Deep learning (LSTM) and Queuing theory to estimate resource requirements. We have considered the user's feedback to optimize the mechanism, which was not considered in previous studies. The main contributions of the paper are as follows:

- Design of an auto-scaling method based on deep learning is proposed to enhance the service through resource management.
- An LSTM based regression model to predict host load is presented.
- A queuing model to forecast the number of resources under the provision in the hybrid cloud.
- An assessment using real-server load information is given.

The rest of the paper is organized as follows: Section 2 discusses related work on autoscale strategies using Machine learning(ML) for workload prediction. Section 3 defines the proposed predictive approaches Section 4 discusses proposed predictive model and algorithm , Section 5 discuss experiment and analysis of proposed work, Section 6 and 7 discuss results and conclusions.

2. Related Work. The majority of research focuses on auto scaling mechanisms to manage the resources of Cloud. Web application workload is highly dynamic in nature. Recent studies in [11-13] have provided extensive review of resource management approaches.

As discussed in [14], the workload can be classified into five different classes such as once-in-a-lifetime, static, continuously changing, periodic, and unpredictable. Bayesian classifiers consist of probabilistic gives good results on CPU intensive tasks and the Markov method provides reliability for memory-intensive tasks. Auto-scaling of web applications with time series forecasting methods are discussed in [15-17]. A workload factoring technique was discussed in [18] for a hybrid cloud where a threshold-based technique was used to classify into two classes. The classes capture the base workload and flash workload. Threshold-based techniques are also applied by public cloud providers(Amazon EC2,Microsoft). The policies are framed based on the metrics provided, to scale up the resources or scale down resources of the environment. Improvement in the threshold-

based policy was suggested in [19] at fine-grain level by adding more levels in order to apply the decision of scaling the resources.

A Neural Network based workload prediction which implemented differential evolution and particle swarm optimization to estimate the workload was discussed in [20]. A Reinforcement Learning (RL) is a technique in which decision-making agents learn and decide best action without prior knowledge of the system. Actions are in the form of adding or removing the resources to obtain the highest rewards (e.g minimum response time, maximum throughput). For the cloud environment, authors [21] applied a Q-learning algorithm with an optimal scaling policy to reduce the execution time. Machine learning-based techniques like Support Vector Machine (SVM), Artificial Neural Network (ANN) model, and deep learning have found to provide excellent performance with time series data based on the minimization principle [27-28]. A number of authors have estimated resource scaling decisions with time series techniques like auto-regression, moving average and exponential smoothing. Machine learning models explore techniques like K-Nearest Neighbour (KNN), Support Vector Machine (SVM), Neural Network (NN), and Linear Regression which involve analysis, training, and prediction of metrics to optimize resource management in cloud [29].

The Deep Learning techniques are also preferred to explore features of Web traffic. Resource allocation and power management framework is discussed in [23] to predict the workload using long short-term memory (LSTM) recurrent neural network. The workload prediction mechanism with four LSTM units to improve the accuracy of the predictor has been presented in [24]. Another proposal has been presented to predict the workload of VM using DL approach [25]. Deep learning explores representational learning to generate the model which estimates the future values of workload in the cloud [30]. Queue Network Model (QN) can be used to analyse the performance of a distributed system. In a Distributed system, multiple servers handle the client requests with the optimization of parameters such as average queue time and average response time. These methods can be an open queue or closed queue. In [26], the authors proposed a proactive framework to improve the QoS of web application users' workload behaviour to allocate the virtual machines (VM) using the queuing model M/G/m.

Overall, the goal of proactive approaches suggested by Machine learning is to capture patterns or trends in the historical data. Prediction metrics obtained by these proactive approaches improve resource allocation in the cloud. In order to achieve elasticity, we explored the proactive approaches which provide future demand for resources and extended it with a queuing approach to propose a performance model in cloud.

3. The proposed predictive approaches. In Predictive techniques, time series data must be organized in a training set by splitting it with time series variable (T). So time series training set is defined as input and output set X, Y, respectively. Here we have applied a sliding window approach to process log data. Also, data is normalized before the process.

$$X = \begin{bmatrix} x_1 & x_2 & \dots & x_k \\ x_2 & x_3 & \dots & x_{k+1} \\ \vdots & & & \\ x_{n-1} & x_{n-2} & & x_{n-k} \end{bmatrix}, Y = \begin{bmatrix} x_{k+1} \\ x_{k+2} \\ \vdots \\ x_n \end{bmatrix}$$

SVM (Support Vector Machine) and Deep Learning (DL) are state-of-the-art machine learning methods to solve problems having time series data. SVMs learn from non-linear training data and map high dimensional feature space using kernel functions. An SVM regression model explores the number of requests application on a host to infer new feature space with the following function:

$$\hat{y}_t = b + \sum_{m=1}^{M} w_m \times K\left(x_t, x_m\right) \tag{3.1}$$

where w_m are the weight vector (W) and x_t is the time series data window at time t and b is a constant (bias term); and K is the kernel function. The optimal weight vector, W, is output of SVM to minimizes the regularized risk, R_{reg}, defined in equation 3.2. The Vapnik loss measured as error between predicted and actual

data is defined in equation 3.3. Here constant C and ϵ are chosen by the user and are data dependent [34].

$$R_{reg} = \frac{1}{2}\sum_{m=1}^{M} w_m^2 + C\sum_{m=1}^{M} L_\epsilon(y_m, \hat{y}_m) \tag{3.2}$$

$$L_\epsilon\left(y_m, \hat{y}_m\right) = \begin{cases} |y_m - \hat{y}_m| - \epsilon & \text{if} |y_m - \hat{y}_m| > \epsilon \\ 0, & \text{otherwise} \end{cases} \tag{3.3}$$

The main objective of kernel function is to transform input data into high-dimensional feature space. Linear, Polynomial and Gaussian are the most popular kernels. The selection of the kernel (K) is based on the underlying problem. We have used linear and Polynomial kernels for time series data in this work.

$$K\left(x, y\right) = \left(x^T \cdot y + c\right) \tag{3.4}$$

$$K\left(x, y\right) = \left(x \cdot y + 1\right)^p \tag{3.5}$$

Deep learning (DL), one of the classes of ANN (Artificial Neural Network) which exploit learning ability using multilayer deep networks and improve the influence of NNs (Neural Network). A standard NN mimics the working of the brain to process information using units called neurons. The neurons are triggered by a peripheral vision sensor and some neurons are triggered by the weighting of the previously active neurons. The neural learning network will have a collection of values for weights between neurons utilizing information flowing through them. Communication between neurons is done using forward direction to process information and generate output for the next layer. The non linear function f which perform this task is given in equation 3.6, where b is the bias and weights of connections defined by w_i.

$$f\left(W^t x\right) = f\left(\sum_{i=1}^{n} W_i x_i + b\right) \tag{3.6}$$

The most common activation functions are Sigmoid function, hyperbolic tangent function (*tanh*), and rectified linear function (*ReLU*). Their formulas are as follows:

$$f\left(W^t x\right) = Sigmoid\left(W^t x\right) = \frac{1}{1 + exp\left(-W^t x\right)} \tag{3.7}$$

$$f\left(W^t x\right) = tanh\left(W^t x\right) = \frac{e^{W^t x} - e^{-W^t x}}{e^{W^t x} + e^{-W^t x}} \tag{3.8}$$

$$f\left(W^t x\right) = Relu\left(W^t x\right) = max\left(0, W^t x\right) \tag{3.9}$$

A recurrent neural network (RNN) is considered as a sub class of artificial neural networks where temporal sequence of data are modeled with nodes which form a directed graph.

One of the improvement proposed in RNN to handle problem of vanishing gradient by in incorporating LSTM(Long short-term memory) network architecture Fig. 3.1. The traditional LSTM contains recurrent connections with input gates, forget gates,output gates and connected to output layer. In LSTM computation is performed on input/output training set with activations,using following formulas:

$$i_t = \sigma(W_{ix} x_t + W_{im} m_{t-1} + W_{ic} c_{t-1} + b_i \tag{3.10}$$

$$f_t = \sigma(W_{fx} x_t + W_{mf} m_{t-1} + W_{cf} c_{t-1} + b_f) \tag{3.11}$$

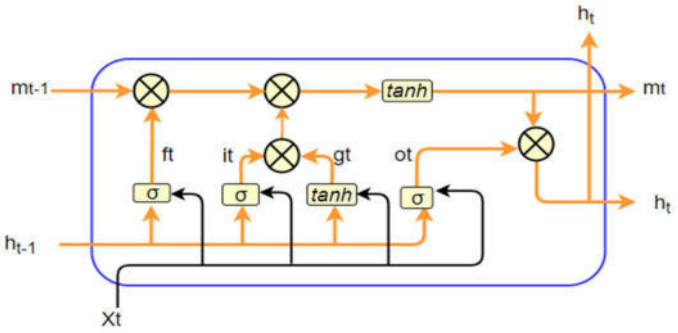


FIG. 3.1. *Architecture of an LSTM unit*

$$c_t = f_t \odot c_{t-1} + i_t \odot g(W_{cx}x_t + W_{cm}m_{t-1}) \tag{3.12}$$

$$o_t = \sigma(W_{ox}x_t + W_{om}m_{t-1} + W_{oc}c_t + b_o) \tag{3.13}$$

$$m_t = o_t \odot h(c_t) \tag{3.14}$$

$$y_t = W_{ym}m_t + b_y \tag{3.15}$$

For the proposed work, we have explored Bidirectional LSTM which is an extension of traditional LSTMs and Bidirectional RNN [32]. With Bidirectional LSTM, the model takes input sequences in a bidirectional way and concatenates interpretations which boost prediction efficiency [36]. This can provide additional context to the network and results in faster and enhanced learning on current problems. The dataset is split in 80 and 20 for training and testing respectively.

Different error measures can be used to evaluate the accuracy of the Predictive models. Some of the most common error measures are the following:

– Mean Absolute Error (MAE):

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{3.16}$$

– Mean Squared Error (MSE):

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{3.17}$$

– Root Mean Squared Error (RMSE):

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{3.18}$$

From the above error measures we obtain training error and real error when applied to the training set and testing set of time series data.

3.1. Queuing model. Cloud Computing provides an elastic service to users. To fulfill elastic service with a dynamic workload, it is required to allocate the optimal number of resources (servers). Predictive models discussed earlier provide the estimation of requests that will be required to fulfill SLA contracted with users. As shown in Figure 2, such estimation of requests can be processed by a proposed Queuing model (M/M/c) to plan the resources. The purpose of using the Queuing model is to find a minimum number of servers(c) with system utilization. In Queueing model(M/M/c), parameters such as c, λ and μ are considered to find system utilization ρ using following

$$\rho = \frac{\lambda}{c\mu} \tag{3.19}$$

$$c = \frac{\lambda}{\rho\mu} \tag{3.20}$$

where c is number of servers , λ is arrival rate of user requests (real and predicted) and μ is considered as service rate of a server per time unit. In order to serve the users, system utilization μ should be less than 1. There are other parameters like W_p (queue time) and Q_w (response time) for modeling system performance. In this proposal we have considered time period of one hour to predict average system utilization and then based of proactive prediction model and using Queuing model to find the oscillation of resources to assign for future time period. One hour time period is considered as generalization scenario for user to obtain service from cloud providers. Following framework has been considered to achieve the task of resource management.

In this paper, we have considered a time period of one hour to predict average system utilization and then based on a proposed proactive prediction model. The Queuing model is proposed to find the oscillation of resources to assign for future periods. One hour time period is considered as a generalization scenario for users to obtain service from cloud providers. The following framework has been considered to achieve the task of resource management.

4. The predictive framework of a hybrid cloud. This section presents the proposed predictive model that employs an auto scale approach to improve the allocation of resources. The framework is depicted in Figure 4.1. Consider a hybrid cloud where private and public data centers have M machines and hosting A applications. Each application needs a certain amount of different resources. The system receives input from the data center's workload data and performs time series analysis to estimate the future workload and resources required to handle it. An autoscale module analyses such n time instances and forecasts the expected resources as a function of demand. Thus, the objective is to minimize the variance between forecast resources and demand resources to optimize resource allocation. The framework considers the queuing theory to plan estimated resources for the future workload. The execute process signals a resource manager to allocate or deallocate the resources. The Resource manager optimally associates the resources to a private cloud or a public cloud environment as per the availability.

The AutoScaleHybrid Model analyses user requests to forecast the number of resources in a hybrid cloud using a closed Queueing network. The closed Queueing network is considered where constant numbers of users will circulate in the system and are replaced by new users. The AutScaleHybrid algorithm (1) uses a historical load of user requests. It uses Predictor algorithm (2) to estimate the resources using ML techniques and Queueing methods. The algorithm will compare the allocated and forecast result and provide the current status of resources to scale up or down. Here resource managers will allocate the resources to private or public clouds based on decision parameters.

The time complexity of the proposed algorithm is $O(N)$, where N is the number of samples to be taken for analysis. So the system with linear time complexity is scalable and expanded with respect to time.

5. Experiments and analysis. The experiment was conducted on a dataset obtained from the university server log. The private cloud was set up to measure the utilization of the system with a web server. The unit of log data considered for the experiment was for an hour. Figure 4 shows the hourly average load of a web server of one month. The following setup had been configured (Table 5.1) for the experiment. The total number of hours is 6072. The data were extrapolated. As per algorithm 2 it is required to choose the lag period to identify

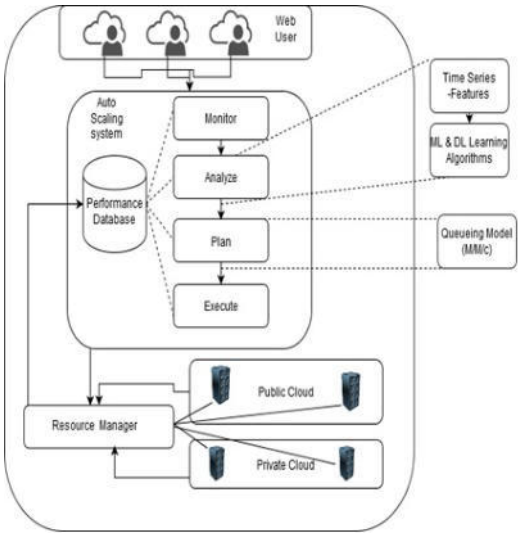


FIG. 4.1. *A Hybrid Cloud system with a predictive framework*

Algorithm 1 AutoScaleHybrid

1: **Initialization:** History of UserRequests (R_{req}) , Monitortime, UserRequest,status ,decision= $NULL$.
2: **repeat** for every monitortime t {
3: status =Predictor (History of Resources Usage)
4: If status =UP and UserRequest=Private then
PrivateScaleUp(new VMs)
5: Else If status =UP and UserRequest=Public then
PublicScaleUp(new VMs)
6: Else If status =DOWN and UserRequest=Private then
PrivateScaleDown(VMs)
7: Else
PublicScaleDown(VMs)
8: }

the footprint of seasonal data. The R programming tool is used to find autocorrelation of lag in time series data set. As can be observed from figure 5, data shows autocorrelation for different lag intervals. As per the data with lag (L)=24 gives the highest correlation value for seasonal data. So Input training data set can be prepared with the chosen lag value.

In order to apply ML (Machine Learning) based predictive models discussed earlier, it is required to set some basic parameters to obtain reasonable forecasting accuracy. In the SVM regression model, the parameter of loss function and the *"c"* parameter of regularized risk must be adjusted by the user. There are many directions given in the proposal for selection of parameters [34, 33]. Based on the literature, we have chosen the best value to build a good model. In a Bidirectional LSTM model, there are two recurrent layers side-by-side supplying the input data as it is and reverse copy of input data to the second. Here Relu activation function is used for hidden layers. In this work we have used adam optimizer which captures optimal learning rate with gradient. The R programming is configured with keras environment for this work. In this work, we suggest that the Bidirectional LSTM regression model outperforms other predictive methods. Here we have compared LSTM model with other statistical and non-statistical forecasting methods:

- Naive Method : $\hat{y}(t) = y(t-1)$ where $y(t-1)$ is the load of server of previous time interval and $y(t)$ is new estimated load.
- Auto-Regressive Method: Auto-regressive method is based on linear regression with lag interval of 24.

Algorithm 2 Predictor

Inputs: R_{req}; L.
Output:Status
Compute autocorrelation of data trace upto L
$L \leftarrow R_{req}$ /* Determine the lag with significant autocorrelation*/
sample $\leftarrow$ getsample(R_{req},L) /*Prepare the input data according to the lag L*/
$R_{fut} \leftarrow$ **Predict**(sample)
$R_{actual} \leftarrow Q_c(R_{req})$
$RR_{fut} \leftarrow Q_c(R_{fut})$
If $RR_{fut} > R_{actual}$ then status=UP
If $RR_{fut} < R_{actual}$ then status=Down
retrun status

TABLE 5.1
Server configuration

Model	HP DL380 10TH GENRATION
CPU	INTEL XEON SILVER 4110 (2 NOS.)
RAM	128 GB (32GB*4) DDR4-2666 MHZ
HTTP server	Apache 2.4
Guest OS	Ubuntu Server 14.04 LTS
Host OS	Centos 7.0

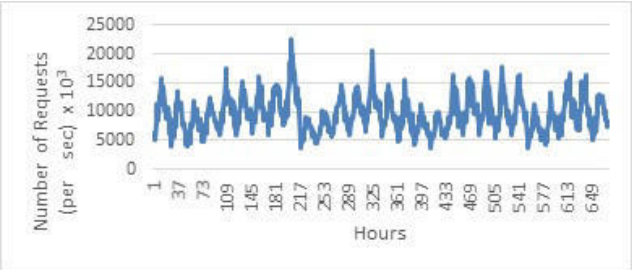


FIG. 5.1. *Historical workload of a web server*

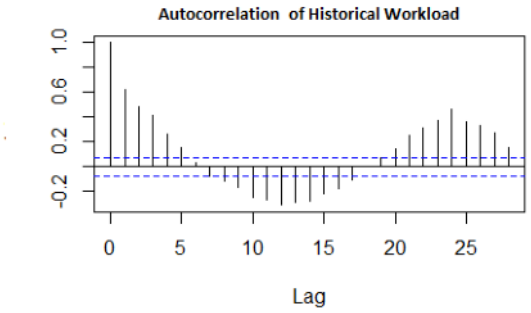


FIG. 5.2. *Autocorrelation of server workload*

TABLE 5.2
Parameters selection and Performance Measures Accuracy of Predictive Methods

Methods	Parameters and Values	MAE	RMSE
Naive	-	0.135	0.167
ARIMA	p=0,q=2,d=2	0.146	0.174
KNN	k=5	0.156	0.184
SVR (linear Kernel)	-	0.125	0.154
SVR(Poly. Kernel)	c=1.9,ϵ=0.027, p=2	0.113	0.138
Bi-LSTM (Proposed)	Batch size=64,learning rate =0.01 , layers=1	0.093	0.112

Here *autoarima* function of R programming environment has been considered. We have performed grid search to choose the parameter values (p=0,q=2,d=2) with AIC criteria.

- K-Nearest Neighbor: This is based on K - Nearest neighbors algorithm which forecast load of server based on similarity measure. Here in this work we have considered five neighbors for similarity measure and k-fold cross-validation to enhance the model.
- Support Vector Regression with Kernel Method: This method estimates the load of server with using equation 3.5 with degree of p=2. Here we have used grid search method to tune the hyper parameters (c and ϵ).
- Bidirectional LSTM Method: Here input layer that specifies length of input data with one feature. The two copies of hidden layer are available with *Relu* as activation function. Table 5.2 shows accuracy of different predictive methods with prediction error obtained by comparing with real values.

As seen from Table 5.2, SVR (Polynomial Kernel) and Bi-LSTM are considered as improved models with respect to prediction errors. The Bi-LSTM method provides better results as compared to other methods. As per the estimated load(requests) obtained from the above methods, we have implemented an M/M/c Queuing model to plan the number of servers that must be allocated with performance criteria. Based on the results of the Queuing model, we can plan the near-optimal resources allocated to the cloud. The resources (servers) which are over-provisioned if near-optimal allocation of resources are less and estimated resource utilization is more.

However, under-provisioned will occur when near-optimal allocation is more than the estimated resource allocation. In this work, we have considered Response Time as one of the performance criteria. The Response Time for user service is imposed by SLA. In the proposed Queuing model, service rate μ is considered as the throughput of the server (number of requests processed per unit of time). In an ideal scenario, a server having dynamic content will process 200 request/s [35]. The system utilization ρ must be less than 1 to provide efficient service to the user. In this work, the maximum response time which is 5ms is considered as per the SLA. Following the performance, criteria provide the effectiveness of the methods discussed above. Table 5.3 shows the estimation of resources using the methods discussed in Table 5.2.

- The number of allocated resources: The cost of infrastructure will increase when the estimated load is more than the optimal load. In such scenarios, the service rate will be improved. The best model will have an estimated load closer to the optimal load which leads to less number of over provisional resources.
- The number of SLA violations: The more number of SLA violations will occur when the number of allocated resources is less than the optimal. The best estimation model has less SLA violation which will solve the problem of under provision.

6. Results and Discussion. We have measured the performance of the proposed model and compared with other models. From the experiment results shown in tables 5.3 and table 5.4, we can see the elasticity of resources as per the estimated load compared to the server's real load. The dataset contains a log of user requests for one month. The model uses time lag data of 24-hour consecutive steps and forecast next day resources. The model's input vector is 3D and divided into training and testing sets for LSTM and 2D for other methods. During the training process of the proposed method, a mini-batch gradient descent method is used. The model optimizes the mean squared error (MSE) loss using adam optimizer and an early stopping mechanism

TABLE 5.3
Comparison of estimated resources using the Proposed and other Predictive methods with optimal resource allocation derived using Queuing Model.

Time (hour)	Optimal Load	Naive	ARIMA	KNN	SVR-L	SVR-P	Bi-LSTM
0:00	44	44	49	41	42	44	43
1:00	42	37	46	37	39	40	43
2:00	34	42	42	26	29	27	29
3:00	45	34	41	46	47	46	44
4:00	52	45	42	55	54	55	54
5:00	41	52	43	32	32	32	33
6:00	58	41	49	66	68	66	65
7:00	55	58	51	52	53	50	53
8:00	70	55	55	79	80	77	78
9:00	78	70	65	80	80	80	76
10:00	87	78	72	91	93	91	92
11:00	74	87	78	64	64	66	65
12:00	81	74	80	84	84	86	85
13:00	89	81	82	99	101	98	96
14:00	74	89	80	69	70	71	68
15:00	73	74	79	68	69	72	70
16:00	76	73	78	74	75	79	82
17:00	51	76	70	34	35	37	38
18:00	68	51	68	71	74	74	73
19:00	61	68	64	60	60	60	66
20:00	62	61	60	71	72	67	69
21:00	51	62	61	46	48	45	45
22:00	49	51	55	45	46	45	49
23:00	48	49	52	57	57	51	48

TABLE 5.4
Prediction of Resource status using Predictive Models with Queuing system

Resource status Estimation Method	Naive	ARIMA	KNN	SVR-L	SVR-P	Bi-LSTM
Under Provision	108	80	77	63	60	56
Over Provision	97	79	61	72	56	57
% SLA Violation	15	20	12	15	12	10

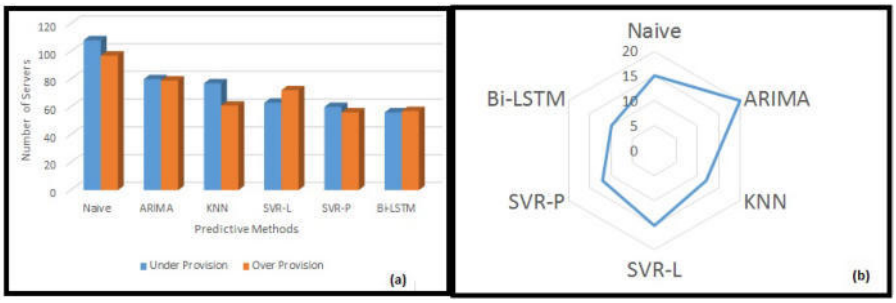


FIG. 5.3. *a. Forecasted resources (servers) suggested by Predictive Methods. b. Forecasted SLA violation (Percentage) of Predictive Methods*

is used to avoid over-fitting. The baseline model (Naive) is considered and compared with the proposed model. We compared the performance of the proposed model with other forecasting models with MAE and RMSE. As per Table 5.2, which suggests the proposed model performs well compared to other models. Also the results shown in Table 5.4 suggest the proposed model having less number of oscillations (5%)in predicted resources with respect to optimal load (Figure 5.3). Hence it suggests less number of SLA violations with other models, which imply that the model performs well in the future horizon.

7. Conclusions. In this paper, a Bi-directional LSTM based approach for resource allocation in a hybrid cloud environment is proposed. Further, the predictive methods based on ML and DL are evaluated using time series data. The Autoscale Hybrid model optimizes the SLA violation which leads to an effective response time of the service with optimal resource demand forecast. The auto scale module has very less time to execute and predict the resources. In the cloud environment, instances are forked which takes less than 10 minutes. The AutoscaleHybridModel framework considered data on an hourly basis in which instances can be configured in less time. The results show that the Bi-directional LSTM model provides closer to the real resource demand than the other models. In the future, the model with more optimization parameters can be incorporated to improve cloud resources usage predictions.

REFERENCES

[1] Masdari, Mohammad, et al., "Towards workflow scheduling in cloud computing: a comprehensive analysis." Journal of Network and Computer Applications 66 (2016): 64-82.

[2] Toosi, Adel Nadjaran, Richard O. Sinnott, and Rajkumar Buyya, "Resource provisioning for data-intensive applications with deadline constraints on hybrid clouds using Aneka." Future Generation Computer Systems 79 (2018): 765-775.

[3] Bresciani, Stefano, Alberto Ferraris, and Manlio Del Giudice, "The management of organizational ambidexterity through alliances in a new context of analysis: Internet of Things (IoT) smart city projects." Technological Forecasting and Social Change 136 (2018): 331-338.

[4] Xu, Xiangqiang, and Xinghui Zhao, "A framework for privacy-aware computing on hybrid clouds with mixed-sensitivity data." 2015 IEEE 17th International Conference on High Performance Computing and Communications, 2015 IEEE 7th International Symposium on Cyberspace Safety and Security, and 2015 IEEE 12th International Conference on Embedded Software and Systems. IEEE, 2015.

[5] Gandhi, Anshul, et al, "Optimal power allocation in server farms." ACM SIGMETRICS Performance Evaluation Review 37.1 (2009): 157-168.

[6] Ye K, Huang D, Jiang X, Chen H, Wu S, Virtual machine based energy-efficient data center architecture for cloud computing: A performance perspective In: Green Computing and Communications (GreenCom) 2010 IEEE/ACM Int'l Conference on Int'l Conference on Cyber, Physical and Social Computing (CPSCom), 171–178.

[7] Messias, Valter Rogério, et al., "Combining time series prediction models using genetic algorithm to autoscaling web applications hosted in the cloud infrastructure." Neural Computing and Applications 27.8 (2016): 2383-2406.

[8] Calheiros, Rodrigo N., et al., "Workload prediction using ARIMA model and its impact on cloud applications' QoS." IEEE Transactions on Cloud Computing 3.4 (2014): 449-458.

[9] Islam, Sadeka, et al., "Empirical prediction models for adaptive resource provisioning in the cloud." Future Generation Computer Systems 28.1 (2012): 155-162.

[10] Kumar, Jitendra, and Ashutosh Kumar Singh., "Workload prediction in cloud using artificial neural network and adaptive differential evolution." Future Generation Computer Systems 81 (2018): 41-52.

[11] Amiri, Maryam, and Leyli Mohammad-Khanli., "Survey on prediction models of applications for resources provisioning in cloud." Journal of Network and Computer Applications 82 (2017): 93-113.

[12] Manvi, Sunilkumar S., and Gopal Krishna Shyam., "Resource management for Infrastructure as a Service (IaaS) in cloud computing: A survey." Journal of network and computer applications 41 (2014): 424-440.

[13] Weingärtner, Rafael, Gabriel Beims Bräscher, and Carlos Becker Westphall., "Cloud resource management: A survey on forecasting and profiling models." Journal of Network and Computer Applications 47 (2015): 99-106.

[14] Panneerselvam, J., Liu, L., Antonopoulos, N., Bo, Y., Workload analysis for the scope of user demand prediction model evaluations in cloud environments. In: Proceedings of the 2014 IEEE ACM 7th International Conference on Utility and Cloud Computing, pp. 883–889. IEEE Computer Society (2014)

[15] Messias, Valter Rogério, et al., "Combining time series prediction models using genetic algorithm to autoscaling web applications hosted in the cloud infrastructure." Neural Computing and Applications 27.8 (2016): 2383-2406.

[16] Roy, Nilabja, Abhishek Dubey, and Aniruddha Gokhale, "Efficient autoscaling in the cloud using predictive models for workload forecasting." 2011 IEEE 4th International Conference on Cloud Computing. IEEE, 2011.

[17] Moren Vozmediano, Rafael, Ruben S. Montero, and Ignacio M. Llorente, "Elastic management of web server clusters on distributed virtual infrastructures." Concurrency and Computation: Practice and Experience 23.13 (2011): 1474-1490.

[18] Zhang, H., Jiang, G., Yoshihira, K., Chen, H., Proactive workload management in hybrid cloud computing. IEEE Trans. Netw. Serv. Manag. 11(1), 90–100 (2014)

[19] Hasan, Masum Z., et al., "Integrated and autonomic cloud resource scaling." 2012 IEEE network operations and management symposium. IEEE, 2012.
[20] Mason, Karl, et al., "Predicting host CPU utilization in the cloud using evolutionary neural networks." Future Generation Computer Systems 86 (2018): 162-173.
[21] Barrett, Enda, Enda Howley, and Jim Duggan, "Applying reinforcement learning towards automating resource allocation and application scalability in the cloud." Concurrency and Computation: Practice and Experience 25.12 (2013): 1656-1674.
[22] Tesauro, Gerald, et al., "A hybrid reinforcement learning approach to autonomic resource allocation." 2006 IEEE International Conference on Autonomic Computing. IEEE, 2006.
[23] Liu, Ning, et al., "A hierarchical framework of cloud resource allocation and power management using deep reinforcement learning." 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS). IEEE, 2017.
[24] Kumar, Jitendra, Rimsha Goomer, and Ashutosh Kumar Singh, "Long short term memory recurrent neural network (lstm-rnn) based workload forecasting model for cloud datacenters." Procedia Computer Science 125 (2018): 676-682.
[25] Zhang, Qingchen, et al., "An efficient deep learning model to predict cloud workload for industry informatics." IEEE transactions on industrial informatics 14.7 (2018): 3170-3178.
[26] Kaur, Pankaj Deep, and Inderveer Chana, "A resource elasticity framework for QoS-aware execution of cloud applications." Future Generation Computer Systems 37 (2014): 14-25.
[27] Hani, Ahmad Fadzil M., Irving Vitra Paputungan, and M. Fadzil Hassan, "Support vector regression for service level agreement violation prediction." 2013 International Conference on Computer, Control, Informatics and Its Applications (IC3INA). IEEE, 2013.
[28] Jiang, Yexi, et al., "Asap: A self-adaptive prediction system for instant cloud resource demand provisioning." 2011 IEEE 11th International Conference on Data Mining. IEEE, 2011.
[29] Allende H, Moraga C, Salas R, (2002) Artificial neural networks in time series forecasting: a comparative analysis. Kybernetika 38(6):685–707.
[30] Wang Y, Huang M, Zhao L, et al., (2016) Attention-based lstm for aspect-level sentiment classification In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, 606–615.
[31] Cherkassky, Vladimir, and Yunqian Ma., "Practical selection of SVM parameters and noise estimation for SVM regression." Neural networks 17.1 (2004): 113-126.
[32] Schuster, Mike, and Kuldip K. Paliwal., "Bidirectional recurrent neural networks." IEEE transactions on Signal Processing 45.11 (1997): 2673-2681.
[33] Boardman M, Trappenberg T., (2006) A heuristic for free parameter optimization with support vector machines In: The 2006 IEEE International Joint Conference on Neural Network Proceedings, 610–617.
[34] Cherkassky V, Ma Y, (2004) Practical selection of svm parameters and noise estimation for svm regression. Neural Netw. 17(1):113–126
[35] Iqbal W, Dailey M, Carrera D, Janecek P, (2011) Adaptive resource provisioning for read intensive multi-tier applications in the cloud. Future Gener. Comput. Syst. 27(6):871–879.

Edited by: Dana Petcu
Received: Sep 11, 2020
Accepted: Dec 13, 2020


Scalable Computing: Practice and Experience, ISSN 1895-1767, http://www.scpe.org
© 2020 SCPE. Volume 21, Issues 4, pp. 673–688, DOI 10.12694:/scpe.v21i4.1806

FORGERY PROTECTION OF ACADEMIC CERTIFICATES THROUGH INTEGRITY PRESERVATION AT SCALE USING ETHEREUM SMART CONTRACT

AUQIB HAMID LONE* AND ROOHIE NAAZ†

Abstract. Academic credentials are precious assets as they form an evidence for one's identity and eligibility. Fraud in issuance and verification of academic certificates have been a long-standing issue in academic community. Due to lack of anti-forgery mechanisms there has been substantial increase in fraudulent certificates. The need of the hour is to have a transparent and reliable model for issuing and verifying academic certificates to eliminate fraud in the process. Decentralized, Auditable and Tamper-proof properties of Blockchain makes it possibly the best choice for issuing and verifying academic certificates. In this paper we propose a model, where regulatory body authorizes higher education Institutes (universities and colleges) for issuing academic certificates to students in a decentralized way. Anyone in the world can verify the authenticity of the certificate by triggering appropriate smart contract functions, thus eliminating any possibility of fraud in the process. In addition we used multi signature scheme where certificates are required to be signed by designated authority from Higher Education Institutes, thus allowing for multi-level checks on certificate contents before being successfully deployed on Blockchain. We have also provide Proof of Concept in Ethereum Blockchain and evaluated its performance in terms of cost, security and scalability.

Key words: Certificate Integrity, Forgery Protection, Smart Contract, Ethereum Blockchain

AMS subject classifications. 68M14, 94A60

1. Introduction.

Higher Education has global impact and coverage and information regarding everyone's educational accomplishments should flow in a smooth and secured way. Academic credentials are used worldwide and are an important asset for both students and professionals pledging for jobs, scholarships or academic visibility. Traditional methods for recording, issuing and verifying academic credentials are expensive (due to intermediaries that charge fees for their services), inefficient (due to delays in executing agreements) and vulnerable (because it uses central system which can be compromised due to fraud, cyberattack, or a simple mistake which can effect the entire system). Need of the hour is to have secure, immutable and trustable academic credentials.

Blockchain in its simplicity is a series of connected tamper evident data structures called blocks, which contain or record everything that happens on some distributed systems on a peer-to-peer network . Each block is linked to and depends on previous block forming a chain, resulting in an append only system: a permanent and irreversible history that can be used as a real time audit trail by any participant to verify the accuracy of the records by simply reviewing data itself.

Blockchain was first developed for Bitcoin cryptocurrency and serves as distributed public ledger and transactions or events recorded on it are nearly impossible to tamper[16]. The driving force behind the interest in Blockchain research has been its key characteristics that provide security, anonymity and integrity without relying on trusted third party organisations. Initially Blockchain usage was restricted to cryptocurrencies only, since the advent of Ethereum: A next generation smart contract and decentralised application platform [8], applications beyond cryptocurrencies are being developed and explored.

Smart contract [20] is a piece of code which is stored in the Blockchain network (on each participant database). It defines the conditions on which all parties using contract agrees and certain actions described in the contract can be executed if the required conditions are met. As the smart contract is stored on every computer in the network, they all must execute it and get to the same result.

*Department of Computer Science and Engineering, NIT Srinagar, Jammu and Kashmir, India, 190006 (ahl@nitsri.net).
†Department of Computer Science and Engineering, NIT Srinagar, Jammu and Kashmir, India, 190006

The decentralised ledger functionality coupled with security provided by Asymmetric cryptography (Elliptic curve cryptography [15]) and distributed consensus algorithms (Proof of Work in case of Bitcoin and Ethereum [10]) of Blockchain, makes it a very attractive technology to solve the current financial as well as non-financial problems. Blockchain by design enforces integrity, transparency, authenticity, security and audit-ability thus making it possibly the best choice to make trust-less(distributed trust) systems to solve or improve traditional means for recording, issuing and verifying academic credentials. The goal of Blockchain is to provide anonymity, security and transparency to all its users.

1.1. Challenges and Limitations with Traditional methods for issuing and verifying academic certificates . One of the key challenges in traditional means of certificate verification is to deal with fake certificates. Traditional methods do not guarantee authenticity, security, tamper-resistance of academic records. Major limitations in traditional methods for issuing and verifying academic certificates are enumerated below:

- *Cost*: Traditional methods of certificate verification are costly as verification agencies charge fee for each certificate to be verified.
- *Time* : A great amount of time is lost in existing methods for verifying certificates as it depends on location and the response time of the issuing authority.
- *Availability*: Physical documents are susceptible to loss or damage. In case of loss or damage, individuals cannot readily avail duplicates of the certificates with existing process.
- *Third party dependency*: In traditional methods, organizations depend on third party verification agencies to verify the authenticity of the certificates with the issuing authorities.

Rest of the paper is organized as follows: Section 2 presents a brief overview of previous attempts for improving issuance and verification of academic certificates. Section 3 presents the description of proposed model with PoC in Ethereum Blockchain. Section 4 provides evaluation and analysis of proposed model. Finally, Section 5 concludes the paper and references are listed in the end. We also provide smart contract code for proposed model as an Appendix.

2. Related Work. Collecting literature to have a longitudinal and representative view of Blockchain applicability in certificate issuance and verification is challenging because of its rich diversity of Blockchain applications. In order to have clear-cut, unbiased, complete and broader perspective many sources have been explored including major online databases. One of the factors behind exploring major databases is their rich library of journals with high impact factors.

The University of Nicosia is the first university in the world to issue academic certificates whose authenticity can be verified through the Bitcoin blockchain. A Certificate granted to a student is issued as a PDF document. The Hash of this certificate is computed and registered in the Bitcoin blockchain as a transaction [1]. Another outstanding attempt in this direction is Blockcerts: an open standard for creating, issuing, viewing and verifying blockchain-based certificates. These digital records are registered on a blockchain, cryptographically signed, tamper-proof, and shareable. Blockcerts allows for batch issuance of certificates using Merkle trees to optimize storage in Blockchain [9]. Both [1] and [9] exploit Bitcoin's OP_RETURN feature to record certificate hashes. Blockcerts comes with one major shortcoming as it stores revocation list on centralized server, which could be exploited by the attackers to comprise the whole process. To overcome this author in [18] proposed Hypercerts, a decentralized mechanism for credential revocation, by combining the capabilities of Ethereum smart contracts and InterPlanetary File System (IPFS). The work presented in [5] proposed a Blockchain based scheme that guarantees integrity, proof of existence and also allowed to assess trust in students' data. This data is gathered in a so-called ePortoflio and includes completed assignments, course transcripts and granted certificates. It is stored in a peer-to-peer system called Interplanetary File System. Hashes of academic data are registered in Ethereum blockchain. Authors in [19] proposed a blockchain solution that allows users to assess the trust in academic data in terms of the reputation of the creators of such data. The assumption is that individuals trust a piece of data to be genuine if it has been issued by a reputable person or entity. To measure reputation, a currency called kudos is proposed.

Authors in [14] proposed a model of confidence in open and ubiquitous higher education, based on Blockchain technology. Proposed model is used to certify the acquisition of competencies by student trained in different educational institutions and is based on a consensus protocol of experts who are part of the system itself. Authors in [13] gave a different concept of using hybrid Blockchain comprising of 2 Blockchains. One private

TABLE 2.1
Summary of Related Work

Scheme	***Blockchain Technology***	***Features***					
		Multi-Sig	**Regulatory Authority Accredetion**	**Transparency**	**Privacy**	**Issuing Authority & Certificate Revocation**	**Accessibility**
[11]	Bitcoin	No	No	Yes	Yes	No	Yes
[10]	Bitcoin	No	No	Yes	Yes	Partial	Yes
[12]	Ethereum & IPFS	No	No	Yes	No	Partial	Yes
[17]	Ark	Yes	Yes	Partial	Yes	No	Partial
[18]	Ethereum	No	Yes	Yes	No	Yes	Yes

Blockchain for storing individual records and one public Blockchain for storing authentication information of private Blockchain in order to prevent tempering. Authors in [12] proposed Blockchain based platform for issuing, validating and sharing of Educational certificates. Authors in [21] proposed a blockchain-based higher education credit platform (EduCTX) with proof of concept in Ark Blockchain Platform, for creating a globally trusted higher education credit and grading system. Authors in [11] proposed a theoretical model for graduation Certificate verification using Blockchain Technology. The work presented by the authors in [17] proposed a Blockchain and smart contracts based scheme for higher education registry in Brazil. The authors in particular aims at digitization of degree certificates and academic credits for higher education in the Brazilian education system. Authors in [7] provided a detailed review on applicability of Blockchain in Education. In essence they discussed challenges and benefits of applying Blockchain in Education sector.

Proposed model provides numerous benefits more specifically to Higher Education Industry and in general to other industries also. Proposed model is generic in nature and can be applied to protect other digital documents also. Multi Signature scheme in proposed model allows for multi-level checks of certificate before being deployed to Blockchain. The potential benefits that proposed model could bring to Higher Education Industry are briefly summarized below:

- Certificate credentials published to Blockchain are immutable, trustful and verifiable.
- Simplifies the workflow of certificate issuance and verification, thus making whole process efficient and economical.
- Has the potential to transform education industry, by making processes involved more efficient, transparent, democratic and secure.
- Can be extended to be used for authentication and verification of other official statements or files also.

Proposed model is beneficial for Issuing Authority as it helps them to issue cryptographically-secure records that cannot be forged, records that are secure and auditable. Participants who have been issued degrees through proposed model will get benefited as they now own and can share cryptographically secure records, with instant access to academic accomplishments as Blockchain highly available in nature and instant verification as Blockchain eliminates dependency on issuing authority to verify records.

3. Proposed Model. In this paper we propose a model for forgery protection of academic certificates through integrity preservation using Ethereum Blockchain in Higher Education. Simplified architecture of proposed model comprises of three parts as shown in figure 3.1. First part is about authorization request and response: wherein Higher Education Institutes request Higher Education Regulatory Authority for issuing certificates by presenting appropriate approval certificates and a list of delegated signature authorities. In response to Higher Education Institutes request, Regulatory Authority after proper verification, authorizes the institutes for issuing and revoking certificates to and from students. Part second of the proposed model is about certificate signing, issuance and revocation of certificates: wherein first delegated signature authorities of authorized Higher Education Institutes sign the certificates and then Institutes issue certificates to students. Under certain rare situations certificates are revoked from students by authorized institutes. Third and last part is about verification: wherein verifiers verify the authenticity of the certificates. Verification result is true if integrity test passes i.e. if details present on physical copy of certificate matches with the one present on the Blockchain else it returns false. Communication takes place via Blockchain network. For already issued certificates that are not deployed on Blockchain we have provided an export functionality in proposed model which helps Higher Education Institutes to put them on Blockchain.

Architecture of proposed model comprises of five main components namely Participants, Front-End, Core

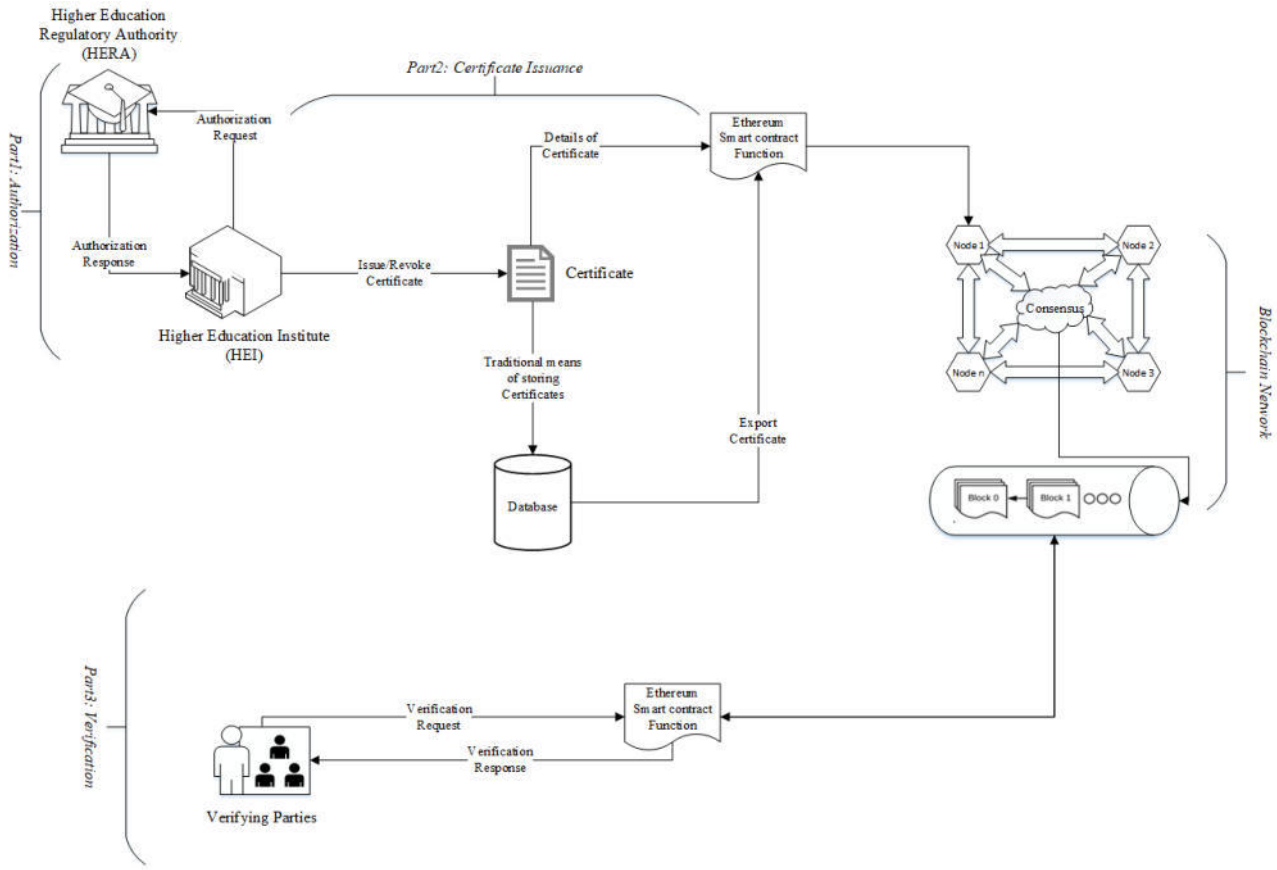


FIG. 3.1. *Simplified Architecture of Proposed Model*

Modules, Blockchain Network and Distributed Storage as shown in figure 3.2.

3.1. Participants. They are the real actors in any network. Participants mainly represent business but have the potential of representing people, regulators or other stakeholders. In proposed model participants include Higher Education Regulatory Authority, Higher Education Institutes and Verifiers. Regulatory authority and Institutes act as full nodes, storing entire copy of the Blockchain, while as Verifiers need not to store any Blockchain data. Verifiers verify claims made by any participant by fetching proof from Blockchain via front-end and appropriate core module of the proposed model. Every action performed by participants gets recorded on Blockchain via appropriate communication link. Thus in proposed model participants can be made liable for their actions on Blockchain network.

3.2. Front-End. Proposed model front-end is developed with the help of HTML5 and CSS. Appropriate JavaScript libraries (Web3.js [6]) have been used to connect front end with Blockchain network. It is beneficial to perform certain tasks at client side itself rather than computing them on Blockchain, because every computational step has a cost in Blockchain to be paid by participant who initiated the task. Certain validation checks can also be enforced at client side itself.

3.3. Core Modules. They are heart and soul of the proposed model and facilitate the communication of participants with Blockchain Network. Participants store and retrieve the certificate details to/from the Blockchain by calling an appropriate core module. Core modules are basically Ethereum smart contract functions. Proposed model Smart contract is written in Solidity language [4], compiled and tested both on Remix IDE [3] and Ganache(testrpc) [2] private network. We first defined the academic certificate and higher education

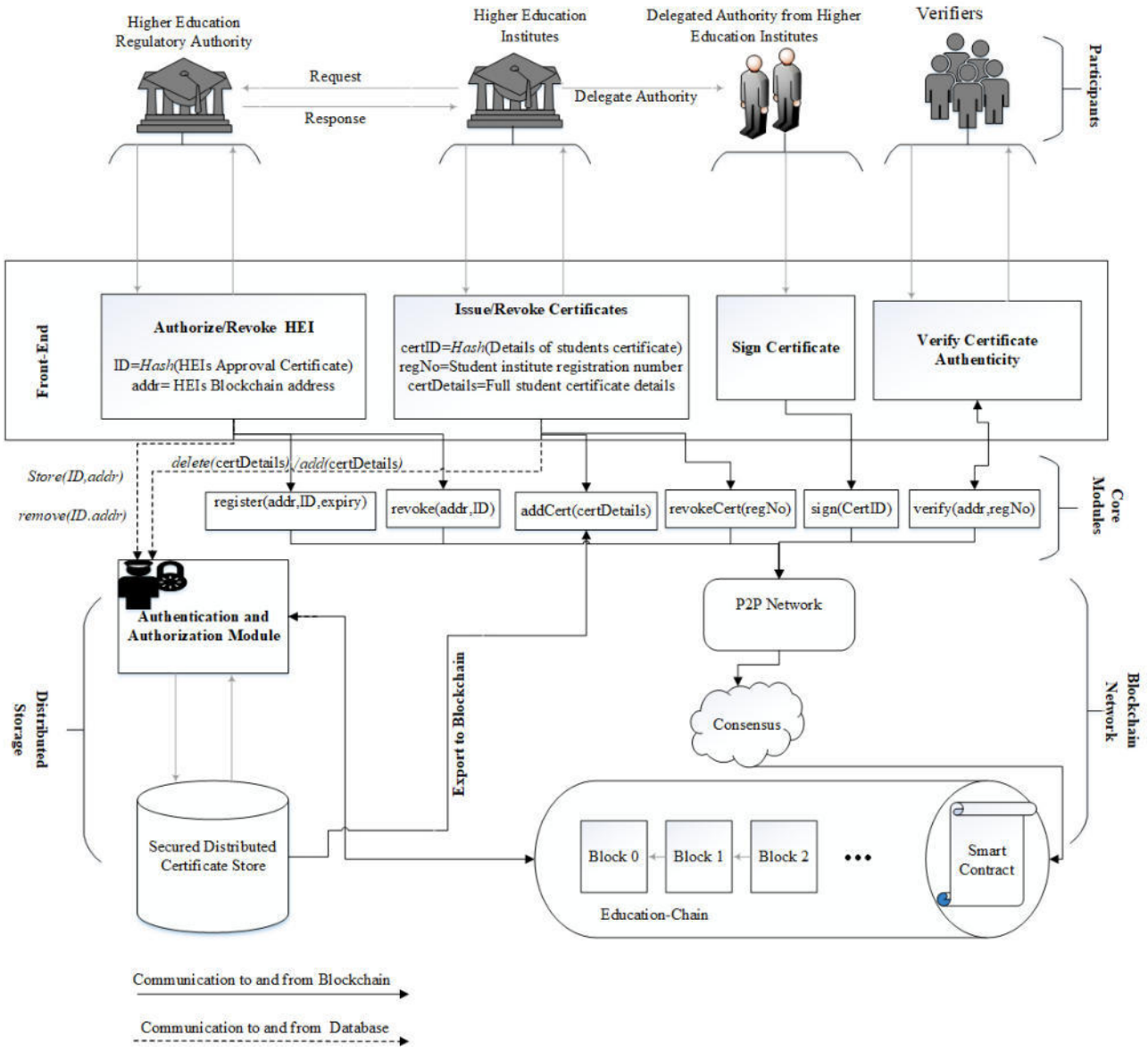


FIG. 3.2. *Operational Flow of Proposed Model*

institutes approval certificate as a solidity structures comprising of following information:

Struct *approvalCert* **contains**
- bytes32 *hash*;
- bool *registered*;
- uint *expiry*;
- bool *revoked*;
- address[] *signAuth*;
- `// other optional stuff`

- *hash*: Uniquely identifies the contents of Higher Education Institutes approval certificate issued by Higher Education Regulatory Authority . Hash is obtained by taking the SHA256 of contents of the approval certificate and other related information.
- *registered*: It comprises of boolean value, indicating whether Higher Education Institute is authorized by Higher Education Regulatory Authority.
- *expiry*: It is a unix timestamp indicating validity of approval certificate.
- *revoked*: It comprises of boolean value, indicating whether Higher Education Institutes approval certificate is expired. If revoked is true Higher Education Institute can no longer issue certificates.
- *signAuth*: An array of type address, containing list of Ethereum account addresses from Higher education Institute authorized for signing the certificate .

```
Struct studentCert contains
    bytes32 certHash;
    string regNumber;
    string instName;
    string stuName;
    string stuGrade;
    string stuDegree;
    address[] signedBy
    // other optional stuff
```

- *certHash*: Uniquely identifies the contents of students academic certificate issued by Higher Education Institute . Hash is obtained by taking the SHA256 of contents of the certificate and other related information.
- *regNumber*: regNumber is registration number assigned to the student by Higher Education Institute. It uniquely identifies the credentials of student.
- *instName*: Full name of the Higher Education Institute which issued the certificate.
- *stuName*: Full name of the student to whom certificate was issued.
- *stuGrade*: This represents academic credits that student obtained.
- *stuDegree*: Name of the course that student completes.
- *signedBy*: Array of Ethereum account addresses who signed the certificate.

In order to ensure the integrity of smart contract execution on participant actions, core modules (smart contract functions) are restricted based on the role. All functionalities of the smart contract were effectively screened for role limitation. Role restrictions are achieved with the help of solidity modifiers. Following modifiers are used in the proposed model for role restrictions.

1. *onlyOwner*: This modifier allows Higher Education Regulatory Authority which deploys smart contract to grant/revoke authorization to Higher Education Institutes. No other participant can authorize/revoke Higher Education Institutes, as owner is set in smart contract constructor itself. Hence, if any participant other than Higher Education Regulatory Authority tries to authorize any other participant, the transaction reverts as *onlyOwner* modifier returns false.
2. *onlyAuth*: This modifier allows only authorized Higher Education Institutes to issue certificates. Hence, if any participant other than authorized Higher Education Institute tries to issue certificates, the transaction reverts as *onlyAUth* modifier returns false.
3. *ifExists*: This modifier allows *addCert()* function to execute only if certificate hasn't been issued earlier.
4. *isSignedByAll*: This modifier allows *addCert()* function to execute only if certificate has been signed by delegated authorities.
5. *notSet*: This modifier allows *sign()* function to execute only if certificate hasn't been already signed by the delegated authority.
6. *onlyIncharge*: This modifier allows *sign()* function to execute only if certificate is being signed by authorized delegate authority, else transaction reverts.

Core modules in proposed model perform four basic functions 1) Higher Education Regulatory Authority

authorizing or removing Higher Education Institutes for issuing academic certificates. 2) Higher Education Institutes issuing or revoking certificates to or from students. 3) Authorized signers from Higher Education Institute signing the certificate 4) Verification about the authenticity of certificates. Core modules (Ethereum smart contract functions) are triggered by the participants via front-end in the network. The constraints like who should access what function and under what conditions access should be granted to participants, are all enforced through solidity modifiers defined earlier. Pseudocode of the functions is presented below in the form of algorithms.

Authorize Higher Education Institute. No Higher Education Institute can issue or revoke certificates unless they are authorized by Higher Education Regulatory Authority (responsible for deploying smart contract). *register()* smart contract function takes Higher Education Institutes Ethereum account address, hash of Higher Education Institutes approval certificate (issued by Higher Education Regulatory Authority), expiry date of approval certificate and array of Ethereum account addresses (whose signatures are required for the execution of *addCert()* transaction) as input. On successful execution of the *register()* function, Higher Education Institute gets registered under Higher Education Regulatory Authority. Appropriate modifier is used so that *register()* smart contract function can only be executed Higher Education Regulatory Authority. *register()* smart contract function is briefly summarized in Algorithm 10.

Algorithm 1: ***register()***

Input: *address* institution, *bytes32* certHash, *uint* notValidAfter, address[] signee
Result: Registers the Higher Education Institute under Higher Education Regulatory Authority for certificate issuance and revocation

```
if onlyOwner then
    if Higher Education Institute aleardy authorized then
        revert
    else
        1. Set the hash of approvalCert struct corresponding to Higher Education Institutes
           address(institution) with certHash
        2. Set the expiry of approvalCert struct corresponding to Higher Education
           Institute address(institution) with notValidAfter
        3. Set the revoked of approvalCert corresponding to Higher Education Institutes
           address(institution) with false
        4. Set the registered of approvalCert struct corresponding to Higher Education
           Institutes address(institution) with true
        5. Push the signAuth of approvalCert struct corresponding to Higher Education
           Institutes address(institution) with signee
        6. Emit Registered event
else
    revert
```

Remove Higher Education Institute. revoke() smart contract function takes Higher Education Institutes Ethereum account address as input. On successful execution of *revoke()* function Higher Education Institute gets removed from Higher Education Regulatory Authority approved institutes list and no longer can issue or revoke certificate to or from students. Appropriate modifier is used so that *revoke()* smart contract function can only be executed Higher Education Regulatory Authority. The removed Higher Education Institutes address (institution) is pushed to revoked array, to prevent them from issuing certificates in future. *revoke()* smart contract function is briefly summarized in Algorithm 11.

Add Certificate. Add Certificate function takes certificate details as input and associates them with registration number of the student and Ethereum account of Higher Education Institute. Only authorized Higher Education Institutes can execute *addCert()* function. Furthermore all designated signers from Higher Education Institute must sign on the certificate for the *addCert()* function to get successfully executed. *addCert()* smart contract function is briefly summarized in Algorithm 12.

Algorithm 2: ***revoke()***

Input: *address* institution
Result: Removes Higher Education Institute under Higher Education Regulatory Authoritys authorized list
if *onlyOwner* **then**
  if *Higher Education Institute aleardy authorized* **then**
    1. Set the revoked of *approvalCert* corresponding to Higher Education Institutes address(institution) with true
    2. Push institution address to Higher Education Regulatory Authoritys revoked array
    3. Emit Revoked event
  else
    revert
else
  revert

Algorithm 3: ***addCert()***

Input: *bytes32* hash, *string* regno, *string* instname,*string* name,*string* grade, *string* degree
if *authorized && isSignedByAll* **then**
  if *certificate aleardy exists* **then**
    revert
  else
    1. Set the hash of *studentCert* struct corresponding to students regno(registration number) and Institute account address with certHash
    2. Set the regNumber of *studentCert* struct corresponding to students regno(registration number) and Institute account address with regno
    3. Set the instName of *studentCert* corresponding to students regno(registration number) and Institute account address with instname
    4. Set the stuName of *studentCert* struct corresponding to students regno(registration number) and Institute account address with name
    5. Set the stuGrade of *studentCert* struct corresponding to students regno(registration number) and Institute account address with grade
    6. Set the stuDegree of *studentCert* struct corresponding to students regno(registration number) and Institute account address with degree
    7. Emit CertAdded event
else
  revert

Sign Certificate. sign() function takes student registration number, certificate hash and Institute address as input and on successful execution signs the certificate. Only authorized signature authority from concerned Higher Education Institute are allowed to sign the certificate and that too only once. Delegated signature authority approves certificate by signing on the hash of certificate contents corresponding institute address and student registration number. *sign()* smart contract function is briefly summarized in Algorithm 13.

Revoke Certificate. revokeCert() function takes student registration number as input and removes the certificate details from Blockchain by deleting corresponding entry from smart contract mapping data structure. The only check this function does is to ensure function is executed only by authorized Higher Education Institute and certificate already exists. Successful execution of this function results in negative Gas, given as an incentive for freeing some storage space on Blockchain. *revokeCert()* smart contract function is briefly summarized in Algorithm 14.

Verify Certificate. verify() smart contract function takes registration number and Hash printed on certificate as input and on successful verification returns the certificate view from Blockchain else verification fails. *verify()* function does not modify the state of the Blockchain, it only displays information retrieved from Blockchain. *verify()* smart contract function is briefly summarized in Algorithm 15.

Algorithm 4: ***sign()***

Input: Higher Eduaction Institute address, Hash of Certificate, Registration Number of Student
Output: Signs the certificate
if *onlyAuthorizedSigner && notSignedEarlier* **then**
    1. Sign the certificate
    2. Push the signer address to studentCert array *signedBy*
    3. Emit signed event
else
    revert

Algorithm 5: ***revokeCert()***

Input: *string* regno
if *authorized* **then**
    if *certificate aleardy exists* **then**
        1. Delete the certificate details of *studentCert* struct corresponding to students regno(registration number) and Institute account address
        2. Emit RevokedCert event
    else
        revert
else
    revert

3.4. Blockchain Network. It comprises of Peer-to-Peer (P2P) network and Consensus protocol that governs the communication over P2P network. In proposed model Blockchain network is private, where regulatory authority and authorized institutes can only act as validators or miners validating transactions, periodically collecting and creating blocks in the network. Higher Education Regulatory Authority is responsible for configuring, operating and maintaining Blockchain network. Regulatory authority also manages how other participants access and use the network.

3.5. Distributed Certificate Store. It comprises of distributed storage with authorization and authentication module for safely storing and preserving the original certificate details. This represents the traditional method for storing and preserving certificate records. But in proposed model it is optional for Authorized Higher Education Institutes to store newly issued certificates in distributed certificate store, however they can use export functionality of the proposed model to export already issued certificates from certificate store for deploying them to Blockchain.

4. Experimental Evaluation and Analysis. This section presents the details of Experimental Evaluation and Analysis of proposed model. For experimentation we used Remix [3] IDE in-browser developing and testing environment connected to private Ethereum Blockchain. We also used appropriate JavaScript code snippet for catching events which get triggered on execution of various smart contract functions for analysis purpose.

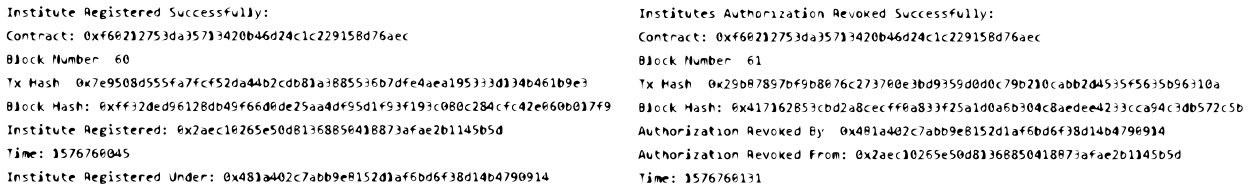
We first performed testing to validate and verify three key scenarios in proposed model 1) Authorization/Revocation by Higher Education Regulatory Authority 2) Certificate signing and issuance/revocation by authorized Higher Education Institutes and 3) Verification by any participant for certificate authenticity by analysing outputs and logs of corresponding emitted smart contract events.

Authorization/Revocation by Higher Education Regulatory Authority. Higher Education Regulatory Authority uses *register()* smart contract function to register and authorize Higher Education Institutes for issuing certificates to students. Any participant calling *register* other than Higher Education Regulatory Authority which deployed smart contract leads to execution failure. Revocation of Higher Education Institute from authorized list is done by calling *revoke* smart contract function by Higher Education Regulatory Authority. The Results of successful authorization and revocation of Higher Education Institute by Higher Education

Algorithm 6: ***verify()***

Input: Student Registration Number, Higher Eduaction Institute address, Hash of Certificate
Output: Displays the appropriate Certificate instance from EDUChain
if *Registration Number exists && input Hash == Hash from Blockchain* **then**
1. Verification Successfull
2. Return the Certificate view from Blockchain
3. Emit verified event

else
return verification unsuccessfull



(a) Successful Authorization (b) Successful Revocation

FIG. 4.1. *Authorization and Revocation of Higher Education Institute by Higher Education Regulatory Authority*

Regulatory Authority is shown in Figure 4.1.

Certificate Issuance and Revocation by authorized Higher Education Institutes to students. Only after authorization from Higher Education Regulatory Authority, Higher Education Institutes can issue or revoke certificates to/from students via Blockchain. Higher Education Institutes issue certificates by calling *addCert()* smart contract function and revoke by calling *revokeCert()*. Designated signature authority sign Certificates using *sign()* smart contract function to approve them for being deployed to Blockchain. The successful signing of certificate hash by delegated signature authority are shown in figures 4.2.a and 4.2.b. Figures 4.2.c and 4.2.d shows successful certificate issuance and revocation by authorized Higher Education Institute respectively.

Verification by any participant for certificate authenticity. Participants can verify the authenticity and integrity of certificate by calling *verify()* smart contract function with appropriate parameters. Figure 4.3 shows decoded output of successful execution of *verify()* function.

After successfully validating and verifying different functionalities of the proposed model we then analysed the feasibility of the proposed solution in terms of cost, security and scalability.

4.1. Cost Analysis. Every transaction executed on Ethereum Blockchain costs some Gas(unit of cost for a particular operation). In Ethereum Gas is paid in terms of Ether, as it is crypto fuel for running applications on the Blockchain network. There are transaction and execution gas costs for each function performed on the blockchain network. Execution cost includes the cost of internal storage in the smart contracts as well cost associated with any manipulation of Blockchain state. Transaction cost includes execution cost and the cost related to other factors like contract deployment and sending data to Blockchain network.

Table 4.1 shows the gas costs of smart contract functions of the proposed model. Smart contract functions are executed by the participants of the proposed model. *verify()* function costs least because it does not involve any updation in the Blockchain state, while as *addCert()* function costs the most because it considerably changes the state of the variables stored on Blockchain. *constructor()* function is a special function as it is related to the deployment of smart contract and is executed once in the life-cycle of the proposed model.

4.2. Security Analysis. In this section we present brief security analysis on how our proposed solution ensure key security goals such as integrity, non-repudiation, authorization, availability and accountability.

1. Integrity: Proposed model ensures integrity by storing traceability provenance data in immutable Blockchain infrastructure. Cryptographic hash functions make Blockchain immutable in nature.
2. Non-Repudiation: Every action is recorded in tamper-proof logs in proposed model and all actions

(a) Successful Signature 1

(b) Successful Signature 2

(c) Successfully Issued Certificate

(d) Successful Revocation

FIG. 4.2. *Signing and Issuance/Revocation of Certificates by Higher Education Institute*

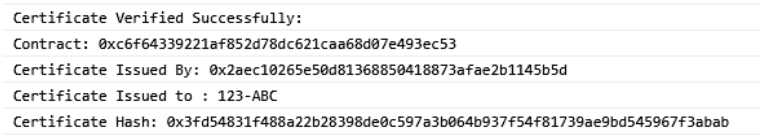


FIG. 4.3. *Successful Certificate Verification*

are linked and cryptographically signed by the initiator. No participant can deny their actions as everything is saved in the tamper-proof logs.

3. Authorization: In proposed model role restrictions have been enforced by using solidity modifiers to ensure proper authorization checks before executing any smart contract function.
4. Availability: Once certificate is deployed on Blockchain, it immediately becomes available to verifiers. The information stored on the Blockchain is saved in distributed and decentralized fashion and thus is immune to single point of failure.
5. Accountability: Since Ethereum account address of Higher Education Institutes is linked with approval certificate, thus Higher Education Institutes can be made accountable for their actions on Blockchain.

4.3. Scalability Analysis. Since Blockchain is append-only database, thus its size only increases with time. Size of Blockchain at any instant of time t is the sum total of the size of all blocks present in the blockchain at time t. Size of the single block is the sum total of the size of all transactions in it and size of the block header H_S which is a constant. Scalability analysis of the proposed model is based on the following assumptions:

- Chain configuration parameters are same as that of Ethereum mainnet
- Blocks are being generated at a constant rate of 15s i.e. $T = 15$ seconds
- Header size is same as that of Ethereum mainnet

Blockchain size heavily depends on the workload of the system, which in turn depends on many different factors like block gas limit G (maximum amount of gas that transactions in a block can consume) and block time period T (represents the rate at which new blocks are created in the Blockchain network.). if $I_t(txn)$ is set of transactions included in the Blockchain at time t, H_s is the header size of the single block, then size of

TABLE 4.1
Cost of Smart Contract Functions in proposed model

Function Caller	**Function Name**	**Gas Used**	**Transaction size in Bytes**
Higher Education Regulatory Authority	*constructor*	2963591	10948
Higher Education Regulatory Authority	*register()*	152015	447
Higher Education Regulatory Authority	*revoke()*	86412	255
Higher Education Institute	*addCert()*	182043	735
Higher Education Institute	*removeCert()*	32870	351
Signature Authority	*sign()*	87708	383

the Blockchain $B_s(t)$ at time t can be approximated by the following equation:

$$B_s(t) = \frac{t}{T} * H_s + \sum_{txn \in I_t(txn)} S(txn) \tag{4.1}$$

where *S(txn)* is size of the transaction. Since block creation rate is constant, with new block being generated after every T second, thus equation (4.1) can be rewritten as

$$B_s(t) = N * H_s + \sum_{txn \in I_t(txn)} S(txn) \tag{4.2}$$

where N is number of Blocks in the Blockchain at time t. The factor $N * H_s$ is the total overhead due to block headers in the Blockchain at time t, thus equation (4.2) can rewritten as:

$$B_s(t) = H_o + \sum_{txn \in I_t(txn)} S(txn) \tag{4.3}$$

where H_o represents total overhead due to headers in the Blockchain at time t. The first term in equation (4.3) is dependent on block period T and second term is variable in nature depending on the time, block gas limit G and workload of the system. The growth of Blockchain size over a time interval $[t_1{:}t_2]$ can be approximated by the equation below:

$$B_s(t_1 \rightarrow t_2) = H_o^{t_1 \rightarrow t_2} + \sum_{txn \in I_{t_1 \rightarrow t_2}(txn)} S(txn) \tag{4.4}$$

where $H_o^{t_1 \rightarrow t_2}$ is the total overhead due to block headers recorded in the Blockchain from time period t_1 to t_2 and $I_{t_1 \rightarrow t_2}(txn)$ is set of transactions recorded in the Blockchain from time period t_1 to t_2. Since we were not able to find any publicly available statistics about number of approval certificates given by Higher education regulatory authority to Higher education institutes and number of degree certificates issued by approved Higher education institutes to students. We considered synthetic workloads with n number of institutes authorized for issuing certificates and each institute issuing $10n$ certificates per year. Thus equation (4.4) reduces to:

$$B_s(t_1 \rightarrow t_2) = H_o^{t_1 \rightarrow t_2} + n + 10n^2 + 10n^2 \tag{4.5}$$

2^{nd} term in equation (4.5) represents total register() transactions, 3^{rd} term represents total *issueCert()* transactions and 4^{th} term represents total *sign()* transactions, as at least one signature from signature authority of authorized higher education institute is required for certificate to be broadcasted on Blockchain. We used equation (4.5) for computing the annual growth in Blockchain size with different classes of workloads. Results are presented in the Table 4.2. The second column of Table 4.2 contains the values for Blockchain growth rate per year with block period equal to 15 seconds, third column of Table 4.2 includes the growth rate without considering block headers overhead and fourth column contains overhead percentages. Results have shown even

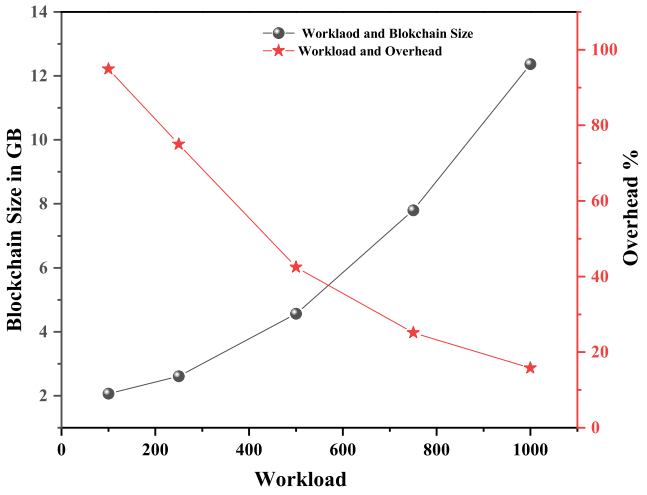


FIG. 4.4. *Blockchain Size/Year and Block Header Overhead % with different classes of Workload*

TABLE 4.2
Growth in Blockchain size with different values for n

Workload	$B_s(t)$ ***with*** $T = 15s$	$B_s(t)$ **-** H_o	$H_o/B_s(t)$ **%**
$n = 100$	2.062 GB/Year	0.104 GB/Year	94.9
$n = 250$	2.608 GB/Year	0.650 GB/Year	75
$n = 500$	4.561 GB/Year	2.603 GB/Year	42.9
$n = 750$	7.799 GB/Year	5.841 GB/Year	25.1
$n = 1000$	12.370 GB/Year	10.412 GB/Year	15.8

in case when 1000 institutes are authorized by regulatory body and institutes issues total of 10 million certificates per year, the growth rate is around 12.3 GB per year, which is acceptable given the storage capacities of modern-day high-end devices.

It is evident from the Figure 4.4, with increase in system workload Blockchain size increases, however overhead percentage decreases, thus proposed model scaling is limited by the workload of the system. In case of higher workload, Regulatory Authority can allow Institutes to issue certificates in batches, where a single transaction is done for issuing certificates to a batch of students thus reducing workload on the system. The concept of issuing certificates in batches was first implemented in Blockcerts [9].

5. Conclusions and Future Work. In this paper, we have proposed Blockchain based model for Academic certificate issuance and verification in Higher Education. Our proposed model architecture, algorithm, testing and implementation details are generic enough and can be applied to issue and verify other kind of certificates apart from academic ones. We provided the prototype of proposed model based on Ethereum Blockchain and evaluated its performance in terms of cost, security and scalability. Results confirm the feasibility of the proposed model. The future work will aim at developing complete optimized, privacy preserving end to end integrated framework for issuing and verifying academic certificates backed by distributed and decentralized storage with detailed case study on Indian Higher Education.

REFERENCES

[1] *Blockchain certificates (academic and others).*

[2] *Ganache a personal blockchain for ethereum development.* `https://www.trufflesuite.com/docs/ganache/overview`. Accessed: 19-12-2019.
[3] *Remix ide for ethereum smart contract programming.* `https://remix.ethereum.org/`. Accessed: 19-12-2019.
[4] *Solidity a high-level language for implementing smart contracts.* `https://solidity.readthedocs.io/en/develop/`. Accessed: 01-08-2019.
[5] *University to open. open blockchain.*
[6] *Web3 javascript api to interact with ethereum nodes.*
[7] A. Alammary, S. Alhazmi, M. Almasri, and S. Gillani, *Blockchain-based applications in education: A systematic review*, Applied Sciences, 9 (2019), p. 2400.
[8] V. Buterin, *Ethereum: A next-generation smart contract and decentralized application platform, 2013*, URL {http://ethereum. org/ethereum. html}, (2017).
[9] E. Durant, A. Trachy, and . O. of Undergraduate Education, *Digital diploma debuts at mit*, Oct 2017.
[10] A. Gervais, G. O. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, and S. Capkun, *On the security and performance of proof of work blockchains*, in Proceedings of the 2016 ACM SIGSAC conference on computer and communications security, ACM, 2016, pp. 3–16.
[11] O. Ghazali and O. S. Saleh, *A graduation certificate verification model via utilization of the blockchain technology*, Journal of Telecommunication, Electronic and Computer Engineering (JTEC), 10 (2018), pp. 29–34.
[12] W. Gräther, S. Kolvenbach, R. Ruland, J. Schütte, C. Torres, and F. Wendland, *Blockchain for education: lifelong learning passport*, in Proceedings of 1st ERCIM Blockchain Workshop 2018, European Society for Socially Embedded Technologies (EUSSET), 2018.
[13] K. Kuvshinov, I. Nikiforov, J. Mostovoy, D. Mukhutdinov, K. Andreev, and V. Podtelkin, *Disciplina: Blockchain for education*, 2018.
[14] D. Lizcano, J. A. Lara, B. White, and S. Aljawarneh, *Blockchain-based approach to create a model of trust in open and ubiquitous higher education*, Journal of Computing in Higher Education, (2019).
[15] H. Mayer, *Ecdsa security in bitcoin and ethereum: a research survey*, CoinFaabrik, June, 28 (2016), p. 126.
[16] S. Nakamoto, *Bitcoin: A peer-to-peer electronic cash system*, 2008.
[17] L. M. Palma, M. A. Vigil, F. L. Pereira, and J. E. Martina, *Blockchain and smart contracts for higher education registry in brazil*, International Journal of Network Management, 29 (2019), p. e2061.
[18] J. Santos, *A non-siloed blockchain-based certification service.*
[19] M. Sharples and J. Domingue, *The blockchain and kudos: A distributed system for educational record, reputation and reward*, in European Conference on Technology Enhanced Learning, Springer, 2016, pp. 490–496.
[20] N. Szabo, *The idea of smart contracts*, Nick Szabo's Papers and Concise Tutorials, 6 (1997).
[21] M. Turkanović, M. Hölbl, K. Košič, M. Heričko, and A. Kamišalić, *Eductx: A blockchain-based higher education credit platform*, IEEE access, 6 (2018), pp. 5112–5127.

Appendix A. Smart contract code of the proposed model.

```
pragma solidity >=0.4.22 <0.6.0;
contract Main{
address private owner;
address[] private revoked;
// Events
event CertAdded (address indexed_from,string reg_no, uint date);
event signed(address indexed_from,bytes32 certHash, uint date);
event verified(address indexed_from,bytes32 certHash, string regno,address[] signers);
event Revoked(address from, address to, uint date);
event RevokedCert(address by ,string  regno, uint date);
event Registered(address from, address to, uint date);
//Structures
struct BlockCert{
bytes32 hash;
string reg_no;
string inst_name;
string name;
string grade;
string degree;
bool isSet;
address[] signedBy;
}
struct AuthCert {
bytes32 apprCert;
uint expiry;
bool revoked;
address[] inch;
bool registered;
}
```

```
30 constructor() public{
31 owner=msg.sender;
32 }
33 // Mappings
34 mapping (address => AuthCert) private authorized;
35 mapping(address=>mapping(string=>BlockCert)) certs;
36 mapping(address=>mapping(bytes32=>bool)) signatures;
37 // Main Modules
38 function register(address institution, bytes32 cHash, address[] memory incharges, uint notAfter)
       public onlyOwner {
39 authorized[institution].apprCert = cHash;
40 authorized[institution].expiry = notAfter;
41 authorized[institution].revoked = false;
42 authorized[institution].registered = true;
43 authorized[institution].inch = incharges;
44 emit Registered(owner, institution, now);
45 }
46 function revoke(address institution) public onlyOwner {
47 require(authorized[institution].registered);
48 authorized[institution].revoked = true;
49 revoked.push(institution);
50 emit Revoked(owner, institution, now);
51 }
52 function check(address institution) public view returns(bool) {
53 require(authorized[institution].registered);
54 if (authorized[institution].revoked || authorized[institution].expiry < now)
55 return false;
56 return true;
57 }
58 function sign(address inst, bytes32 certHash, string memory regno) public notSet(msg.sender,certHash
       ) onlyIncharge(inst,msg.sender)
59 {
60 signatures[msg.sender][certHash] = true;
61 certs[inst][regno].signedBy.push(msg.sender);
62 emit signed(msg.sender,certHash,now);
63 }
64 function isIncharge(address inst,address addr) private view returns (bool)
65 {
66 for (uint i=0; i< authorized[inst].inch.length; i++)
67 {
68 if(authorized[inst].inch[i] == addr) return true;
69 }
70 return false;
71 }
72 function addCert(bytes32 hash,string memory regno,string memory  instname,string memory  name,string
        memory grade,
73 string memory degree) public onlyAuth(msg.sender) ifExists(msg.sender,regno) isSignedByAll(msg.
       sender,hash) {
74 certs[msg.sender][regno].hash= hash;
75 certs[msg.sender][regno].reg_no=regno;
76 certs[msg.sender][regno].inst_name=instname;
77 certs[msg.sender][regno].name=name;
78 certs[msg.sender][regno].grade=grade;
79 certs[msg.sender][regno].degree=degree;
80 certs[msg.sender][regno].isSet= true;
81 emit CertAdded(msg.sender,regno,now);
82 }
83 function revokeCert(address inst, string memory regno) onlyAuth(msg.sender) ifExists(msg.sender,
       regno) public
84 {
85 delete certs[inst][regno];
86 emit RevokedCert(msg.sender, regno, now);
87 }
88 function verify(address inst,string memory  regno)   public
89 returns (bytes32  hash,string memory  instname,string memory  name,
```

```
string memory grade,string memory degree, address[] memory signers){
hash= certs[inst][regno].hash;
regno= certs[inst][regno].reg_no;
instname= certs[inst][regno].inst_name;
name=certs[inst][regno].name;
grade=certs[inst][regno].grade;
degree=certs[inst][regno].degree;
signers= certs[inst][regno].signedBy;
emit verified(inst,hash,regno,signers);
}
function isSigned(address inst, bytes32 hash) private view returns (bool){
uint count=0;
for ( uint i = 0; i < authorized[inst].inch.length; i++)
{
if(signatures[authorized[inst].inch[i]][hash]){
count++;
}
}
return (count==authorized[inst].inch.length) ;
}
// Modifiers
modifier onlyIncharge(address inst, address sender){
require (isIncharge(inst,sender));
_;
}
modifier onlyAuth(address auth)
{
require (check(auth)) ;
_;
}
modifier notSet(address signer,bytes32 hash)
{
require(signatures[signer][hash]== false);
_;
}
modifier isSignedByAll(address escow, bytes32 certHash)
{
require (isSigned(escow,certHash));
_;
}
modifier onlyOwner {
require(msg.sender == owner);
_;
}
modifier ifExists (address inst,string memory regno  ) {
require( certs[inst][regno].isSet== false);
_;
}
}
```

Edited by: Dana Petcu
Received: Sep 22, 2020
Accepted: Dec 13, 2020


Scalable Computing: Practice and Experience, ISSN 1895-1767, http://www.scpe.org
© 2020 SCPE. Volume 21, Issues 4, pp. 689–699, DOI 10.12694:/scpe.v21i4.1807

NVIDIA GPU PERFORMANCE MONITORING USING AN EXTENSION FOR DYNATRACE ONEAGENT

TOMASZ GAJGER *

Abstract. This work presents a Dynatrace OneAgent extension for gathering NVIDIA GPU metrics using NVIDIA Management Library (NVML). The extension integrates GPU metrics into an industry-leading platform for Application Performance Management extending its capability of monitoring important business workloads to the GPU-oriented computational nodes. A practical approach for acquiring and processing NVML metrics via Python bindings is described. The work also proposes and discusses implementation of helper applications for convenient simulation of performance problems in a multi-tier web application. These applications are then used in combination with OneAgent-based monitoring and appropriate configuration of Dynatrace platform for web application monitoring. Next, an end-to-end production-like scenarios are presented, which exemplify extension usefulness in test setup resembling a real world implementation. The extension has been released on GitHub under MIT license.

Key words: Application Performance Management, GPU Performance Monitoring, Dynatrace, GPGPU, CUDA, NVML

AMS subject classifications. 68M20, 68W10

1. Introduction. The drive towards using GPUs for various, general-purpose tasks (GPGPU paradigm) has been growing stronger over recent years. Not only are the GPUs utilized in HPC applications, but an increasing number of software companies start to leverage them for running computational workloads in their backend systems [30]. This evolution is fueled by large enterprises leveraging machine learning methods to provide service to their customers or analyze business-relevant data [7], and backed by extensive research on improving GPUs performance and efficiency [22]. Examples of such workloads are: machine learning, big data, blockchain processing, database queries acceleration, and more. As the GPUs become a vital piece of modern IT infrastructure, a challenge arises to have a reliable and convenient approach for monitoring their performance. While most of the Application Performance Management (APM) vendors offer a robust support for resources like CPUs, RAM, network, hard disks, etc. in their portfolio, same cannot be said for GPUs. Some integrations and dedicated tools exist (see Section 3), still there is yet much to uncover and develop in this area.

This work presents a Dynatrace OneAgent extension for gathering NVIDIA GPU metrics using NVML[1]. Dynatrace is one of the industry leaders in APM [17, 15, 21]. The extension augments Dynatrace platform by feeding it with GPU data for processing by its AI engine for continuous performance tracking and automatic *root cause analysis* (RCA).

Considering NVIDIA's dominance in the field of GPGPU, it was decided to focus initial efforts on said hardware. For example, 27% of supercomputers on the TOP500 list [2] are using these, up from 20% in 2018 or 13% in 2016. Industry has also adopted NVIDIA GPUs for various purposes [30]. In future, the extension could be modified to cover additional types of devices, e.g. these offered by AMD or Intel.

The main contribution of this paper is in extending Dynatrace platform with GPU monitoring capability and making the source code for the extension freely available. Proposed extension is a valuable addition, enabling the platform to support novel use cases in an emerging, dynamically-expanding market. Considering existing APM tools landscape, this is an innovative approach and can serve as a base for further research and development.

The remainder of this paper is organized as follows: Section 2 explains important aspects related to the GPU performance monitoring and briefly describes the Dynatrace product. Section 3 lists related work, while in

*Department of Computer Architecture, Faculty of Electronics, Telecommunications and Informatics, Gdańsk University of Technology, Narutowicza 11/12, 80-233 Gdańsk, Poland (tomasz.gajger@pg.edu.pl)

[1] https://github.com/tomix86/oneagent-nvml-extension

Section 4 proposed extension design and implementation details are explained. Section 5 describes the testbed, testing methodology, and presents the results. Finally, Section 6 summarizes results and outlines the proposed direction of future research.

2. Background.

2.1. GPU Performance Monitoring. Application execution can be monitored either from the outside (*shallow monitoring*) or from within (*deep monitoring*) the process running in the system. In case of **shallow monitoring** the performance metrics are gathered using system, driver or programmatic counters (offered by software framework), or alternatively a logic for exposing them needs to be incorporated up-front into the application during development. For CPU-centric applications, the monitoring focuses on observing measurable effects of the application execution, e.g. CPU usage, RAM consumption. For NVIDIA GPU-based applications, such metrics are collected by CUDA Runtime and are queryable via NVML API [31].

Deep monitoring methods work by instrumenting the application code and extracting execution (performance) metrics from it. For NVIDIA GPUs this type of API is provided by CUDA Profiling Tools Interface (CUPTI) [29]. They usually allow for collection of more fine-grained metrics at the cost of increased overhead, that varies depending on the type of metric being collected. We can distinguish two basic paradigms for instrumenting application code: static (compile-time) and dynamic (run-time). **Static methods** require the instrumentation code to be incorporated into the application during its development, either via direct calls to the API of interest (e.g. CUPTI) or in a form of SDK (e.g. OneAgent SDK for C++ [8]). Applying them requires the monitored application to be recompiled. **Dynamic methods** work after the application was already developed and compiled, often by intercepting certain library (e.g. `ltrace`) or system (e.g. `strace`) calls and/or inserting specialized instrumentation code directly into the application being monitored (e.g. deep monitoring provided by OneAgent). In case of languages that are compiled just-in-time (e.g. CUDA PTX into SASS) it is possible to recompile application code in runtime and insert instrumentation code this way. This is something in-between static and dynamic instrumentation, technically it is dynamic since no explicit code changes are required, but on the other hand the application code representation changes significantly. Hence, two subtypes of dynamic methods can be named: **recompiling** and **automatic**.

The challenge with *static* and *dynamic recompiling* methods is that frameworks or toolkits often ship with already compiled kernels, what does not allow for code changes. Even with access to framework's code, modifying it is not really feasible for any serious production use case, it is simply too cumbersome and time consuming for anyone to bother. Use of *dynamic recompiling* method would incur additional overhead during initial kernel launch as it would have to be compiled from PTX into SASS (see Section 2.1.1 of [16]). The significant benefit of *shallow* and *dynamic automatic* methods is that they do not require this recompilation step and thus provide out of the box support for existing applications. *Static* methods are inferior to other types when it comes to ease of use and coverage for existing applications. While it is apparent that *dynamic automatic* methods are best in terms of usability and provided level of detail, it comes at a cost of difficult development process and increased overhead. *Shallow* methods are good because of their lack of overhead, ease of use, easy development, and best applicability compared to others. Their only drawback is that they offer a considerably limited view into the application behavior.

2.2. NVML. NVML [31] is a C-based API for monitoring and managing NVIDIA GPUs. It enables developers to build applications on top of it and has bindings to several other languages, including Python [28]. NVML, among others, allows to retrieve: list of processes running on a GPU, global memory utilization and current clock rates. It does not allow to retrieve detailed information related to the performance of the kernel, occupancy, SM utilization, and alike.

2.3. Application Performance Management. Application Performance Management (or Monitoring) [19] encompasses software products comprising digital experience monitoring (DEM), application discovery, topology mapping, tracing, diagnostics (identifying faults and aiding in resolving them [3]), integrations into CD pipeline, business analytics, and purpose-built artificial intelligence for IT operations and application developers alike.

Dynatrace is one of vendors delivering APM solutions. It is not only the company name but also the name of the product that they offer - a software platform [11] for monitoring and managing performance of

applications, digital experience and business analytics. **OneAgent** [10] component is a piece of the platform that users install on their operating systems. The agent gathers infrastructure metrics, monitors log files, detects processes and instruments them. The performance data is then reported to the **Dynatrace Cluster**. **OneAgent SDK** [8] may be used to add process-level request tracing to applications not supported by the OneAgent natively. **OneAgent Extensions** offer a mechanism to extend the collection of metrics OneAgent gathers by writing scripts in Python. An SDK [9] is provided that is a base for writing custom extensions that developers may use to collect, process, and send the data to the Dynatrace Cluster. Within the cluster a component called **Davis** operates, which is a deterministic AI causation engine. It analyzes all incoming data, including but not limited to: process and infrastructure metrics (also the ones from custom extensions), application topology, log files, performance and availability events. By leveraging automatic baselining, it detects anomalies in metrics and if such occur, or in case an unexpected event is encountered, it reports a *problem* informing the system operator about the issue at hand. On top of that, using a deterministic algorithm it performs an RCA for each given problem, pointing at the chain of events that led to it, its root cause and impact on end user. **Synthetic and real user monitoring** [12] are key elements for the DEM offered by the Dynatrace platform. Both components are used to monitor actual performance of web pages, the former one by configuring and sending pre-created web requests to analyze their behavior based on received responses. The latter one gathers statistics from real user visits on webpage in question as they traverse throughout the application stack.

3. Related work and software. This section presents related work relevant to the paper subject, a differentiation is made between software build purposely to gather GPU metrics and support for these in scope of generic APM products.

3.1. GPU metrics gathering software. Two basic types of GPU metrics gathering software can be recognized, one that operates in scope of a single host, and distributed solutions providing a centralized access to metrics coming from an arbitrary number of hosts.

Single host can be monitored using NVIDIA-made utility called `nvidia-smi` [27], which wraps NVML in a convenient command-line interface. There are various similar tools, most popular of which seem to be `gputstat` [5] and `NVTOP` [32].

Distributed monitoring is offered by NVIDIA via `DCGM` [25], also a wrapper over NVML, but able to work in a clustered environment. It offers storage for historical data, batch queries, healthchecks, and diagnostics. Integrations for following third-party products are available: `Prometheus`, `Grafana`, `IBM Spectrum LSF`, and `collectd`. `Bright Cluster Manager` [4] is a proprietary product that leverages DCGM underneath. On the other hand, `Ganglia` [1], builds directly on top of NVML, leveraging its Python bindings [28], while `Influxdata Telegraf` [20] uses `nvidia-smi` to collect the metrics. The authors of [14] propose an extension for existing `Integrated Performance Monitoring` toolset to monitor GPUs by dynamically instrumenting application code to gather timing metrics, and have proven its usefulness in a multi-node GPU cluster.

3.2. GPU monitoring support in APM products. In current APM offering, the following products offer a built-in (or opt-in via extension) support for GPU monitoring. `Google Cloud operations suite` available on Google Cloud Platform (GCP) [18]. `New Relic` provides support to a limited extent by using metrics coming from GCP Kubernetes Engine [24]. The case is similar for `Datadog` [6], which is able to use metrics coming from other providers (GCP, Kubernetes, Mesos, Amazon SageMaker, Alibaba Cloud), there are also user-made plugins for it [23] - an approach similar to the one proposed in this paper.

4. Extension design and implementation. The extension's implementation leverages Python bindings for NVML [28]. It is is capable of monitoring multiple GPUs, the metrics coming from all the devices are aggregated and sent as combined time series. There is no support for sending separate time series per device.

4.1. Metrics. Once Dynatrace OneAgent is installed and extension is deployed, NVML metrics are periodically gathered and sent to the Dynatrace Server. *HOST* metrics are reported for the host entity, while *PGI* metrics are reported for process entity. PGI is an acronym for *Process Group Instance* and, in simplification, denotes a process running in a system. The list includes:

- `gpu_mem_total` (HOST) - total available global memory,

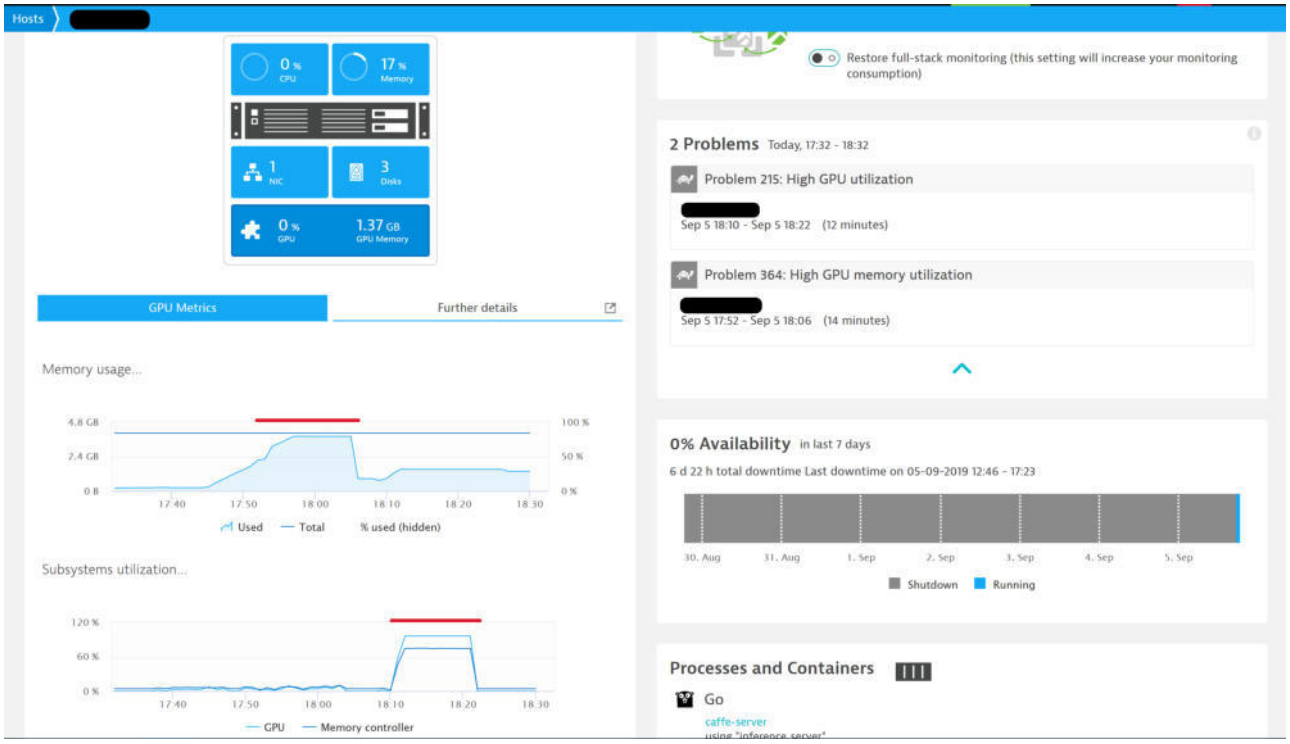


Fig. 4.1. *Host metrics reported by the extension*

- `gpu_mem_used` (HOST) - device (global) memory usage,
- `gpu_mem_used_by_pgi` (PGI) - global memory usage per process,
- `gpu_mem_percentage_used` (HOST) - artificial metric for raising *High GPU memory* alert,
- `gpu_utilization` (HOST) - percent of time over the past sample period (within CUDA driver) when a kernel was executing on the GPU,
- `gpu_memory_controller_utilization` (HOST) - percent of time over the past sample period (within CUDA driver) when global memory has been accessed,
- `gpu_processes_count` (HOST) - number of processes making use of the GPU.

If there are multiple GPUs present, the metrics are displayed in a joint fashion:

- `gpu_mem_total` is a sum of all the devices' global memory,
- `gpu_mem_used` and `gpu_mem_used_by_pgi` is the total memory usage across all the devices,
- `gpu_utilization` and `gpu_memory_controller_utilization` is an average from per-device usage metrics,
- `gpu_processes_count` shows unique count of processes using any of the GPUs. That is, if a single process is using two GPUs, it is counted as one.

These metrics can be used to observe GPU performance on the system in question:

- GPU core utilization, see Fig. 4.1,
- GPU memory subsystem utilization, see Fig. 4.1,
- Per-process GPU memory usage, see Fig. 4.2,
- Automatic notifications about performance problems (e.g. in case GPU usage exceeds predefined threshold), see Fig. 4.3.

Metrics are automatically correlated with particular hosts and processes, and thus are used by the Davis for evaluating root cause of performance problems. See Fig. 4.4 for an example where metrics anomalies are reported.

In the current extension version (v0.2.0), the following NVML device queries are utilized to gather metrics:

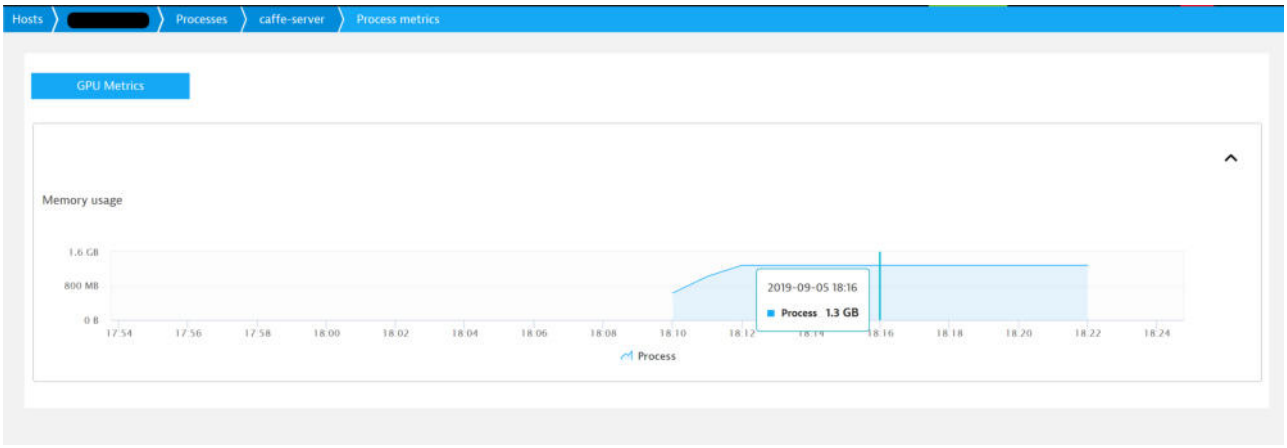


FIG. 4.2. *Process metrics reported by the extension*

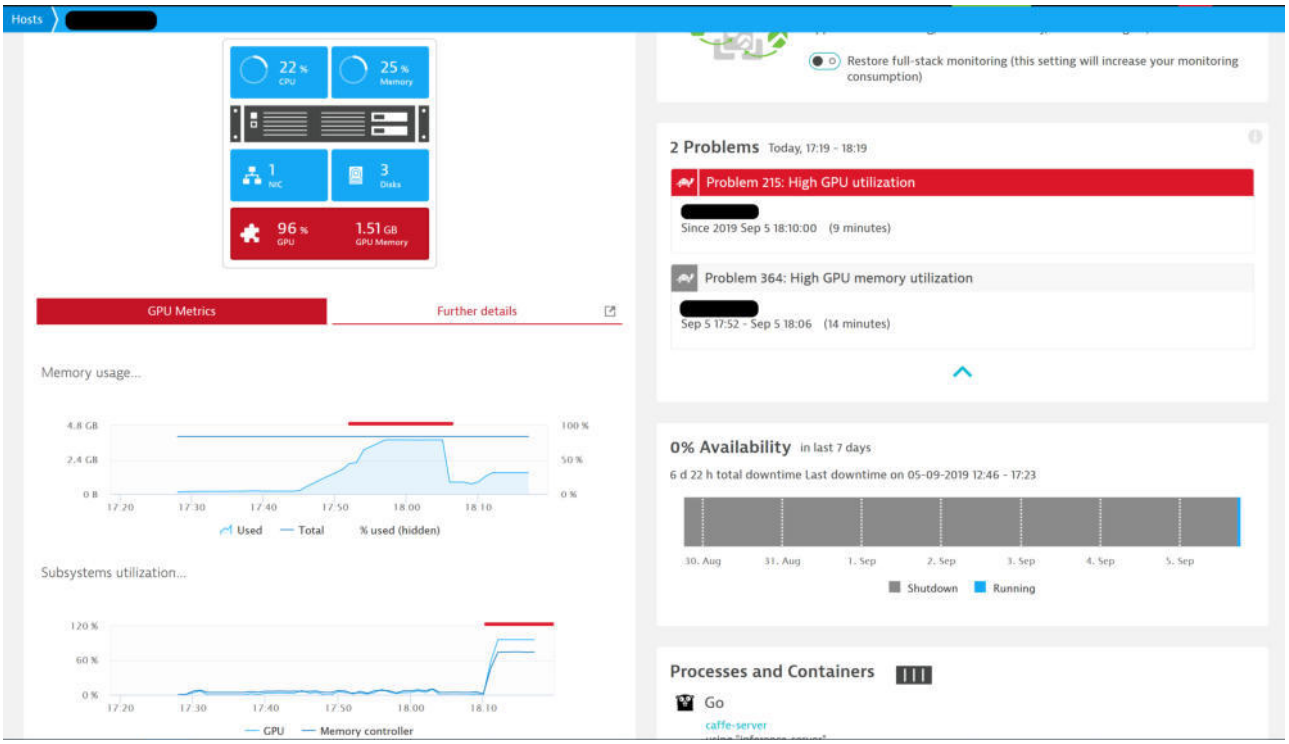


FIG. 4.3. *Alerts as seen on the host screen*

- `nvmlDeviceGetComputeRunningProcesses` - get information about processes with a compute context (non-graphics) on a device,
- `nvmlDeviceGetGraphicsRunningProcesses` - get information about graphics-based processes,
- `nvmlDeviceGetMemoryInfo` - get the amount of used, free and total memory available on the device,
- `nvmlDeviceGetUtilizationRates` - get the current utilization rates for the device's major subsystems: graphics unit and global memory.

Gathering metrics externally via NVML (contrary to CUPTI) does not incur any additional overhead on the observed applications. Internally, the extension collects several data samples and aggregates them before passing them on to the framework execution engine. By default, 5 samples in 2 second intervals are collected.

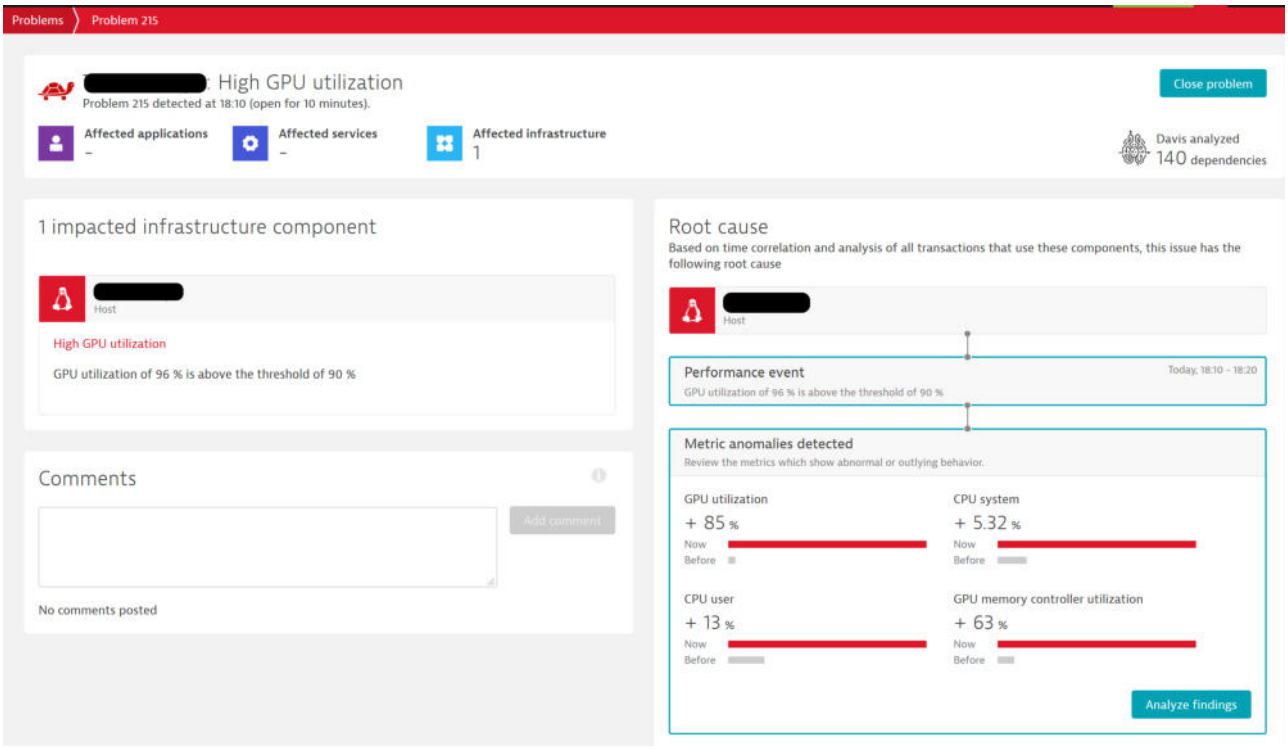


FIG. 4.4. *Problem screen with high GPU utilization alert*

These values are customizable.

Concerning per-PGI memory usage, on Windows this metric won't be available if the card is managed by WDDM driver, the card needs to be running in TCC (WDM) mode. This mode is not supported by GeForce series cards prior to Volta architecture [26].

4.2. Alerting. Three alerts are predefined in the extension, all three are generated by Davis when metrics exceed certain threshold values. These alerts are reported for the host entity and are visible on the host screen (see Fig. 4.3):

- *High GPU utilization* alert - raised when `gpu_utilization` exceeds predefined threshold (default: 90%) in given time period,
- *High GPU memory controller utilization* alert - raised when `gpu_memory_controller_utilization` exceeds predefined threshold (default: 90%) in given time period,
- *High GPU memory utilization* alert - raised when `gpu_mem_percentage_used` exceeds predefined threshold (default: 90%) relative to `gpu_mem_total` in given time period.

Alerts thresholds are customizable on the WebUI. Note that high GPU memory utilization alert is based on two separate metrics, due to current extension framework limitations, it is not possible to define such alert server-side. Thus, an artificial metric that is hidden on the memory usage chart, representing percentage usage of the GPU memory had to be introduced.

5. Experiments and results.

5.1. Helper applications. Following helper applications were used to aid with testing, two of which were developed for the purpose of this work:

- A C++ application simulating heat distribution in a two dimensional solid body[2], further referred to as *Backend Application*,

[2] `https://github.com/tomix86/webserver-test-app`

- A load generator[3] capable of using arbitrary amount of GPU global memory with control over how the usage progresses in time,
- glmark2 [13], launched to run infinitely: `$ glmark2 --run-forever`.

Backend Application is a simple webserver, exposing REST endpoints, that responds to queries with image depicting computed result. A sample query looks as shown in Listing 1, where subsequent GET parameters stand for: CUDA block size X, CUDA block size Y, 2D body mesh side length, algorithm iterations.

```
<address>/heat-distrib?16&32&100&1000
```

Listing 1
Example query

Internal implementation leverages CUDA, NVTX and C++ REST SDK. It simulates heat distribution in a solid, where finite difference method was used to discretize steady state differential equation describing heat diffusion in the object. Environment temperature, which is a boundary condition, was assumed to be constant. Additionally, the object is assumed to be a two-dimensional square mesh, with a side of size N, so there are $N \times N$ distinct points on the mesh for which the value of the temperature has to be updated during each simulation step. The final equation, applied to each cell is a five-point 2D stencil.

5.2. Testbed setup. To showcase extension capabilities, a testbed was prepared that exemplifies, in a simplified form, a production-like environment. It consists of:

1. Host running Apache2, with a public IP address, acting as both a *Frontend Server* and reverse proxy that exposes the webpage to the internet, monitored by OneAgent,
2. *Backend Server*, monitored by OneAgent with NVML extension installed, running *Backend Application* instrumented via OneAgent SDK,
3. Machine with *ActiveGate* installed and *Synthetic monitoring* (a private location [12]) enabled:
   - *Browser monitor* [12] configured, by recording a clickpath, to query http://<*Frontend Server* IP>/heatpage.html. This is a webpage that upon load fetches additional resource from `/heat-distrib` endpoint exposed by *Backend Server*,
   - *HTTP monitor* [12] configured to query http://<*Frontend Server* IP>/heat-distrib, which goes via reverse proxy to the *Backend Server*.

Browser monitor and *HTTP monitor* are jointly referred to as *Synthetic monitoring*.

Above setup contains 3 essential components of a modern web application: a frontend page, a webserver serving said page, and a backend host running computations. Hence, they can act as a minimal, but still an accurate setup where extension usefulness can be proven. It is not very different from, let's say, a webpage with an interactive voice assistant, which needs to query a backend underneath that performs natural language recognition using machine learning methods on GPU-based computational nodes. If performance of such assistant would be impaired, it would affect real users of said webpage and could thus lead to profit loss and customer dissatisfaction.

5.3. Tests. Two tests were conducted, both of which present an end-to-end monitoring scenario with causality analysis when a problem occurs. *Synthetic monitoring* is used to measure *Frontend Server* response times, while NVML extension feeds GPU data into Davis. The problems are generated by impairing the GPU performance on *Backend Server* causing **increased response times** and **request processing failures** respective for test cases one and two. For both scenarios an RCA for performance degradation measured via *Synthetic monitoring* is shown.

5.3.1. Test case 1. This case is an end-to-end monitoring scenario, where *Backend Application* is responding slower due to high (close to 100%) GPU core utilization, with load generated by `glmark2`.

As shown in Fig. 5.1, the problem screen displays a causality analysis and pinpoints the root cause to be a malfunctioning computational node. Note the metric anomalies being detected, these are coming from instrumentation via OneAgent SDK. One can then drill-down to the problem analysis (Fig. 5.2) and see that the extension has detected the issue, with the cause clearly identified to be high GPU utilization, as shown on the host screen (Fig. 5.3). The scenario presented here shows data from host GPU metrics reported by extension

[3] `https://github.com/tomix86/cuda-load-generator`

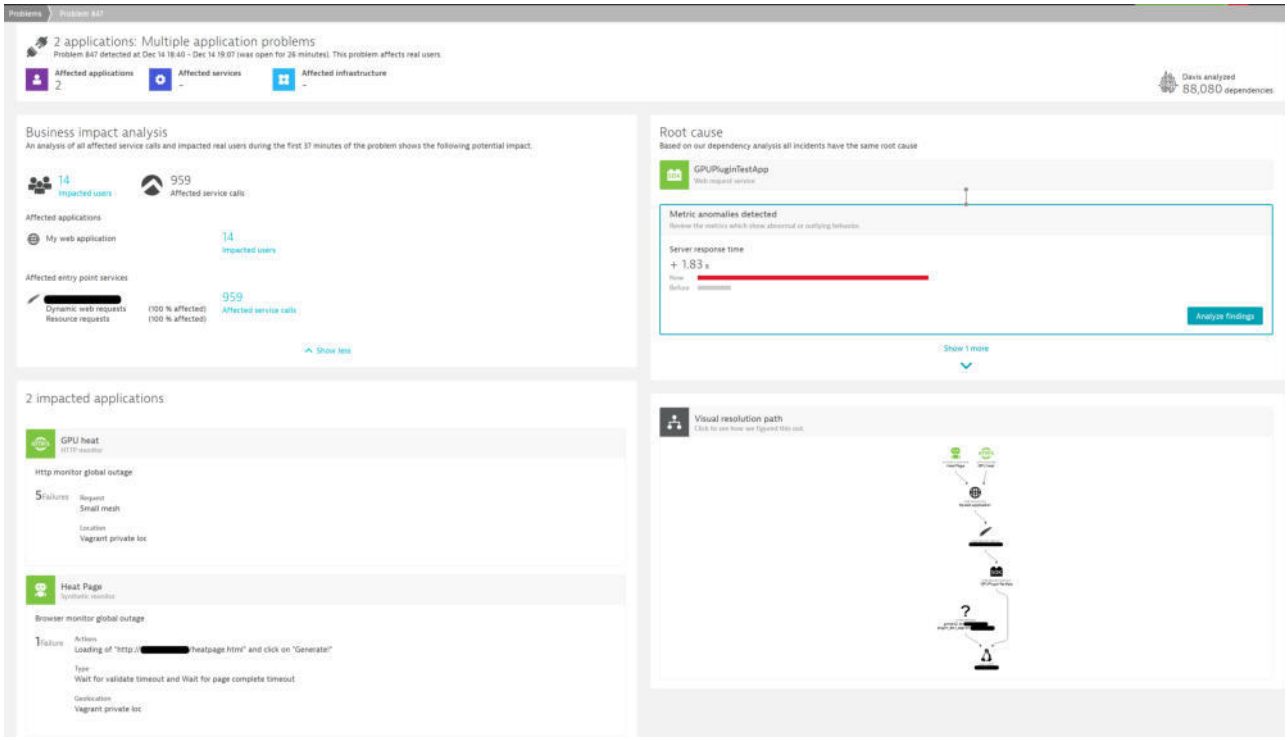


FIG. 5.1. *Detected performance problem (left) and RCA (right)*

FIG. 5.2. *Problem analysis screen with visual resolution path*

being automatically correlated to availability issues in web application. In future version of the Dynatrace platform an improvement could be made such that the problem screen (Fig. 5.1) would directly display the metrics from extension, effectively providing accurate RCA without a need to perform additional steps - such as host screen inspection.

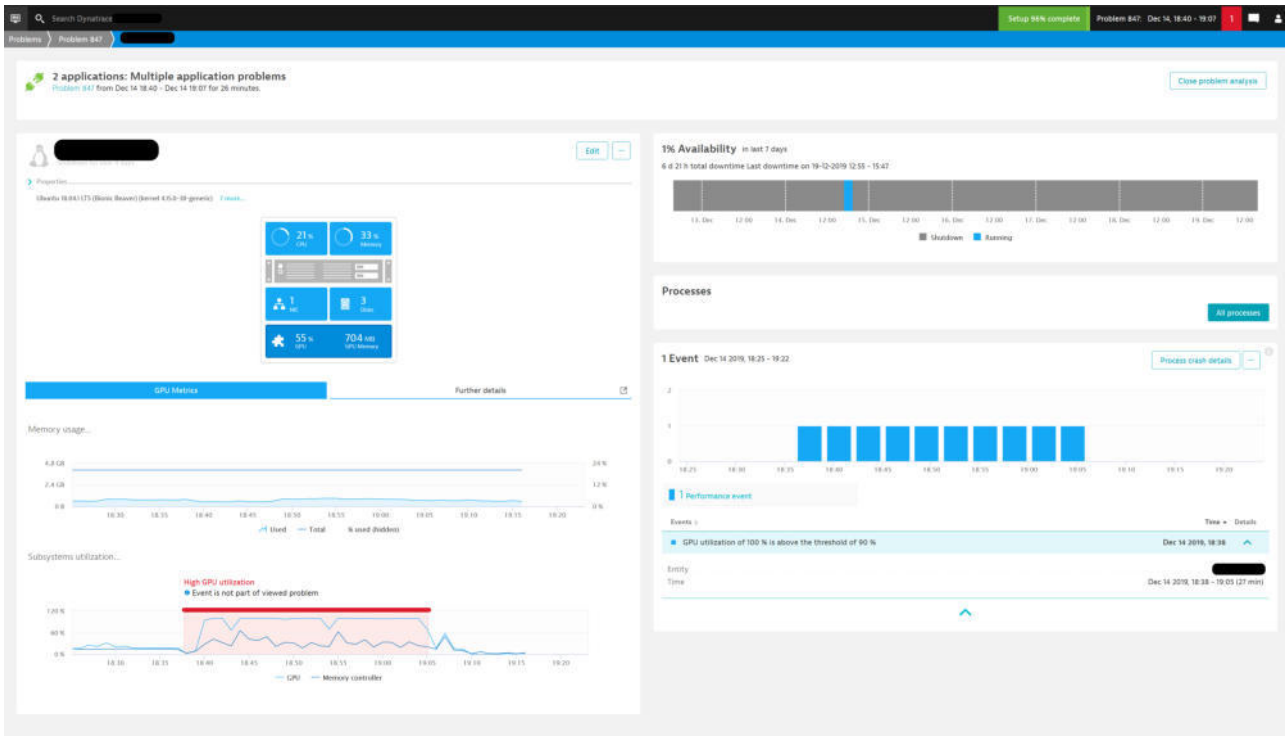


FIG. 5.3. *Host screen with high GPU utilization alert*

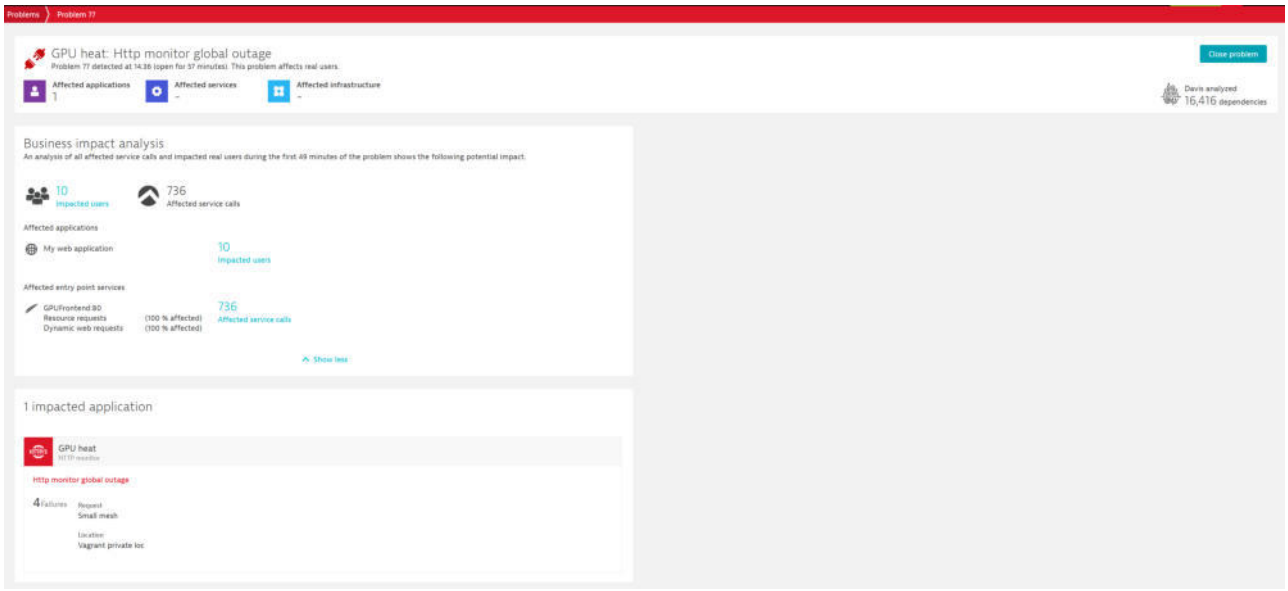


FIG. 5.4. *Detected availability problem (left) and missing RCA (right)*

5.3.2. Test case 2. In this scenario, *Backend Application* failures due to inability to allocate sufficient memory are simulated, the memory is occupied by CUDA Load Generator. Contrary to the previous test case, Dynatrace fails to identify the root cause (Fig. 5.4), even though an alert for high GPU memory occupancy is raised for the host in question. By viewing the host screen in context of the mentioned problem, one can see an

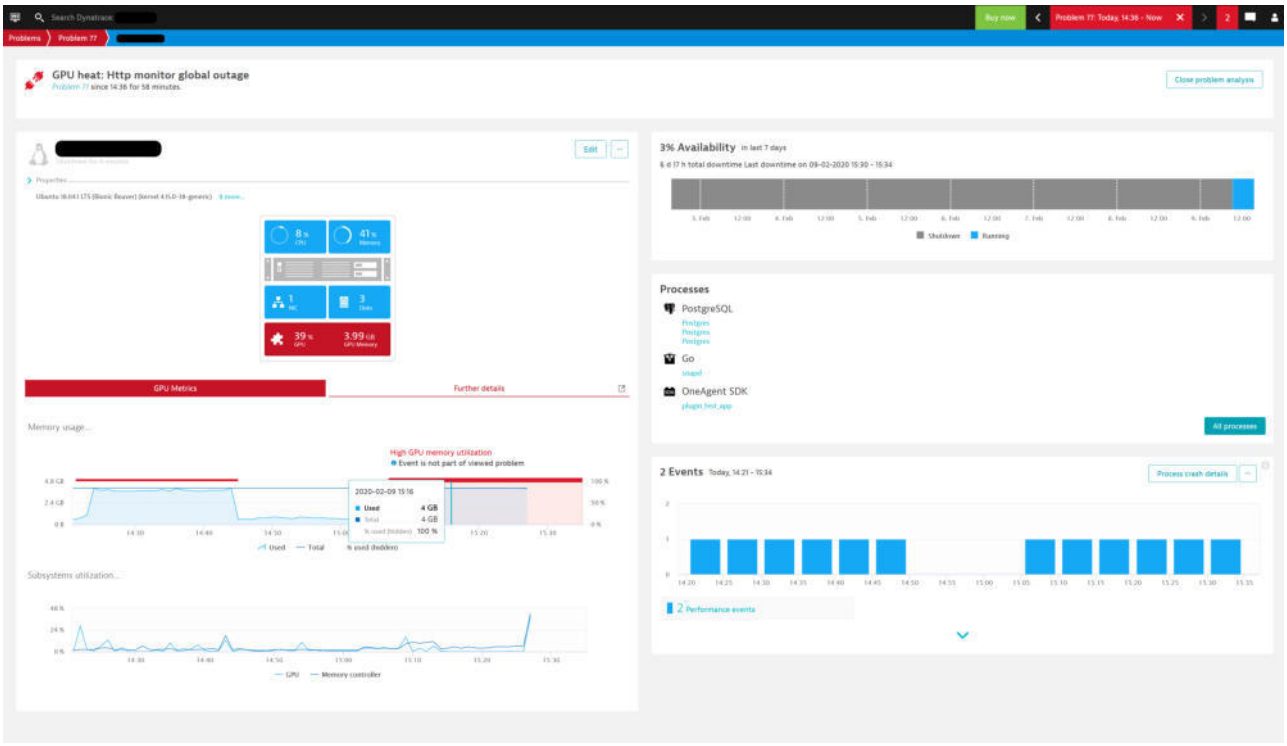


FIG. 5.5. *Host screen in context of the availability problem*

explicit tooltip indicating lack of correlation (Fig. 5.5). Even though there is no clear RCA, the data provided by extension is still a valuable source of information and shows that GPU ran out of memory resources. On this basis one can easily infer that this had most likely caused request processing failures of the web service in question.

6. Summary and future work. This paper presented an extension for Dynatrace OneAgent that enabled it to monitor GPU performance by gathering NVML metrics. It was shown how to leverage existing tools and software libraries to monitor the performance of a GPU and alert about performance problems. Extension usefulness was proven using two production-like scenarios, where it helped to quickly identify the root cause for performance degradation and availability problem in a customer-facing web application. During solution validation few shortcomings of the Dynatrace platform were identified, namely lack of out-of-the-box support for including extension metrics in RCA and deficiencies in extension SDK. Nevertheless, the end result was satisfying.

The author believes that the extension will be useful for existing users of Dynatrace, who manage infrastructure that includes GPU-enabled nodes, and that the implementation details, plus the freely available source code would help other developers to extend rest of the APM tools with similar support for GPU monitoring, which is very limited at the time of writing.

In future, several improvements are planned. Scope of metrics gathered should be broadened to include: CUDA properties, ECC errors count, GPU core and memory clock speeds, power cap, and throttling events. By leveraging SDK-like approach, where users would need to modify their application's code, the author plans to also collect kernel execution and memory transfer durations to identify kernels that violate an automatically predetermined baseline for a given set of input parameters. Improvements in available alerting profiles should also be considered to identify compound problem patterns, e.g. alert on unexpected underutilization of a GPU (device being idle, while it is expected to process data). The extension should also be validated on a wider set of GPU architectures, especially Volta or higher to verify if per-process memory occupancy can be reported

on Windows. Lastly, there are plans to develop another extension, that leverages CUPTI for collection of code-level metrics.

REFERENCES

[1] *Ganglia monitoring system*. https://developer.nvidia.com/ganglia-monitoring-system.
[2] *Top 500 list*. https://top500.org/lists/top500/2020/06/.
[3] T. M. Ahmed, C. Bezemer, T. Chen, A. E. Hassan, and W. Shang, *Studying the effectiveness of application performance management (apm) tools for detecting performance regressions for web applications: An experience report*, 2016 IEEE/ACM 13th Working Conference on Mining Software Repositories (MSR), (2016), pp. 1–12.
[4] Bright Computing, *Bright cluster manager*. https://www.brightcomputing.com/documentation.
[5] J. Choi, *gpustat*. https://github.com/wookayin/gpustat.
[6] Datadog, *Documentation*. https://docs.datadoghq.com.
[7] S. Dutta, *An overview on the evolution and adoption of deep learning applications used in the industry*, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 8 (2017).
[8] Dynatrace, *Oneagent sdk documentation*, October 2019. https://github.com/Dynatrace/OneAgent-SDK-for-C.
[9] ———, *Extension sdk documentation*, September 2020. https://www.dynatrace.com/support/help/shortlink/oneagent-extensions-tutorial.
[10] ———, *Oneagent documentation*, September 2020. https://www.dynatrace.com/support/help/shortlink/oneagent-hub.
[11] ———, *The software intelligence platform*, September 2020. https://www.dynatrace.com/platform.
[12] ———, *Synthetic monitoring documentation*, September 2020. https://www.dynatrace.com/support/help/shortlink/synthetic-hub.
[13] A. Frantzis and J. Barker, *glmark2 - an opengl 2.0 and es 2.0 benchmark*, June 2015. https://github.com/glmark2/glmark2.
[14] K. Fürlinger, N. Wright, and D. Skinner, *Comprehensive performance monitoring for gpu cluster systems*, IEEE International Symposium on Parallel and Distributed Processing Workshops and Phd Forum, (2011), pp. 1377 – 1386.
[15] G2, *Best application performance monitoring (apm) software*, September 2020. https://www.g2.com/categories/application-performance-monitoring-apm.
[16] T. Gajger and P. Czarnul, *Modelling and simulation of gpu processing in the merpsys environment*, Scalable Computing: Practice and Experience, 19 (2018), pp. 401–422.
[17] Gartner, *Application performance monitoring market*, September 2020. https://www.gartner.com/reviews/market/application-performance-monitoring.
[18] Google, *Compute engine - monitoring gpu performance*. https://cloud.google.com/compute/docs/gpus/monitor-gpus.
[19] C. Heger, A. van Hoorn, M. Mann, and D. Okanović, *Application performance management: State of the art and challenges for the future*, Proceedings of the 8th ACM/SPEC on International Conference on Performance Engineering, (2017), pp. 429–432.
[20] InfluxData, *Telegraf*. https://www.influxdata.com/time-series-platform/telegraf/.
[21] IT Central Station, *Best application performance monitoring & management (apm) tools*, September 2020. https://www.itcentralstation.com/categories/application-performance-management-apm.
[22] M. Khairy, A. Wassal, and M. Zahran, *A survey of architectural approaches for improving gpgpu performance, programmability and heterogeneity*, Journal of Parallel and Distributed Computing, 127 (2019).
[23] T. Nagai, *datadog_nvml*. https://github.com/ngi644/datadog_nvml.
[24] New Relic, *Google kubernetes engine monitoring integration*. https://docs.newrelic.com/docs/integrations/google-cloud-platform-integrations/gcp-integrations-list/google-kubernetes-engine-monitoring-integration.
[25] NVIDIA Corporation, *Dcgm*. https://developer.nvidia.com/dcgm.
[26] ———, *Supported Compute Debugger Configurations*. https://developer.nvidia.com/nsight-visual-studio-edition-supported-gpus-full-list#SupportedComputeConfigs.
[27] ———, *nvidia-smi - nvidia system management interface*, July 2016. http://developer.download.nvidia.com/compute/DCGM/docs/nvidia-smi-367.38.pdf.
[28] ———, *Python bindings for the nvidia management library*, June 2017. https://pypi.org/project/nvidia-ml-py3.
[29] ———, *Cupti user's guide*, August 2020. https://docs.nvidia.com/cuda/pdf/CUPTI_Library.pdf.
[30] ———, *Gpu applications catalog*, September 2020. https://www.nvidia.com/en-us/gpu-accelerated-applications/.
[31] ———, *Nvml api reference guide*, June 2020. https://docs.nvidia.com/pdf/NVML_API_Reference_Guide.pdf.
[32] M. Schmitt, *nvtop*. https://github.com/Syllo/nvtop.

Edited by: Dana Petcu
Received: Sep 28, 2020
Accepted: Dec 7, 2020


Scalable Computing: Practice and Experience, ISSN 1895-1767, http://www.scpe.org
© 2020 SCPE. Volume 21, Issues 4, pp. 701–707, DOI 10.12694:/scpe.v21i4.1837

PARALLEL ALGORITHM FOR NUMERICAL METHODS APPLIED TO FRACTIONAL-ORDER SYSTEM

FLORIN ROŞU *

Abstract. A parallel algorithm is presented that approximates a solution for fractional-order systems. The algorithm is implemented in CUDA, using the specific GPU capabilities. The numerical methods used are Adams-Bashforth-Moulton (ABM) predictor-corrector scheme and Diethelm's numerical method. A comparison is done between these numerical methods that adapts the same algorithm for the approximation of the solution.

Key words: GPU processing, HPC processing, parallel algorithm, numerical methods, fractional-order systems

AMS subject classifications. 65Y05

1. Introduction. A few applications of the fractional-order derivative system that models real world phenomena are presented in [1, 14, 15, 17]. Unfortunately, the analytical method are not yet discovered for finding the solution of fraction-order derivatives system. The numerical methods provides an approximation of the solution.

The Adams-Bashforth-Moulton predictor-corrector method is widely study [2, 3] and still improved to get more accurate solution [18]. There are several implementation of this method using different parallel computing technologies: MPI and OpenMP [4, 16]; Matlab [5].

The parallel numerical algorithm that was implemented for BlueGene/P supercomputer [7] and adapted to run on GPU [8] is using the Adams-Bashforth-Moulton predictor-corrector method that estimates the solution for Caputo-type fractional-order system. In this paper we adapt the same algorithm to estimate the solution using Diethelm's method [13] and compare them from different point of view.

2. Preliminaries. The fractional order equations has several definitions and to mention a few there is the Riemann-Louville definition, the Caputo type, Grünwald-Letnikov. In some conditions, some of them are equivalent [11].

The Caputo-type definition is also known as the initial value problem. We will considered the simplified written form:

$$\begin{cases} \mathrm{D}_*^{\alpha} y(t) = f(t, y(t)), t \in [0, T] \\ y(0) = y_0 \end{cases} \tag{2.1}$$

where $0 < \alpha < 1$.

The interval $[0, T]$ which represents the elapsed time T is split into steps of h. The accuracy of the estimation is given by h with a smallest value possible. The number of points where the estimation is computed is $N = \frac{T}{h}$. A nth value in our solution it is caracterized for a point in time $t_n = n\dot{h}$. So, the elapsed time from 0 to T will be simulated by t_n values.

2.1. ABM predictor-corrector method. The numerical method Adams-Bashforth-Moulton predictor corrector is described [2] by the following steps.

Having the initial condition as y_0 and at the time t_n having computed the $y_n = y(t_n)$ and $f_n = f(t_n, y_n)$, in order to compute the next value of y_{n+1} first we have to compute the **predictor**, which will give a first

* Dept. of Mathematics and Computer Science, West University of Timişoara, Timişoara, Romania, (florin.rosu@e-uvt.ro)

approximation y_{n+1}^P of our solution:

$$y_{n+1}^P = \sum_{k=0}^{\lceil\alpha\rceil-1} \frac{t_{n+1}^k}{k!} y_0^{(k)} + h^\alpha \sum_{k=0}^{n} b_{n-k} f_k, \tag{2.2}$$

where

$$b_n = \frac{(n+1)^\alpha + n^\alpha}{\Gamma(\alpha+1)}.$$

When computing the **corrector**, we will have an approximation of the solution for the time t_{n+1} with:

$$y_{n+1} = \sum_{k=0}^{\lceil\alpha\rceil-1} \frac{t_{n+1}^k}{k!} y_0^{(k)} + h^\alpha \left(c_n f_0 + \sum_{k=1}^{n} a_{n-k} f_k + \frac{f(t_{n+1}, y_{n+1}^P)}{\Gamma(\alpha+2)} \right), \tag{2.3}$$

where the weights a_n and c_n are defined as:

$$a_n = \frac{(n+2)^{\alpha+1} - 2(n+1)^{\alpha+1} + n^{\alpha+1}}{\Gamma(\alpha+2)}$$

and

$$c_n = \frac{n^{\alpha+1} - (n-\alpha)(n+1)^\alpha}{\Gamma(\alpha+2)}$$

2.2. Diethelm's method. The Caputo-type definition of fractional ordered equation can be written also in the form [13]:

$$D^\alpha[y - y_0] = \beta y(t) + f(t), \text{ where } 0 \le t \le 1 \tag{2.4}$$

As it can be seen in the equation 2.4 the time interval is considered to be $[0, 1]$, while the initial condition y_0 is incorporated in the equation itself.

According to Diethelm [13], the approximation of the solution is given by:

$$y_k = \frac{1}{\Theta_{0k} - \left(\frac{k}{n}\right)^\alpha \Gamma(-\alpha)\beta} \left(\left(\frac{k}{n}\right)^\alpha \Gamma(-\alpha) f_k - \sum_{j=0}^{k} \Theta_{jk} y_{k-j} - \frac{y(0)}{\alpha} \right) \tag{2.5}$$

where the weight Θ_{jk} are obtained as a solution of the equation:

$$\alpha(1-\alpha)k^{-\alpha}\Theta_{jk} = \begin{cases} -1 \text{ when } j = 0 \\ 2j^{1-\alpha} - (j-1)^{1-\alpha} - (j+1)^{1-\alpha} \text{ when } 0 < j < k \\ (\alpha-1)j^{-\alpha} - (j-1)^{1-\alpha} + j^{1-\alpha} \text{ when } j = k \end{cases} \tag{2.6}$$

3. Parallel numerical simulation in CUDA.

3.1. Numerical algorithm for ABM method. The Adams-Bashforth-Moulton predictor corrector method was implemented in CUDA using a parallel algorithm [8]. The core of the algorithm can be described in the Algorithm 1.

The main challenge in the algorithm 1 is the computation of predictor and corrector in a parallel environment. More precise, for the predictor (2.2) the part $\sum_{k=0}^{n} b_{n-k} f_k$ and for the corrector (2.3) the part $\sum_{k=1}^{n} a_{n-k} f_k$.

In this implementation, at start, the weights a_n, b_n and c_n are computed in parallel and stored in the global memory.

Algorithm 1: Parallel Algorithm in CUDA

Data: T end of the time interval.
Data: N global number of points.
Data: B number of Blocks.
Data: $Threads$ number of threads.
Data: SOL numerical solution.
$N_P \leftarrow N/P$;
$y_0 \leftarrow$ initial condition;
$Threads \leftarrow 1024$;
`WEIGTHS<<<`B`,` $Threads$`>>>`(N, a, b, c);
for $n \in [1, N]$ **do**
  $B = \sqrt{n/Threads} + 1$;
  `cudaDeviceSynchronize();`
  `/* Compute the partial predictor/corector in each block */`
  `SUM<<<`B`,` $Threads$`>>>`(n, PP_B, PC_B);
  `cudaDeviceSynchronize();`
  `/* Reduce predictor/corector and compute new` SOL_n `*/`
  `Reduce<<<1,B>>>`(n, PP_B, PC_B, SOL);
end
`cudaMemCopy(`SOL`, DeviceToHost);`

Algorithm 2: Partial SUM computation

`SUM`(n, PP_B, PC_B)
begin
  for *(each blockId)* **do**
    Data: $shmP$[1024] shared memory for predictor
    Data: $shmC$[1024] shared memory for corrector
    for *(each threadId)* **do**
      `compute` $shmP[threadId]$;
      `compute` $shmC[threadId]$;
    end
    $PP\_B[blockId] \leftarrow$ *reduce* $shmP$;
    $PC\_B[blockId] \leftarrow$ *reduce* $shmC$;
  end
end

In CUDA the sum reduction can be done using two kernels [9]. The kernel *SUM* computes partial sums for each block. The partial sum in blocks are computed in parallel using 1024 threads. Each thread compute it's own partial sum. The algorithm for *SUM* kernel is presented in the Algorithm 2.

The reduction in the *SUM* kernel is done at block level, using shared memory in each block [10]. As a detailed implementation in CUDA, the reduction is done in two steps. In the first step, all 1024 threads are divided in 2 groups, each thread from the lower id's will add the result from the thread in the upper part. In this case, the threads from $0 \ldots 511$ will add the sum from the threads $512 - 1023$. Then, the $0 \ldots 511$ is split again in 2 groups, so the threads $0 \ldots 255$ will add the sum from the threads $256 \ldots 511$. This process of splitting and adding it is repeated until it remains only 32 threads.

The second step is the reduction from the last 32 threads, that can be computed directly. This is possible with the NVidia hardware architecture [9]that runs in parallel instructions from warps of 32 thread.

The second kernel runs in one block, with the number of threads as the number of blocks from the previous

kernel. The partial sums for predictor and corrector were computed by the *SUM* kernel and stored the results in *PP_B* and *PC_B*.In the *Reduce* kernel the final reduction from each block is done, so the critical part in computing the predictor and corrector is solved. In the same kernel, after the reduction is done, in one of the thread, the ABM method will be apply and with the predictor and corrector, the approximated solution will be computed. The algorithm 3 describes how the final value for the solution at step n is obtained.

Algorithm 3: Reduce and approximate SOL

```
Reduce(n, PP_B, PC_B, SOL)
begin
    Data: sumP sum for predictor
    Data: sumC sum for corrector
    Data: predictor the value of the predictor
    Data: corrector the value of the corrector
    sumP ← reduce PP_B;
    sumC ← reduce PC_B;
    if (threadId == 0) then
        predictor ← compute from sumP;
        corrector ← compute from predictor and sumC;
        SOL[n] ← compute from corrector;
    end
end
```

All the computations and the results are stored in global GPU's memory. Only at the end of the algorithm the results are transfer to RAM and saved the numeric solution on HDD.

The most efficient way to compute in a parallel environment is to have the work balanced between threads/blocks. That's why, for a step of n, having 1024 threads for each block, the balanced is establish by having $\sqrt{n/Threads} + 1$ blocks.

3.2. Numerical algorithm for Diethelm method. For the approximation of the solution using Diethelm's method [13], the Algorithm 1 is adapted. The core of the algorithm remains the same.

The adaption is done only in the kernels. In the *SUM* kernel, there is only one sum to be computed. The downside is that the weights needs to be generated each time when the partial sum needs them for computation.

The *SUM* kernel can be presented in the Algorithm 4.

Algorithm 4: Partial SUM computation

```
SUM(n, P_B)
begin
    Data: Θ[n] weights for step n
    for (i ∈ [0...n]) do
        compute Θ_{i,n};
    end
    for (each blockId) do
        Data: shmP[1024] shared memory for sum
        for (each threadId) do
            compute shmP[threadId];
        end
        P_B[blockId] ← reduce shmP;
    end
end
```

The *Reduce* kernel is easily adapted from the Adams-Bashforth-Moulton implementation. In Algorithm 5 it is presented the core execution of the kernel, and it can be seen that it's actually a simplified version.

Algorithm 5: Reduce and approximate SOL

```
Reduce(n, P_B, SOL)
begin
    Data: sumP sum that needs to be reduced
    sumP ← reduce P_B;
    if (threadId == 0) then
        SOL[n] ← compute from sumP;
    end
end
```

4. Numerical experiment and simulation results. For our simulations we use the fractional-order equation described in [13] and have the following Caputo fractional-order operator:

$$D^{\alpha}[y - y_0] = \beta y(t) + f(t) \tag{4.1}$$

with $y_0 = 0$, $t \in [0, 1]$, $\beta = -1$, $\alpha = 0.75$ and the function

$$f(t) = t^2 + \frac{2t^{2-\alpha}}{\Gamma(3-\alpha)}$$

The Table 4.1 presents the execution time for our simulations depending on the value of h that splits the $[0, 1]$ interval.

The Diethelm method takes almost 4 times longer than ABM method. It can be seen that having more points to compute, the differences in the computation times increases. These big differences are caused by the fact that computation for the weights are much more complicated in Diethelm method. The weights are computed every time, at each step, for all the coefficients. In ABM method the weights are computed upfront, as they are fixed for each step, so the weights can be stored in global memory and just reused at each step.

The equation has the exact solution of $y(t) = t^2$. At first glance, in the Fig 4.2 the approximation of the solution with both methods show good results.

On a further investigation, performing a zoom and having the step size h with a smaller value, we can observe the Adams-Bashforth-Moulton predictor corrector is more accurate. It is much closer to the exact solution than the Diethelm's method as it is visible in Fig 4.3.

5. Conclusions and future work. The Algorithm 1 for numerical simulations proved to be a generic algorithm. It was implemented and easy to adapt to run for any input functions, even functions in multiple dimensions [7, 8].

Although at first it was design to use the Adams-Bashforth-Moulton predictor corrector, in this paper it was demonstrated that the Algorithm 1 can be applied to use other numerical methods.

Having at the core the same algorithm, it is the perfect framework to make the comparison between different numerical method. We had proven that ABM method is better than Diethelm's method, both by the criteria of execution time and accuracy of the estimated solution.

The algorithm can be adapt to other numerical methods. As the core of the algorithm is to compute efficiently large number of sums with large number of terms, it can be incorporated in running simulations based on other numerical methods. For example, Lubich's fractional linear multi-step method can be implemented with this algorithm and more comparison can be done against ABM method and Diethelm's method.

TABLE 4.1
Simulation results in seconds for different numbers of time steps

#steps	ABM	Diethelm	Ratio
100000	18.406	27.311	1 : 1.483
500000	131.988	357.034	1 : 2.705
1000000	372.181	1254.215	1 : 3.369
1500000	691.348	2667.044	1 : 3.857

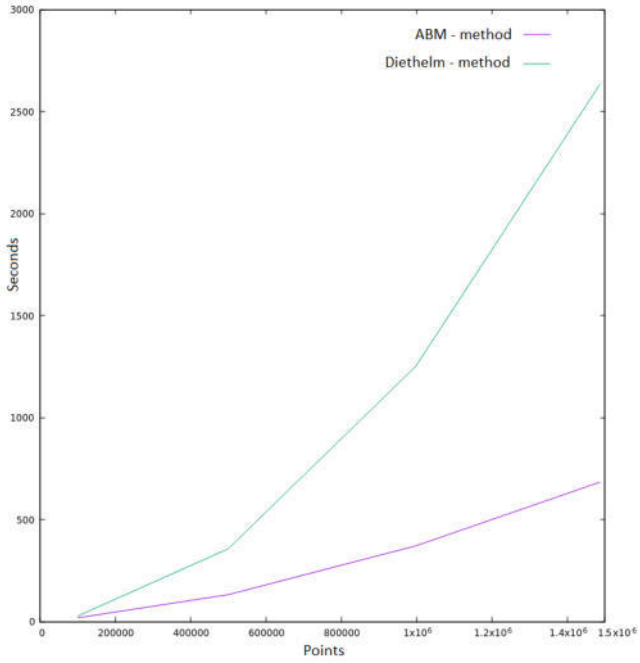


FIG. 4.1. *Execution times*

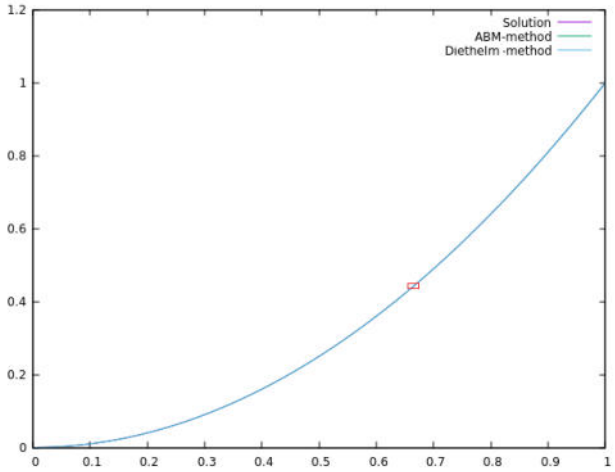


FIG. 4.2. *Aproximated solution with mark for zoom*

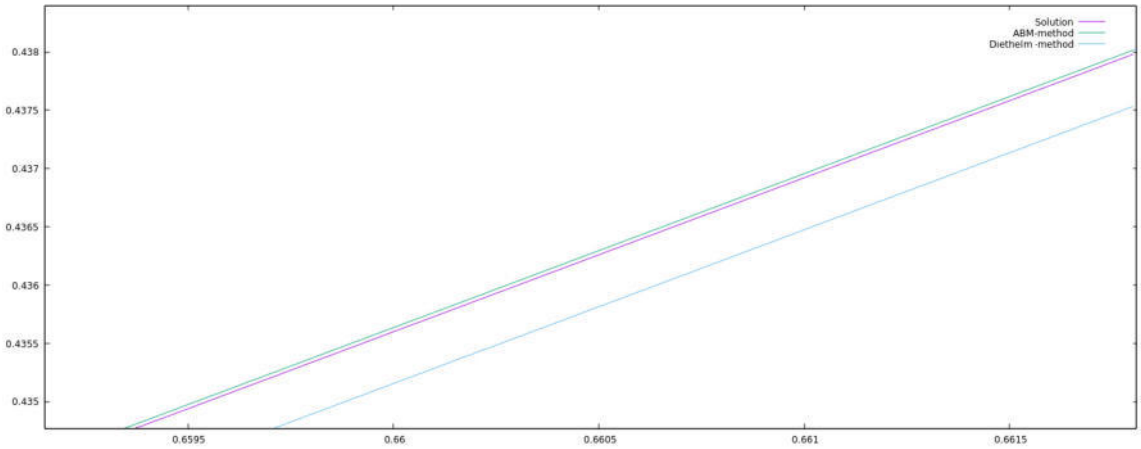


FIG. 4.3. *Aproximated solution zoom region*

REFERENCES

[1] G. COTTONE, M. DI PAOLA, R. SANTORO, *A novel exact representation of stationary colored Gaussian processes (fractional differential approach)*,Journal of Physics A: Mathematical and Theoretical, Volume 43, Page 085002, 2010
[2] K. DIETHELM, N.J. FORD, AND A.D. FREED, *A predictor-corrector approach for the numerical solution of fractional differential equations*. Nonlinear Dynamics, volume 29(1-4), pages 3-22, 2002.
[3] R. GARRAPPA, *On linear stability of predictor–corrector algorithms for fractional differential equations*, International Journal of Computer Mathematics, volume 87, pages 2281-2290, 2010
[4] W. ZHANG, X. CAI, *Efficient implementations of the Adams-Bashforth-Moulton method for solving fractional differential equations* 2012
[5] N.E. BANKS, *Insights from the parallel implementation of efficient algorithms for the fractional calculus*, PhD Thesis University of Chester, United Kingdom 2015
[6] L. GALEONE, R. GARRAPPA, *Explicit methods for fractional differential equations and their stability properties, Journal of Computa- tional and Applied Mathematics*, volume 228(2), pages 548-560, 2009.
[7] C. BONCHIŞ, E. KASLIK, F. ROŞU, *HPC optimal parallel communication algorithm for the simulation of fractional-order systems*, Journal of Supercomputing, volume 75, pages 1014–1025, 2019
[8] F. ROŞU, C. BONCHIŞ, E. KASLIK, *Numerical simulation algorithm for fractional-order systems implemented in CUDA*, SYNASC, 2020, to be published
[9] NVIDIA Corporation, CUDA Programming Guide Version 3.1, 2010.
[10] M. HARRIS, *Optimizing parallel reduction in cuda* URL: http://developer.download.nvidia.com/compute/cuda/1_1/Website/projects/reduction/doc/reduction.pdf, 2007
[11] C. LI, W. DENG, *Remarks on fractional derivatives*, Applied Mathematics and Computation, volume 187(2),777– 784, 2007
[12] D. CAFAGNA, G. GRASSI, *On the simplest fractional-order memristor-based chaotic system*, Nonlinear Dynam. 70 (2012) 1185–1197.
[13] K. DIETHELM, *An algorithm for the numerical solution of differential equations of fractional order*, Electronic Transactions on Numerical Analysis, volume 5, pages 1-6, year 1997
[14] N. HEYMANS, J.-C. BAUWENS, *Fractal rheological models and fractional differential equations for viscoelastic behavior*, heologica Acta 33 (1994), pages 210-219
[15] K. DIETHELM, *The Analysis of Fractional Differential Equations: An application-Orientated Exposition Using Differential Operators of Caputo Type* Springer, 2010
[16] K. DIETHELM, *An efficient parallel algorithm for the numerical solution of fractional differential equations*,Fractional Calculus and Applied Analysis 14 (2011), 475-490
[17] L. SONG, S. XU AND J. YANG, *Dynamical models of happiness with fractional order, Communications in Nonlinear Science and Numerical Simulation*, volume 15, pages 616-628, 2010
[18] V. GEJJI, Y. SUKALE, S. BHALEKAR, *A new predictor–corrector method for fractional differential equations*, Applied Mathematics and Computation, volume 244, 2014

Edited by: Viorel Negru
Received: Nov 18, 2020
Accepted: Dec 18, 2020


Scalable Computing: Practice and Experience, ISSN 1895-1767, http://www.scpe.org
© 2020 SCPE. Volume 21, Issues 4, pp. 709–725, DOI 10.12694:/scpe.v21i4.1838

DECENTRALIZED AND FAULT TOLERANT CLOUD SERVICE ORCHESTRATION

ADRIAN SPĂTARU*

Abstract. This paper proposes a decentralized framework for the orchestration of Cloud Services using heterogeneous resources residing in the homes of private individuals or small-scale clusters. The framework makes use of Ethereum Smart Contracts to provide a decentralized mechanism for discovering the different interfaces exposed by Cloud Components. The paper introduces a novel concept of Component Administration Networks, which are peer-to-peer networks that monitor and ensure the availability of the software components. The concept applied for the Orchestration process to ensure that the deployment of an Application continues in the presence of Orchestrator component failure. Checkpoints are used to address the continuity of the Management components, in general, and of the Orchestrator, in particular. In our proposal, checkpoint metadata is stored in a Smart Contract to assess the execution time of a Service to reimburse the participants that ensure its execution.

Key words: Blockchain, Decentralized Cloud, Service Orchestration

AMS subject classifications. 68M14,68M15

1. Introduction. Cloud Service Providers make use of warehouse-scale computers [1, 2] to provide Infrastructure and Services to businesses and entrepreneurs to accelerate the implementation and deployment of Cloud Applications. The advancement of the Internet of Things field has introduced new challenges concerning the bandwidth required for massive data transfers between the originator (residing at the end of the network) and processing services (running in the Cloud data centres). Fog Computing tackles the movement of Cloud Services closer to the data source. Such systems are generally hierarchical, each level processing data from an underlying level and sending it to an upper level, reaching the Cloud at its top. This reduces the amount of bandwidth necessary for transferring the data. Nevertheless, the cost of implementing Fog Networks is high in terms of both investment and maintenance.

We reckon that peer to peer networks of personal computers and small-scale private Clouds can participate in a Decentralized Cloud Platform, able to provide the resources required to execute a Cloud Service closer to the Service consumers. For example, a 3D artist can run a rendering application, as a Service, on a computer from his/her immediate vicinity. More complex services can be delivered to an area of consumers close to a local Cloud, and services executing on multiple local Clouds can discover and synchronize with each other. A vast amount of investigation has been pursued with respect to peer to peer systems consensus: [3, 4, 5, 6, 7], file sharing services such as IPFS [8], Kademlia [9], BitTorrent [10], and volunteer computing applications such as BOINC [11], XtremWeb [12] as well as dealing with saboteur nodes [13].

Blockchain technologies, such as Bitcoin [14] and Ethereum [15], have increased the number of peers willing to participate in globally distributed networks of computers, providing an economic incentive to maintain and extend a global replicated state machine. The Ethereum Virtual Machine makes transitions using a quasi-Turing complete instruction set, which can be used to define Smart Contracts comprising of arbitrary code. In order to limit the misuse of the platform, each instruction has a cost expressed in gas, out of which the data-store instruction has the highest cost. This renders unfeasible any attempt to use the Blockchain as a Storage Service but is a guard against infinite execution. Instead, Smart Contracts are intended to contain just the business logic of an application. The state machine is updated by transactions which are organized into blocks on a Blockchain. This in turn requires each node participating in the network to execute all state updates from the transactions in a block locally, which hinders the performance of the system. The advantage

*Department of Computer Science, West University of Timişoara. (adrian.spataru@e-uvt.ro). Questions, comments, or corrections to this document may be directed to this email address.

is that each node can read from the state machine locally. At the time of this research, a number of 1,382,198 nodes[1] have been historically registered with the network, out of which at least 10,000 nodes (the maximum shown by the Etherscan interface) are active daily.

Several decentralized platforms have been developed to make use of the Ethereum Blockchain and the power of Smart Contracts. FileCoin [16] offers a peer-to-peer storage service based on IPFS, using Ethereum for payments and access management. Data storage is verified using a Proof of Replication [17] mechanism which validates that a node has dedicated space to hold file blocks under a given replication requirement. Other Blockchain solutions have been investigated in the direction of Autonomous Vehicles cooperation [18, 19] and Internet of Things cooperation [20, 21]. By using a blockchain, the decentralized platform offers trust guarantees concerning the correctness of the Smart Contract execution. Additionally, each state transition can be audited, allowing for a transparent decision-making process.

Nevertheless, peer-to-peer systems (especially networks formed by personal computers) have always suffered from a lack of predictability with respect to the availability of the participating nodes. This is not a limiting factor in the case of a Blockchain system, where nodes are used only to store the blocks of transactions. However, when a node is assigned to run a Cloud Service, then this node should remain available for the execution of this Service. In case of failure, high-availability of the Service can be ensured using replication, yet a more desirable solution is to also restart the failed Service. Moreover, if the system is envisioned to be fully decentralized, then high-availability and fault tolerance should also be ensured for the Cloud Management Components.

This paper enhances an existing architecture (developed in the framework of the CloudLightning Project [22]) to allow for the decentralized management of a public Cloud platform that aggregates resources owned by private individuals. The enhanced platform ensures the Continuity of Cloud Applications in presence of Service Failures and Continuity of the Cloud Platform in presence of Management Component failures. Moreover, we provide mechanisms to ensure that a fair price is calculated based on the time that a node has dedicated to executing a Cloud Service. In summary, this paper provides the following key contributions:

- an architecture that allows for the decentralization of the Components of the Cloud Platform and thus the resource registration and assignment mechanisms, using Smart Contracts
- the concept of Component Administration Networks together with corresponding protocols; these networks provide a bridge from the Smart Contract World and the Software World and ensure the progress and continuity of Management Components by assigning them work, saving checkpoints, and monitoring their availability.
- a mechanism allowing for fault-tolerant Application Orchestration which makes use of the Component Administration Networks; this mechanism ensures that the Services composing an application can continue to be deployed in the presence of an Orchestrator failure and the user is taxed only for the amount of time a Service has executed;

The rest of the paper is structured as follows. Section 2 recalls the CloudLightning Architecture and presents its main features. Section 3, presents our architecture that offers decentralized management. Section 4 defines the concept of Component Administration Networks and investigates its behaviour through simulations of different failure types and rates. In Section 5 we use this concept to define the Application Orchestration mechanism which tracks the execution and computes the price based on the observed execution. At last, conclusions are presented in Section 7.

2. CloudLightning Architecture. Figure 2.1 highlights the principal components of the CloudLightning Architecture focusing on the three layers.

The **Gateway Service** offers user-facing functions such as maintaining a catalogue of Services that can be composed into an Application using an intuitive **User Interface** (UI). Services and Applications are defined using the Topology and Orchestration Specification for Cloud Applications (TOSCA). CloudLightning Services improve the flexibility of the Cloud User by allowing him to describe an Application composed of Abstract Services which have multiple hardware-specific implementations. The **Service Optimization Engine** is a component that reads this description and contacts the Resource Manager Layer to choose the best-suited implementation for each Abstract Service, depending on the requirements set by the user (performance, cost)

[1] https://etherscan.io/nodetracker/nodes, accessed 16 Nov 2020

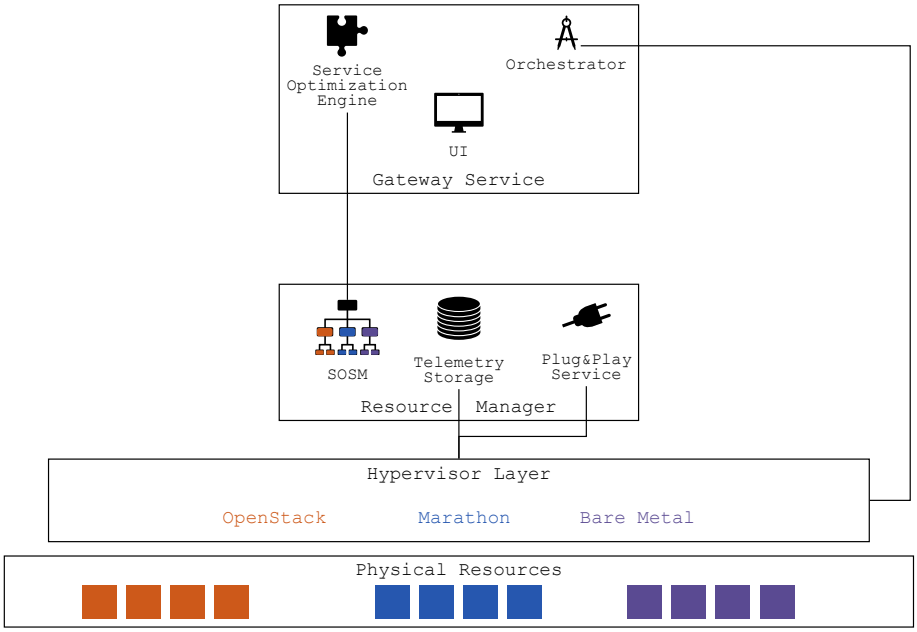


FIG. 2.1. *CloudLightning Components Architecture*

and the availability of the System. This mechanism also improves the flexibility of the Cloud Provider, by allowing him to offer a more efficient resource type like Graphical Processing Units (GPUs) or Many Integrated Cores (MIC) cards, in order to reduce its energy costs. Once the resources are selected, the Abstract Services are replaced by their Concrete Implementation in the Application definition. Finally, the **Orchestrator** component reads a Topology composed of Concrete Services and deploys it, taking into account any dependencies. It then monitors the deployment and execution of the Services, being able to restart a Service in case of failure.

On the bottom layer, physical resources are under the control of a **Hypervisor**, achieving different types of abstractions: Virtual Machines or Containers. The Hypervisor head node needs to contact the Plug&Play component to register with the **Self Organizing Self Managing** (SOSM) Resource Management System. Additionally, a telemetry client must be installed to send data to the Telemetry Service (a time-series database). The Hypervisor APIs are used to create VM and Container Images and to start Instances. Bare Metal servers can also be managed by the creation of accounts when this type of resource is selected. Further configuration and execution are managed using bash scripts.

The SOSM System [23] is a hierarchical resource management system that aggregates monitoring information collected at the bottom level and assesses the efficiency and performance of resources based on some specific business objectives (e.g., energy efficiency, computational performance). The assessment functions are weighted based on some importance for each aspect and propagated up the hierarchy. Each component of the hierarchy is described by a Suitability Index (with respect to the desired state of the system) and can be used to guide the Service requests to reach resources that are more energy or computationally efficient. The system behaviour has been investigated using Large Scale Simulations [24] which show that this system is able to optimize the resource utilization and performance metrics of data centres of over $100,000$ servers.

A self-healing implementation of the SOSM system has been presented in [25]. Each component is replicated and has either the role of a controller or a replica. The controller ensures enough replicas are available to cope with the current load of the system. A Message Queuing System is employed to route the messages between the different layers of the systems, ensuring only one replica is processing a request for resources. The replicas broadcast heartbeat messages to all corresponding replicas and construct a list with their startup time and status. A leader failure is detected if enough time has passed until a heartbeat was received. Then, the eldest replica will take the role of the controller and inform the rest.

The Orchestrator can detect the failure of a Cloud Service and issue the restart process. If any Services were dependent on the failed Service, then they should be reconfigured with the new properties (i.e., endpoints). Virtual Machines and Containers should use data volumes, to avoid losing progress if restarted. The Client can ensure the high availability and load balancing of its Application by requesting multiple instances of the same Service.

3. Decentralization of Cloud Components. In a previous paper, [26], we have proposed mechanisms for Decentralized Scheduling of Cloud Services using Smart Contracts and investigated the operational constraints and cost of such a system. Our investigation has revealed a high transaction cost when encoding the scheduling optimization algorithm in the Smart Contract. Moreover, the high cost reduces the number of transactions that can fit in a block, decreasing the transaction throughput and thus increasing latency for sealing a resource assignment.

A more efficient solution is to allow the Cloud Users to read the state of the Smart Contract and apply the resource selection algorithm locally. Once a resource is selected, a function is called in the Smart Contract to seal this assignment. Our investigation on a real-world Cloud scenario has shown that the cost for this method is 1/5 of the previously mentioned cost. Unfortunately, several users can select the same resource at the same time, which leads to conflicts and thus rejected transactions. Overall, this leads to the latency of the two methods to be comparable. This result leads to the necessity for a component that synchronizes the decisions made by users, such that no conflicting transactions are sent to the Smart Contract.

The modular architecture of the CloudLightning system allows us to decouple several components, paving the way to the decentralization of control and fault tolerance of the management components. Figure 3.1 depicts our proposed architecture. A public Blockchain able to execute Smart Contracts is used to synchronize the knowledge about the available components.

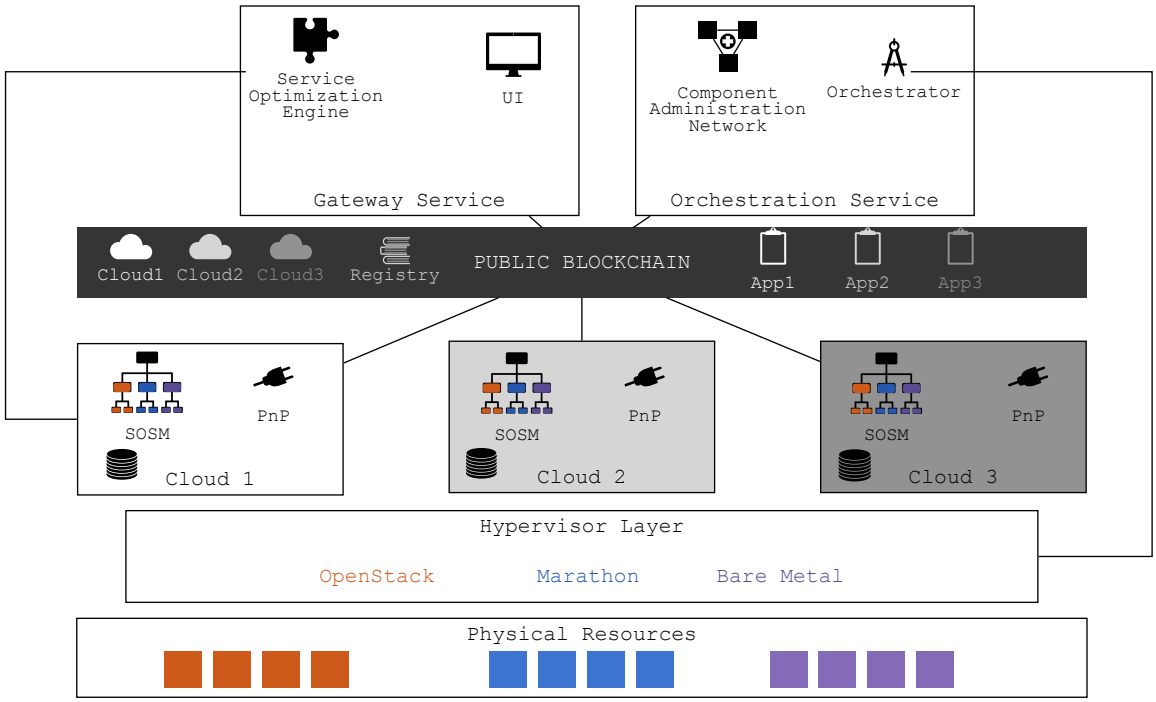


FIG. 3.1. *Augmented Decentralized Architecture*

In order to allow for the discovery of the decentralized components, we use three Smart Contracts:

1. **Registry Contract** – the main contract for the system. It is used to register Cloud Providers, Services and Applications. Additionally, it is used to maintain the contact information and availability of the

nodes taking part in the Component Administration Network.
2. **Cloud Contract** – holds resources descriptions and price, as well as the endpoints for the Scheduler and Plug&Play components.
3. **Application Contract** – created for each Application in order to track deployment status and payments. Checkpoints metadata is stored in this contract, which serves as proof for computing payment to the entities involved in the execution.

In this augmented architecture, the Gateway Service is a client-side application. It can read the state of the Registry Contract from local storage (if the client also runs an Ethereum node) or using a remote REST call (if the client only runs the Gateway Service). From the Registry state, the client can read the available Service and Applications definitions, as well as the registered Clouds. The Service Optimization Engine is able to reach the Cloud Scheduler endpoint in order to ask for resources, by reading the information present in the Cloud Contract selected by the client. A Cloud Contract can be backed by a SOSM Resource Management System or any other Scheduler that implements the Blueprint-based Resource Discovery Protocol described in [28, Ch. 4.1]. The Gateway Service is thus decentralized and multiple instances of this Component are synchronized through the Registry Contract and Cloud Contracts.

The Orchestration process is decoupled from the Gateway. This process is distributed over several nodes that are coordinated by a Component Administration Network (CAN). Instead of mining, a subset of the Blockchain-participating nodes can be part of a CAN and be rewarded for monitoring and storing Management Components states. Component Administration Networks are further detailed in Section 4.

Resource owners are expected to use a thin client that allows them to read the content of the Registry Contract and learn about any available Cloud Contracts. The owner can then contact the Plug&Play Service which, after validation, registers the physical resources with the SOSM system and updates the Cloud Contract.

3.1. System Initialization. An Ethereum Blockchain Network is assumed to work as an external component to our proposed system. To achieve better latency and not suffer from the Ether crypto-currency volatility, we reckon the best practice is to create a new network, used only for this purpose. The system initiator can dedicate some nodes to bootstrap this chain and extend the public ledger. Additionally, the system initiator must dedicate some nodes to bootstrap the Orchestration Component Administration Network (OCAN).

The following steps are required to initialize the system:

1. Deployment of Registry Contract. This is initialized with the address of the owner, an empty list of OCAN candidates and a null value assigned for the OCAN leader.
2. Component Administration Network bootstrap:
   i) Registration of nodes as CAN candidates. Each node is associated with an Ethereum account
   ii) The system initiator selects a node from the CAN candidates to be the leader of the OCAN
   iii) The selected OCAN leader inserts its contact information (e.g., IP, port) in the Registry contract
   iv) Candidate nodes learn the contact information of the leader and contact it to join the OCAN.
   v) Candidate nodes get verified as Orchestration nodes by the OCAN leader.
3. Services and Applications are registered in the Registry Contract using the TOSCA specification.
4. A Cloud Service Provider calls the Registry function to deploy a new Cloud Contract, providing the endpoints of the Scheduler and Plug&Play components. Additionally, a Provider should also provide the public key of the Scheduler, used to sign the reservation for resources to a client.
5. Resources can be registered with the Cloud Contract:
   i) The resource owner contacts the Plug&Play Service defined by a Cloud Contract which he/she wishes to join.
   ii) The Plug and Play Service verifies the Hypervisor API and telemetry endpoints and decides to accept or reject
   iii) If accepted, the resource is added to the management system and registered in the Cloud Contract. Note that resources do not need to run an Ethereum node. They are registered and removed from the Cloud Contract memory by the Plug&Play component.

Steps 2, 3, and 4 do not depend on one another and can be executed in parallel. The resource manager is updated by the Hypervisors with telemetry data periodically. If the data cannot be obtained from the registered resources, it can inform the Plug&Play Service to deregister it.

3.2. Application Deployment. An Application is defined using the TOSCA Topology specification. The Client creates an Application definition using the UI and registers it in the Registry Contract before deployment. Alternatively, Application definitions registered by others can be referenced by id.

An application is deployed in the following steps:

1. The User selects an Application and a Cloud. A Blueprint is generated by the Service Optimization Engine and sent to the Scheduler for assigning the resources and choosing the concrete implementation in case of Abstract Services. In the successful case, the Scheduler will reserve the resources for a given amount of time for this Application. The user is returned a signed message encompassing this reservation. The user can accept the implementation and selected resources, or he/she can ask for their release (such that the resources can be reserved to someone else before the expiration of this reservation).
2. To instantiate the Application, the user calls the corresponding function in the Cloud Contract, providing the signed message from the Scheduler. The Application Contract is deployed if and only if the reservation has not expired and none of the resources has been deregistered in the meantime.
3. The OCAN leader is informed that a new Application Contract has been deployed. This step can be achieved by the Gateway or by the use of Ethereum events, with the OCAN leader listening for new Application Contracts.
4. An Orchestrator is selected by the OCAN leader to be in charge of the Application deployment; this decision is sent to the OCAN nodes to be recorded on the ledger.
5. The Gateway is now able to query any OCAN node to learn about the managing Orchestrator. This Orchestrator is contacted for information about the runtime properties of the Services under deployment.

Several failures can occur throughout the lifetime of the proposed system. *Node failures* refer to the state when a node becomes unavailable, either due to hardware problems or network disruptions. This can, in turn, generate *Management Component failures* if the node is running a Management Component or *Service Failures* if the node is a physical resource executing a Service. Our system cannot replace a failed hardware component or resolve networking disruptions but mitigates their effect by enforcing fault-tolerance at the Management Component level and Application deployment continuity at the Orchestration level.

4. Component Administration Networks. A Component Administration Network has a two-fold purpose. First, it bridges together the land of Smart Contracts with the land of Software Components. Second, it provides the means for monitoring and enforcing a set of replicas in order to tolerate faults. The network of nodes implements a replicated state machine which has two functions. First, it maintains a ledger of transactions related to the network of nodes and the supervised components, different than the Ethereum ledger. Second, the nodes execute a replicated file system to store data associated with the supervised components.

Figure 4.1 presents the layered architecture of this proposal. On the bottom layer, there is the peer to peer network that collaborates for maintaining the Ethereum Blockchain, where the Registry Contract is deployed. Some of these nodes can be part of the Component Administration Network.

The middle layer is concerned with operations for managing the nodes. This is the first layer where we identify the leader of the network, which is responsible for ordering and validating all transactions related to the CAN. The replica nodes will accept any state update from the leader. For this layer we propose the following protocols:

- Join – protocol for a new node to join the network
- RemoveNode – protocol for removing a node that has been discovered to be faulty
- LeadElect – protocol for electing a new leader if the current one has been discovered to be faulty

The top layer is concerned with the administration of Components. Again, the CAN leader is responsible for ordering and validating state updates at this level. This layer is concerned with the following protocols:

- Register – a component gets registered with the network
- Deregister – a component has been unresponsive and is removed
- AssignWork – a component is assigned work
- CheckpointWork – a component requests the network to store a Checkpoint
- ReassignWork – a component has been removed and work is reassigned to another

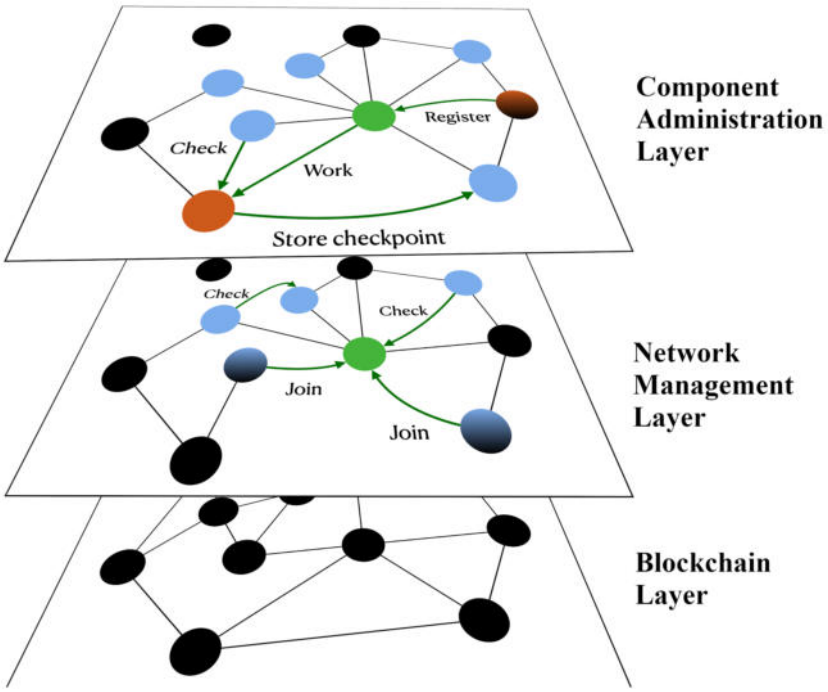


FIG. 4.1. *Layered architecture of a Component Administration Network*

- FinishWork – a component is requested to terminate the execution of a given piece of work (termination of a Cloud Application).

4.1. Network Management Protocols. The network makes use of the Registry Contract to know who is the current leader and what nodes are registered with the network, in order to reward them for their work. Nodes that want to participate in this network must first register in the Registry Contract as candidate nodes. When the system is initialized, the Registry owner selects one of the candidate nodes as the leader of the Network. When the contact information of the leader is known, other candidates can proceed to join the network by contacting the CAN leader.

4.1.1. Joining a network. The protocol for joining a network is presented in Algorithm 1. The Registry owner must set N_{min}, a minimum number of replicas required for the network. The Registry owner must also set N_{max} the maximum number of nodes that can be part of the network. The purpose of N_{max} is to limit the slowdown of the network due to synchronization and data transfers. The *read* function represents a query to the Smart Contract to read specific properties.

A candidate node will attempt to join the network if the size of the network is less than its maximum size. Any candidate node that contacts the leader will be accepted until the minimum number of replicas is reached. After this point, the leader will accept new replicas only if the *churn rate* is greater than the current surplus of nodes (network size minus N_{min}). The churn rate is the difference between nodes that left the network and nodes that joined the network. This is an adaptive mechanism that allows the network to grow at the same pace as nodes are crashing. If a node is accepted, then this decision is broadcast across the nodes and the node's contact information is registered in the Registry Contract.

We analyze the dynamics of this network in terms of its size with different properties. We design an experimental simulation consisting of 100 steps, executing the following at each step:

1. A candidate node will request to join the network with probability $j \in \{.2, .1, .05, .025, .01\}$
2. A CAN node will crash with probability $f \in \{.5, .75\}$. Nodes joining in this step can also crash in this same step.

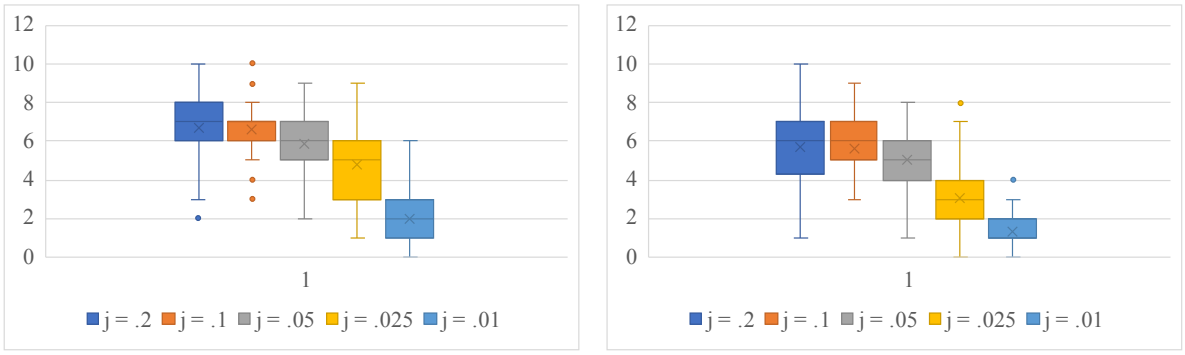
First we consider a network with $N_{min} = 4$, $N_{max} = 10$. We consider $N_{total} = 100$, the total number of

Algorithm 1 Protocol for joining CAN

```
current ← read(Registry, CAN.size)
max ← read(Registry, CAN.max)
leader ← read(Registry, CAN.leader)
function askJoin
    if current < max then
        sendJoinRequest(leader, contactInfo)
    end if
end function
function listenJoin(contactInfo)
    if current = max or checkAvailable(contactInfo) = false then
        return reject
    else
        if current > min then
            if current − min > churn then
                return reject
            end if
        end if
    end if
    broadcast_node_join(peers, props)
    registerPeer(SC, props)
    return accept
end function
```

participating nodes and $C = N_{total} - N_{current}$, the number of candidate nodes. When the network is started, the leader is appointed by the Registry owner.

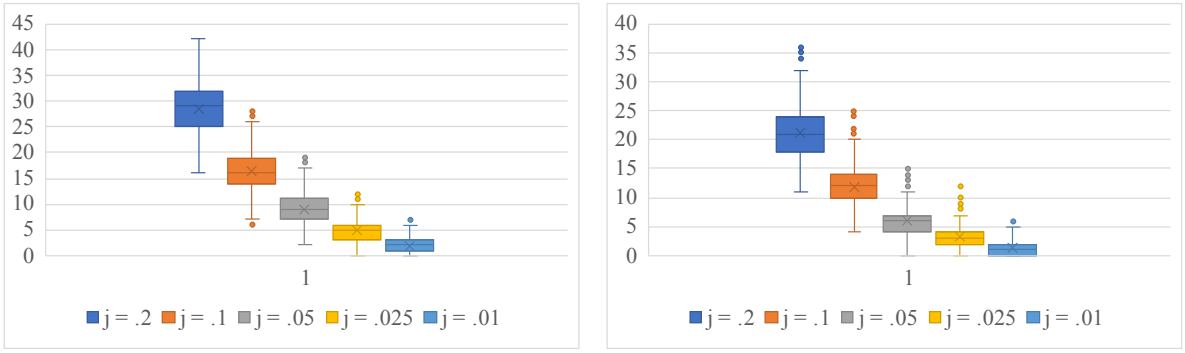
Figure 4.2 shows the size of the network using box plots. Figure 4.2.a represents the size dynamic when $f = .5$, while Figure 4.2.b represents the size dynamic when $f = .75$. For $f = .5$, results indicate that the joining probability of the nodes should be at least 0.025 in each step in order to have, on average, a network of the minimum desired size. If $j = 0.01$ the average size of the network is 2, reaching states where the size is 0. All other joining probabilities reach states where the size of the network is smaller than the minimum desired size. For $f = .75$, if $j \in .01, .025$ then the failure of the network is guaranteed. Other joining probabilities have an average size larger than the minimum but still reach states where the size is smaller than the minimum.



(a) $f = .5$ (b) $f = .75$

FIG. 4.2. *Size of a CAN for different join and failure probabilities for* $N_{max} = 10$

We repeat the experimental simulation, with N_{max} set to 100, with the results presented in Figure 4.3. Although the size of 100 is never reached, the network benefits by having a greater tolerance to node failures.



(a) $f = .5$ (b) $f = .75$

FIG. 4.3. *Size of a CAN for different join and failure probabilities for* $N_{max} = 100$

For $f = .5$, we observe that the candidate nodes must be willing to join with probability $j > 0.1$ in order for the network to maintain a size greater than the minimum desired. For $f = .75$, results are similar. Given the simulation results, Algorithm 1 maintains the network operational if a large enough limit, N_{max}, is set and if sufficient new nodes are willing to join the network. In our simulation observed that for $N_{max} = 100$ and $j = .1$ the network is able to maintain a minimum desired size in presence of failures with probability $f = .75$, although we do not expect probabilities of this size in practice.

4.1.2. Node Removal. We propose a mechanism for checking the state of the nodes in the network. All nodes, with the exception of the leader, will execute the protocol described by function *checkState* in Algorithm 2. A node randomly selects another node of the network, including the leader, to check its status. If the node responds, then the function stops. If the node does not respond within a given timeout, then this node is considered in a failed state. If the node is a leader, then a Leader Election process is started; this process is described in the following subsection. If the node is a normal replica, then a request to remove the node is sent to the network leader.

Algorithm 2 Protocol for removing a CAN node

```
1: procedure CHECKSTATE
2:     node ← randomSelection(peers)
3:     status ← ping(node)
4:     if status ≠ OK then
5:         if node ≠ leader then
6:             sendRemoveRequest(leader, node)
7:         else
8:             leadElect(peers, self)
9:         end if
10:    end if
11: end procedure
1: procedure LISTENREMOVE(R)
2:     v ← collectLeaveVotes(R)
3:     if count(v, accept) > current/2 then
4:         broadcast_node_leave(peers, R)
5:         deregisterPeer(Registry, R)
6:     end if
7: end procedure
```

When receiving a Removal Request for node R, the leader holds a vote to remove this node. This step

is necessary because the proposer node might be the only one seeing R as unavailable (e.g., the proposer experiences a network disruption). If a simple majority of the network considers R to be unavailable, then this node is removed from the network. This decision is broadcast to the network and the contact information for node R is removed from the Registry Contract.

4.1.3. Leader failure and election. A leader failure is detected by the periodical execution of function *checkState*. Algorithm 3 presents the protocol for being selected as the leader. We use the terminology from the Paxos Algorithm [27] and consider the nodes which detect the failure of the leader as *Proposers*. The remaining nodes behave like *Acceptors*. Acceptors run the protocol described by Algorithm 4.

Algorithm 3 Procedure for CAN Leader Election

```
1: procedure LEADELECT
2:     n ← read(SC, leaderNonce) + 1
3:     votes ← map(collectLeadPromise(peers, n, self))
4:     if count(v.values, promise) > current/2 then
5:         acc ← [ ]
6:         for p in peers do
7:             if v[p] = promise then
8:                 v ← sendLeadAccept(p, n, self)
9:                 acc.append(v)
10:            end if
11:        end for
12:        if count(acc, accept) > current/2 then
13:            receipt ← registerLeader(SC, n, acc, self)
14:            broadcast_leader_change(peers, receipt)
15:        end if
16:    end if
17: end procedure
```

A *nonce* value is maintained in the Registry Contract to be associated with leader election processes. This value is read by a Proposer, which increases its value by 1 and attempts to collect *promises* from its peers. If multiple Proposers exist, then all of them will have the same nonce value. If a Proposer receives a simple majority of promises, it proceeds to the next step. In this step, it will ask all promising nodes for their accept vote. If a simple majority of final votes is gathered, then this node is the new leader. The new leader must update the Registry Contract, providing the signed accept messages. The Smart Contract checks the signatures, and if all are valid then it updates the nonce value and the leader contact information. The new leader broadcasts a receipt of this update to the nodes, which can read the new state of the Registry.

Algorithm 4 Protocol for promising to a new CAN Leader

```
1: function LISTENLEADELECT(n, node)
2:     nonce ← max(nonce, read(Registry, leaderNonce))
3:     if n ≤ nonce then
4:         return reject
5:     end if
6:     if ping(leader) ≠ OK then
7:         nonce ← n
8:         return promise
9:     end if
10:    return reject
11: end function
```

When receiving a *Promise* request from a peer, a node will first compare the local nonce value with the

one written in the Registry Contract and will update its local value with the maximum between the two. If the proposed nonce, n, is not greater than local nonce, then the promise is rejected. If the leader is indeed unavailable then the local nonce is updated and a promise is returned. When receiving an *Accept* request, the node will return a signed Accept message only if it has already promised to the requester using this nonce.

The reason this protocol takes place outside the Registry Contract is to limit the number of Ethereum transaction required to elect a new leader and thus reducing the recovery time for the CAN. If the protocol took place in the Smart Contract, a minimum number of $n/2 + 1$ Smart Contract calls need to be issued, one for each vote. In our proposed protocol the votes are cast at the Network Management Layer and only one function call is made to the Smart Contract.

4.2. Component Administration protocols. Components register with a CAN by contacting the network leader providing an endpoint where the software is reachable. If the leader is able to access the endpoint then it decides to register this component and broadcasts a RegisterCompoenent(compID) transaction to update the CAN ledger and inform the network about the new Component which needs monitoring.

The failure of a Component is detected using the *checkState* function. Besides checking the state of a randomly selected peer, a node will also check the state of a randomly selected Component. If a Component is considered unavailable, then the node will ask the leader to remove this Component. The leader proceeds by holding a vote, similar to the vote for removing a node. If a component had any work assigned, this will have to be reassigned to another component, together with all the checkpoints that the failed Component was able to generate.

4.2.1. Work assignment. In general, a Work Entry is characterized by an identifier and a description. In our case, the identifier is the address of an Application Contract, and the description is a TOSCA Topology. A client can inform the CAN leader about a work entry, which in turn will select a component to execute this work, for example using the Round Robin method. This is achieved in two steps:

1. The leader contacts the selected component which validates the work entry and returns an Accept message.
2. The leader will update the CAN ledger with a *AssignWork(workID, compID)* transaction, where *workID* is the Work Entry identifier and *compID* is the identifier of the component

A component can save checkpoints during the execution of the Work Entry. This is done by contacting the leader to store the checkpoint information (identified by the digest of the data) on the CAN storage. This is done in two steps:

1. The leader will select a peer as the initial storage node for this checkpoint and return its contact information to the component. The component can start uploading the checkpoint information on this initial node.
2. The leader issues a *CheckpointWork(workID, compID, dig(cp))* transaction, which tells the peers to start replication for the checkpoint identified by digest $dig(cp)$.

In the event of a Component removal because of failure, the leader will reassign any Work Entries to new Components. After the selection of a new Component which validates and accepts the corresponding Work Entry, the leader updates the CAN ledger by broadcasting a *ReassignWork(workID, compID, cpa)* transaction to the network. The newly assigned component can retrieve the data corresponding to the digests in the checkpoint array, *cpa*, from any of the CAN nodes.

A Work Entry can be stopped and its checkpoint information can be removed from the replicated storage system. At first, the leader will inform the assigned Component to execute any shut-down instructions encapsulated in the Work Entry description; a signal will be returned when finished. Finally, the leader issues a *FinishWork(workID)* transaction which informs the peers to remove any checkpoint entries associated with the given Work Entry id.

5. Fault tolerant Orchestration. Given the protocols described by the previous section, we use a CAN to manage the Orchestration process. Orchestrators are registered with the Orchestration Component Administration Network (OCAN) as Components, which execute Work Entries described by Application Contracts.

The leader of this network adds the checkpoints metadata to the Application Contract. A checkpoint metadata contains the checkpoint type, a timestamp and an issuer. For the Orchestration process we consider

4 checkpoint types:

- *SERVICE_UP*,
- *SERVICE_DOWN*,
- *APP_OK*,
- *APP_SHUTDOWN*.

These checkpoints are used to compute payment duties to the different entities involved in the Application execution. The leader is responsible to periodically call an Application Contract function that computes the payment and sets funds at the disposal of the entitled entities.

An Orchestrator selected by the leader proceeds with reading the Application Topology and selected infrastructure and starts the deployment of the Services, using the corresponding Hypervisor API. After each successful deployment, the Orchestrator Component will request the storage of a *SERVICE_UP* checkpoint, containing information about the deployment and its monitoring id issued by the Hypervisor. This information is useful to an Orchestrator replica to continue the deployment in case the current Orchestrator fails.

A Service failure can be detected by the Orchestrator during the periodical checking of the state for each deployed Service. If a Service failure is detected, a *SERVICE_DOWN* checkpoint is made to acknowledge the new state and redeployment is attempted. If the Service cannot be redeployed after a given number of retries set in the Application Contract, then the whole Application is shutdown, issuing a *APP_SHUTDOWN* checkpoint. Periodical checkpoints of type *APP_OK* are issued at intervals set in the Application Contract. These checkpoints are issued only if all Services are executing.

In Figure 5.1 we present a sequence diagram that illustrates the continuity of the deployment process in the presence of Orchestrator component failures.

We consider the case of the Ray Tracing Application developed in the CloudLightning Project, which is composed of two Cloud Services, a back-end rendering engine and front-end Web Service depending on the former for high fidelity object rendering. Before the events presented in the sequence diagram, we consider the client has contacted a Cloud Contract for the creation of an Application Contract, using the steps described by Section 3.2. The OCAN leader is contacted by the client (via the Gateway User Interface) to start the deployment of this Application. The OCAN leader selects *Orchestrator 1*, and after work validation it issues an *AssignWork* transaction. The Orchestrator deploys the first Service and stores a *SERVICE_UP* checkpoint. During the periodical monitoring, an OCAN node observes that this Orchestrator has crashed. The leader removes this Orchestrator from the list of managed components and reassigns the work with one checkpoint to Orchestrator 2, which continues the deployment with the second Service, and issues another *SERVICE_UP* checkpoint. If during the periodical inspection a Service has failed, the *SERVICE_DOWN* checkpoint is issued for this Service. Since all Services are available, an *APP_OK* checkpoint is issued.

5.1. Payment. Several entities are ensuring the execution of an Application: The Cloud and its resources, the OCAN nodes, and the Orchestrators; all of them need to be reimbursed. If an Application runs for a long time and a large number of checkpoints are generated, a transaction computing the total payment might run out of gas. Nevertheless, the granularity of checkpoints increases the fairness of the price the client pays. Therefore, we propose a method for interim payments, described by Algorithm 5.

All checkpoints registered by the OCAN leader are stored in the checkpoint array, *cp*. The last paid checkpoint index is stored in variable *lp*, initialized with -1 during Application Contract creation. Payments are accumulated during the life-cycle of an Application in a map represented by the *salary* variable. This approach solves two problems: on one hand, the OCAN leader (which calls the interim payments function) does not need to pay a fee for transferring the funds to the other entities; on the other hand, all entities can decide to withdraw the funds they earned at the end of the execution and only pay the withdraw fee once. The withdraw fee is the base price for executing a funds transfer transaction on the Ethereum Blockchain. The start time (in Unix timestamps) for each service is stored using a map represented by variable *start*.

The function *InterimPayment(k)* in the Application Contract, where k represents the number of checkpoints to be paid can be simulated locally to make sure a large enough value can be set for k (such that fewer transactions are made) without exceeding the gas limit. When called, the variable *newlp* is initialized with the last payment index and is increased for each paid checkpoint, and the variable *total* is set to 0 and is used to check if the total payment duties exceed the currently available funds.

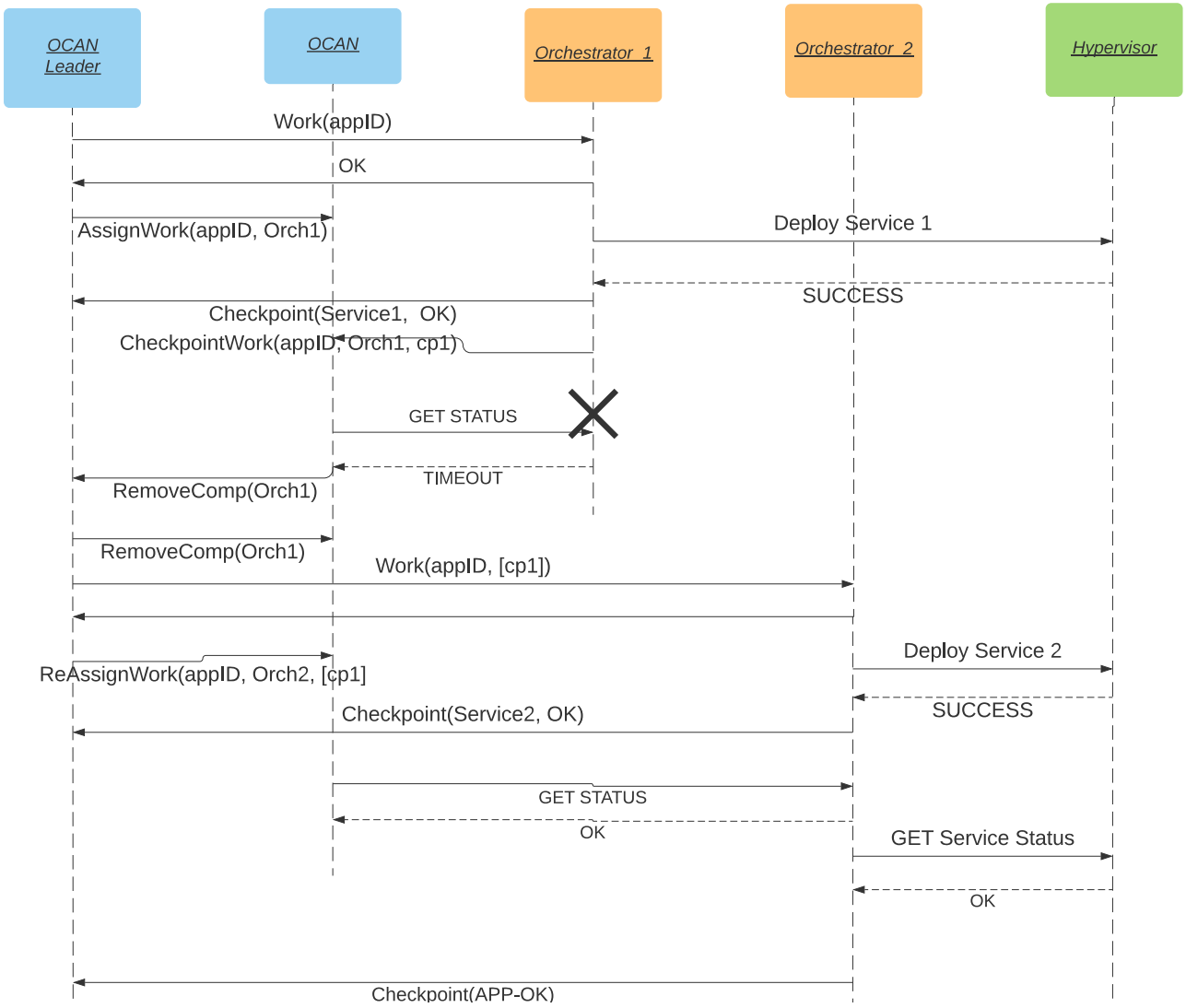


FIG. 5.1. *Example deployment continuity with failing Orchestrator*

The Algorithm starts from the first non-paid checkpoint, and for each checkpoint, the type is inspected. A *SERVICE_UP* checkpoint just sets the start time of a Service. A *SERVICE_DOWN* checkpoint leads to computing the cost for the corresponding Service, and if sufficient funds are available the *salary* map is updated using the *payService* function (described in Algorithm 6). An *APP_OK* checkpoint means all Services are executing and prepares payment for all Services of the Application. If there are not sufficient funds to pay for all the Services, then this checkpoint can not be paid and the loop breaks. In the successful case, the start time of service is updated to be the timestamp of this checkpoint; a further *SERVICE_DOWN* or *APP_OK* checkpoint will be paid in relation to the current one, as the rest has already been paid. Finally, *lp* is updated to the new last paid checkpoint index and the total payable funds are added to a variable, *locked* which restricts the client to withdraw funds that have been promised to be paid. In the end, the total number of paid checkpoints is returned to the OCAN leader. The *APP_SHUTDOWN* checkpoint initiates the same behaviour as the *APP_OK* checkpoint, with the exception that the start time for each service is set to *null*.

The *payService* function computes the reimbursement for each entity and updates the salary map. The Cloud, Physical Resources, and Orchestrators are paid individually, while the OCAN reimbursed trough the

Algorithm 5 Interim Payment Function

```
1: cp – checkpoint array
2: lp – last paid checkpoint
3: salary – map < address, int > with payments
4: start – map < string, int > map with start times per service
5: function interimPayment(k)
6:     newlp ← lp
7:     total ← 0
8:     for i ← lp + 1 to min(lp + k, cp.size) do
9:         c ← cp[i]
10:        if c.type = SERVICE_UP then
11:            start[c.s_name] = c
12:        else
13:            if c.type = SERVICE_DOWN then
14:                x ← computePayment(start[c.s_name], c)
15:                if x + total < getBalance() then
16:                    payService(start[c.s_name], c)
17:                    total+ = x; newlp+ = 1
18:                    start[c.s_name] ← null
19:                else
20:                    break
21:                end if
22:            else
23:                if c.type = APP_OK then
24:                    fail ← False; appTotal ← 0
25:                    for s in services do
26:                        appTotal+ = computePayment(start[s.name], c)
27:                        if total + appTotal > getBalance() then
28:                            fail ← True
29:                            break
30:                        end if
31:                    end for
32:                    if fail then break
33:                    end if
34:                    for s in services do
35:                        payService(start[s.name], c)
36:                        start[s.name] ← c
37:                    end for
38:                    newlp+ = 1
39:                    total+ = appTotal
40:                end if
41:            end if
42:        end if
43:    end for
44:    steps ← newlp − lp; lp ← newlp; locked+ = total
45:    return steps
46: end function
```

Registry contract, which maintains information about the historical availability of each node in the OFTEN. Different Orchestrators can be the contributors of to two consecutive checkpoints; in this case, each of them

gets paid half the orchestration price. In our example, we have considered a per-minute based pricing, and timestamps logged in milliseconds.

Algorithm 6 Service Payment Function

```
1: function payService(cs, ce)
2:     time ← (ce.timestamp − cs.timestamp)/1000/60
3:     res ← Cloud.resourcePrice · time
4:     cloud ← Cloud.price · time
5:     ften ← Registry.ftenPrice · time
6:     orc ← orcPrice · time
7:     salary[Cloud.address]+ = cloud
8:     salary[services[ce.s_name].resAddress]+ = res
9:     salary[RegistryAddress]+ = ften
10:    if ce.orcAddress ≠ cs.orcAddress then
11:        salary[cs.orcAddress]+ = orc/2
12:        salary[ce.orcAddress]+ = orc/2
13:    else
14:        salary[ce.orcAddress]+ = orc
15:    end if
16: end function
```

The *APP_SHUTDOWN* checkpoint can also be issued in the case several interim payments have been tried with 0 successful checkpoints paid. The shutdown checkpoint will trigger the release of the associated resources, marking them as available in the corresponding Cloud Contract. If an Application is in a SHUTDOWN phase, the Application Contract can be destroyed to release space on the Ethereum Blockchain. If salaries have not been collected, the client will support the cost of sending the salaries to the accounts of the entitled entities when requesting the destruction of the Contract. Any unused funds are returned to the client.

6. Related work. Several companies are tackling the offering of Cloud Services through the means of Blockchains, with limited scientific output.

Golem[2] makes use of IPFS [8] to distribute input file blocks in the worker nodes network. Workers are processing data at the block level and all the parallel results will be merged at the user's machine. Compared with them, our proposal is more generic, allowing for user-defined Applications.

SONM[3] achieves a higher level of abstraction, using Docker for executing Container Images. An Ethereum-based side chain is used for managing *Orders*. *Suppliers* interact with the side chain to set up workers acting on their behalf. The workers expose resources such as CPU, RAM, storage, bandwidth in the form of *benchmark identifies*, e.g., 20 *GFLOPS*. A user can rent some resources for a limited amount of time, or on a pay-as-you-go model. There is a limited amount of documentation concerning how the system is matchmaking user requests with resources. In comparison with SONM, our Orchestration process allows for the deployment of Applications as Virtual Machines, Docker Container or Bare Metal infrastructure. Additionally, it supports hardware accelerators (GPUs, MICs) that improve the performance of the Applications. Moreover, this paper proposes a fault tolerance enforcing mechanism for the management Components. SONM does not make clear what mitigation actions are in place for dealing with node failures.

The **iExec**[4] platform is based on renowned of research in volunteer computing [12, 29]. The Ethereum Blockchain is used to manage the platform tokens, and a side chain is used and implement the platform logic. When an application is requested, multiple nodes will execute it and the results are compared. A *Proof of Contribution* (*PoCo*) protocol is used for acknowledging the correct result of an Application, using the sabotage tolerance introduced in [13]. The *PoCo* links two entities: the iExec marketplace (where deals are made) and

[2]https://golem.network/
[3]https://docs.sonm.com/concepts/main-entities
[4]https://iex.ec

the computing infrastructure (based on XtremWeb-HEP middleware [12]). Compared to iExec, our proposal is again more generic. iExec Applications must have a final results, which executing nodes should agree on, while in our proposal each Service should provide a status endpoint which the Orchestrator can check to see if the Service is available.

7. Conclusion. This paper proposed an architecture and mechanism for decentralized Orchestration of Cloud Service on resources residing in homes or small-scale clusters. The proposed framework is able to enforce the fault tolerance of the Orchestration process and to assess the execution time for a Service. The CloudLightning architecture has been conceived to provide efficient and flexible deployment of HPC-aware Services, managing the infrastructure within a data centre. This architecture is augmented to decentralize the Resource Management and Service Orchestration processes, which are synchronized through Ethereum Smart Contracts.

Component Administration Networks provide a bridge from the Smart Contract world to the software world and ensure the fault tolerance of supervised components. Such a network stores checkpoints for the supervised components which allow a new replica to continue the work if another had crashed. Simulation results show that failure rates of up to 75% can be tolerated among the CAN nodes if enough candidate nodes are willing to join the network periodically. This is encouraged by reimbursing the nodes that are part of this network. We have demonstrated the usage of an Orchestration Component Administration Network to ensure the continuity of Application deployment. Additionally, in conjunction with the Application Contract, the checkpoints are used to assess the amount of time different entities in the system have dedicated for the execution of the Application and ensure a fair price for the user and a fair reimbursement all participants.

The proposed architecture and mechanisms pave the way for a decentralized free-market, where individually owned resources meet the demands for Cloud Applications. In the realm of multiple Cloud Contracts, resources prefer the Cloud with the highest pay, but clients prefer Clouds with the lowest price. Resources are free to move to Clouds that may pay less, but more often, and some users may pay a higher price for more efficient hardware.

REFERENCES

[1] Urs Hölzle and Luiz André Barroso. Warehouse-scale computers. *IEEE Internet Computing Magazine*, 14(1):33, 2010.
[2] Luiz André Barroso, Jimmy Clidaras, and Urs Hölzle. The datacenter as a computer: An introduction to the design of warehouse-scale machines. *Synthesis lectures on computer architecture*, 8(3):1–154, 2013.
[3] Miguel Castro and Barbara Liskov. Practical byzantine fault tolerance and proactive recovery. *ACM Transactions on Computer Systems (TOCS)*, 20(4):398–461, 2002.
[4] Bernadette Charron-Bost, Fernando Pedone, and André Schiper. Replication. *LNCS*, 5959:19–40, 2010.
[5] Leslie Lamport, Robert Shostak, and Marshall Pease. The byzantine generals problem. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 4(3):382–401, 1982.
[6] Marshall Pease, Robert Shostak, and Leslie Lamport. Reaching agreement in the presence of faults. *Journal of the ACM (JACM)*, 27(2):228–234, 1980.
[7] Schneider, F.B.: Implementing fault-tolerant services using the state machine approach: A tutorial. ACM Computing Surveys (CSUR) **22**(4), 299–319 (1990)
[8] Benet, J.: Ipfs-content addressed, versioned, p2p file system. arXiv preprint arXiv:1407.3561 (2014)
[9] Maymounkov, P., Mazieres, D.: Kademlia: A peer-to-peer information system based on the xor metric. In: International Workshop on Peer-to-Peer Systems, pp. 53–65. Springer (2002)
[10] Pouwelse, J., Garbacki, P., Epema, D., Sips, H.: The bittorrent p2p file-sharing system: Measurements and analysis. In: International Workshop on Peer-to-Peer Systems, pp. 205–216. Springer (2005)
[11] Anderson, D.P.: Boinc: A system for public-resource computing and storage. In: proceedings of the 5th IEEE/ACM International Workshop on Grid Computing, pp. 4–10. IEEE Computer Society (2004)
[12] Fedak, G., Germain, C., Neri, V., Cappello, F.: Xtremweb: A generic global computing system. In: Cluster Computing and the Grid, 2001. Proceedings. First IEEE/ACM International Symposium on, pp. 582–587. IEEE (2001)
[13] Sarmenta, L.F.: Sabotage-tolerance mechanisms for volunteer computing systems. Future Generation Computer Systems **18**(4), 561–572 (2002)
[14] Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system. URL https://bitcoin.com/bitcoin.pdf (2008)
[15] Buterin, V., et al.: Ethereum white paper, 2014. URL https://github. com/ethereum/wiki/wiki/White-Paper (2013)
[16] Protocol Labs: Filecoin: A Decentralized Storage Network URL https://filecoin.io/filecoin.pdf [Accessed: 8-12-2020].
[17] Fisch, B., Bonneau, J., Greco, N., Benet, J.: Scaling proof-of-replication for filecoin mining. Tech. rep., Technical report, Stanford University, 2018. https://web. stanford. edu … (2018)

[18] Talal Ashraf Butt, Razi Iqbal, Khaled Salah, Moayad Aloqaily, and Yaser Jararweh. Privacy management in social internet of vehicles: Review, challenges and blockchain based solutions. *IEEE Access*, 7:79694–79713, 2019.

[19] Geetanjali Rathee, Ashutosh Sharma, Razi Iqbal, Moayad Aloqaily, Naveen Jaglan, and Rajiv Kumar. A blockchain framework for securing connected and autonomous vehicles. *Sensors*, 19(14):3165, 2019.

[20] Yehia Kotb, Ismaeel Al Ridhawi, Moayad Aloqaily, Thar Baker, Yaser Jararweh, and Hissam Tawfik. Cloud-based multi-agent cooperation for iot devices using workflow-nets. *Journal of Grid Computing*, pages 1–26, 2019.

[21] Moayad Aloqaily, Ismaeel Al Ridhawi, Haythem Bany Salameh, and Yaser Jararweh. Data and service management in densely crowded environments: Challenges, opportunities, and recent developments. *IEEE Communications Magazine*, 57(4):81–87, 2019.

[22] T. Lynn, H. Xiong, D. Dong, B. Momani, G. Gravvanis, C. Filelis-Papadopoulos, A. Elster, M. Khan, D. Tzovaras, K. Giannoutakis, D. Petcu, M. Neagul, I. Dragon, P. Kuppudayar, S. Natarajan, M. McGrath, G. Gaydadjiev, T. Becker, A. Gourinovitch, D. Kenny, and J. Morrison. Cloudlightning: A framework for a self-organising and self-managing heterogeneous cloud. In *the 6th International Conference on Cloud Computing and Services Science*, volume 1, pages 333 - 338, 2016.

[23] Christos Filelis-Papadopoulos, Huanhuan Xiong, Adrian Spataru, Gabriel G Castañé, Dapeng Dong, George A Gravvanis, and John P Morrison. A generic framework supporting self-organisation and self-management in hierarchical systems. In *Parallel and Distributed Computing (ISPDC), 2017 16th International Symposium on*, pages 149–156. IEEE, 2017.

[24] Christos K Filelis-Papadopoulos, Konstantinos M Giannoutakis, George A Gravvanis, and Dimitrios Tzovaras. Large-scale simulation of a self-organizing self-management cloud computing framework. *The Journal of Supercomputing*, 74(2):530–550, 2018.

[25] Paul Stack, Huanhuan Xiong, Dali Mersel, Maxime Makhloufi, Guillaume Terpend, and Dapeng Dong. Self-healing in a decentralised cloud management system. In *Proceedings of the 1st International Workshop on Next generation of Cloud Architectures*, pages 1–6, 2017.

[26] Adrian Spataru, Laura Ricci, Dana Petcu, and Barbara Guidi. Decentralized cloud scheduling via smart contracts. operational constraints and costs. In *The International Symposium on Blockchain Computing and Applications (BCCA2019)*, 2019.

[27] Leslie Lamport et al. Paxos made simple. *ACM Sigact News*, 32(4):18–25, 2001.

[28] CloudLightning. CloudLightning deliverable D5.3: GATEWAY SERVICE . http://cloudlightning.eu/work-packages/public-deliverables/, 2017. Accessed: 2020-11-04.

[29] Mircea Moca, Cristian Litan, Gheorghe Cosmin Silaghi, and Gilles Fedak. Multi-criteria and satisfaction oriented scheduling for hybrid distributed computing infrastructures. *Future Generation Computer Systems*, 55:428–443, 2016.

Edited by: Viorel Negru
Received: Nov 19, 2020
Accepted: Dec 18, 2020

AIMS AND SCOPE

The area of scalable computing has matured and reached a point where new issues and trends require a professional forum. SCPE will provide this avenue by publishing original refereed papers that address the present as well as the future of parallel and distributed computing. The journal will focus on algorithm development, implementation and execution on real-world parallel architectures, and application of parallel and distributed computing to the solution of real-life problems. Of particular interest are:

Expressiveness:

- high level languages,
- object oriented techniques,
- compiler technology for parallel computing,
- implementation techniques and their efficiency.

System engineering:

- programming environments,
- debugging tools,
- software libraries.

Performance:

- performance measurement: metrics, evaluation, visualization,
- performance improvement: resource allocation and scheduling, I/O, network throughput.

Applications:

- database,
- control systems,
- embedded systems,
- fault tolerance,
- industrial and business,
- real-time,
- scientific computing,
- visualization.

Future:

- limitations of current approaches,
- engineering trends and their consequences,
- novel parallel architectures.

Taking into account the extremely rapid pace of changes in the field SCPE is committed to fast turnaround of papers and a short publication time of accepted papers.

INSTRUCTIONS FOR CONTRIBUTORS

Proposals of Special Issues should be submitted to the editor-in-chief.

The language of the journal is English. SCPE publishes three categories of papers: overview papers, research papers and short communications. Electronic submissions are preferred. Overview papers and short communications should be submitted to the editor-in-chief. Research papers should be submitted to the editor whose research interests match the subject of the paper most closely. The list of editors' research interests can be found at the journal WWW site (`http://www.scpe.org`). Each paper appropriate to the journal will be refereed by a minimum of two referees.

There is no a priori limit on the length of overview papers. Research papers should be limited to approximately 20 pages, while short communications should not exceed 5 pages. A 50–100 word abstract should be included.

Upon acceptance the authors will be asked to transfer copyright of the article to the publisher. The authors will be required to prepare the text in LaTeX 2$_\varepsilon$ using the journal document class file (based on the SIAM's `siamltex.clo` document class, available at the journal WWW site). Figures must be prepared in encapsulated PostScript and appropriately incorporated into the text. The bibliography should be formatted using the SIAM convention. Detailed instructions for the Authors are available on the SCPE WWW site at `http://www.scpe.org`.

Contributions are accepted for review on the understanding that the same work has not been published and that it is not being considered for publication elsewhere. Technical reports can be submitted. Substantially revised versions of papers published in not easily accessible conference proceedings can also be submitted. The editor-in-chief should be notified at the time of submission and the author is responsible for obtaining the necessary copyright releases for all copyrighted material.