Study and Evaluation of Test of Times Brooks-Iyengar Algorithm

LATESH KUMAR KJ
Member of Technical Staff, Computer Science and Engineering, Siddaganga Institute of technology, Karnataka, India
and
LEENA H U
Research Scholar, Master of Computer Application, Siddaganga Institute of technology, Karnataka, India

The current fault tolerant computing systems and various computer systems still rely on the outstanding technique of resilient sensor Brook-Iyengars (BI) algorithm and this was invented and published in 1996 with IEEE computing systems. The novel idea proposed of the algorithm institutes groundwork standards in various domains like Real-Time Operating Systems (RTOS), Fault Tolerant Schemes (FTS) and various application computing systems. The crucial contribution of the algorithm is majorly found in enhancing the features of MINIX real-time operating system, and hybrid architecture and scalability of the algorithm is proficient enough to encounter the unreliable distributed sensors data using the Byzantine [1] agreement and distributed decision-making process methods. In this paper, we study and reveal the contribution and influences of BI in MINIX real-time operating systems and their recent enhancement with fault tolerant schemes with a case study.

Keywords: MINIX, FTS, RTOS, BI

1. INTRODUCTION

The distributed systems of real-time services principally depend on the precise results obtained and the time. More precisely, the prime attention is applied on time constraint on thread, process and running tasks of computer system. Additionally, various process and control systems are connected sequentially in different areas ranging from large aviation machinery to small micro-controller in numerous computing domains. Conventionally, to support this kind computing systems, the distributed sensor operating system is used to build and support the real-time application. The prime features used in building the application systems are fault-tolerance, resource prediction and handling the integration of resource constraints with scheduling system. The sensors used in the system afford information on distributed environment, by which the systems is controlled using the actuators whenever a faulty resource is discovered. In this article we depict the exceptional Brooks-Iyengar algorithm also known as BI across world in the sensor networking domain. This algorithm is declared Test of Times by IEEE research society at an IEEE international conference and honored the Inventors Dr. S.S Iyengar with his Student Dr. Brooks, the prime functional area of the algorithm is with sensor network and fault-tolerance in distributed computing system. The algorithm was published in 1996 with a thesis study and since then algorithm scaled in various domains in countless research, product and technology implementation across the world. The chief aim while designing this algorithm was to mitigate redundancy of fault-tolerance in real-time applications and computing systems. Further, the algorithm adopted Byzantine [1] technique and distributed processing methods to encounter the fault-tolerance schemes in hybrid computing architectures and because of this, today the algo-

Acknowledgements: Many thanks to Dr. S.S. Iyengar, Dr. R Brooks, Dr. Regina Pablo, Dr. Garbiel Wainer and colleagues of various School of Computer Science and Engineering. Special thanks to Andres S Tannenbaum for his kind clarification on MINIX architecture. I also offer my thanks to Fault-tolerance research group for timely suggestions.

rithm is completely part of real-time MINIX operating system and numerous domains.

Todays internet ecosphere connected with abundant automation systems of vibrant environments to communicate the services. The dynamic environments are hard to determine in time and hence the computing world heavily depends on sensors to get the right information in right time to complete the computation process, because of this sensors based computing is indisputable. In general, most of native sensors developed and implemented in various automated control systems is strenuous because of limited accuracy discovered from the sensors. Additionally, these sensors omit raw noise in reading data from devices and this destroys the data accuracy of computing system. The Brooks-Iyengar [Richard R. et al. 1996] algorithm is recognized across for it exceptional fault-tolerance schemes and increased accuracy of data and time management by its distributed sensor network technique.

The prime virtues of BI algorithm is that it can function precisely and accurately even when it discovers a defective sensor [Mohammad Ilyas et al. 2004] with its fault-tolerance intelligence and switches the determined and accuracy value from each every peer node. The core feature like fault-tolerance and distributed flexibility of algorithm does not process or transmit faulty data from one node to another though any of the nodes that fails to send right data and hence this is considered to be fusion based sensor system. At the inception, BI is designed to bridge techniques Byzantine and fault-tolerance [D. Dolev 1982] and this was proved in 2016 [Ao, Buke et al. 2016] sensor fusion technique evaluations. Additionally, BI found to be the first of its kind to integrate dissimilar grounds by adopting the Dolevs [L. Lamport 1982] with FCA techniques proposed by Mahaney [Mahaney et al. 1985] and Schneiders. The BI algorithm adopted the Crusaders Convergence Algorithm (CCA) [D. Dolev 1986] to increase the processing capacity in high performance computing systems [Mahaney et al. 198] like MINIX, distributed computing, information fusion and health sector. This scalability features didnt limit the BI to specific domains instead it scattered across various domains and application system as we cited. The hybrid feature of algorithm was able to counter the computational issues of floating-point and achieve the increased consistency on a distributed computing system, this was also found successful in distorting the errors produced by control systems of hardware due to various limitations. The cloud computing resources like network, storage, security and storage are to be served round the clock and these settings demand fault-tolerant and precision [L. Lamport et al. 1982] end to end.

The Brook-Iyengars algorithmic program is beneficial and effective in these instances by achieving sturdy and distributed accuracy owing to the novel intelligence of algorithmic program. Disappointingly, data, service and security area unit compromised usually between them, still the usage of algorithmic program will increase by not sacrificing the accuracy of information, service and security. The fault-tolerance mechanism outlined within the algorithmic program is very helpful in each active and passive cluster computing in primary and disaster computing sites. With this algorithmic program, sturdy distributed computing applications is developed and deployed seamlessly. Todays world is choked mostly with Internet of Things (IoT) and cloud based services during which sensors area unit is important half computing systems. The quantity of information communication across current dynamic environments is resulting in errors, mechanical failures and uncertainties in sensors. To avoid this, the backup mechanism is inserted however fault-tolerance and accuracy cant be managed. The BI algorithmic program has edge and higher binds in achieving the system quality specifically. The prime recognition of BI from the computing is extending the seamless support in interfacing the todays trendy technology appliances, systems and services with the legacy systems without any concern in various software applications.

2. BACKGROUND AND RELATED WORK

The MINIX operating system by Andrew S Tanenbaum is considered to be the foundation block to build Real-Time MINIX operating system by the researcher Gabriel Wainer [Gabriel A. Wainer 1995], [Pablo J. Rogina et al. 1999], this study and implementation elevated the MINIX to Real-Time operating system called RT-MINIX using the exceptional Brooks-Iyengar algorithm in the areas like scheduling algorithm, queue scheduling and in exploiting the fault-tolerance systems. In this section we describe in deep in what segments BI algorithm in involved in building the RT-MINIX, just before that we quickly glance through native MINIX architecture to discover differences between two. The complete understanding of MINIX creates stage for understanding the RT MINIX in control systems and application computing.

The MINIX architecture is designed such that system concerned services, device drivers and users are allotted to run on highest scale on miniscule kernel design as shown in Figure [1]. The user mode operations are provisioned using the prime control structures SYS and CLOCK and these manage the time and system processing procedures with kernel. Additionally, the Memory Management Unit (MMU), Central Processing Unit (CPU), Interrupt handling (INT) and Inter Process Communication (IPC) are additional primitive kernel functioning units. A unique and smart thing of MINIX over other peer operating system is that it has Resource Scheduler at user level in addition to system space, this unit governs all the device and kernel interactions of users separately from the system usage and has self-fault fixing techniques during fault procedures by users.

The kernel communication with all other system libraries are managed transparently and permits request of user to run in user space by creating the user process using the fork () library, this fork internally communicates to memory management unit to reserve desired memory for the process to run with a suitable process space. In the process if any slot for desired process is found then processing unit sends a communication to user by creating the necessary environment, this is operation of MINIX is identical to MINIX. In addition, the SYS call is considered to be the prime in processing the tray as it interfaces core connection between users and kernel processing methods. All user requests must go communicate through the SYS kernel controls such as to recreate data between process threads SYS uses system_vircopy to during processing to communicate the signaling is done through system_set_alaram as listed in below Table [1].

Kernel Call	**Purpose**
SYS_VIDEVIO	Read or Write a vector of I/O ports
SYS_VIRCOPY	Safe copy between address spaces
SYS_GETINFO	Get a copy of kernel information

Table I: MINIX Kernel SYS Calls

The second prime object to be discussed here is CLOCK control using which MINIX kernel communicates and handles process such CPU usage, cron tasks, hardware timers, scheduling of jobs and interrupt handlers. Except all others signals, the interrupt time triggers to capture data from the moment when kernel is on with process id initiated from the core. In addition to this MINIX allows external devices interfacing and device programs to run in process tray using special peripheral device drivers and these are coupled strongly with kernel to counter the failures with various levels of kernel and user communication.

3. BROOKS IYENGAR ON REAL-TIME MINIX OPERATING SYSTEM

The MINIX operating system powered by Tanenbaum [Pablo J et al. 1999] was enhanced by Wainer and Pablo [Junkil Park et al. 2017], [R. Brooks et al. 1996] to real-time MINIX with

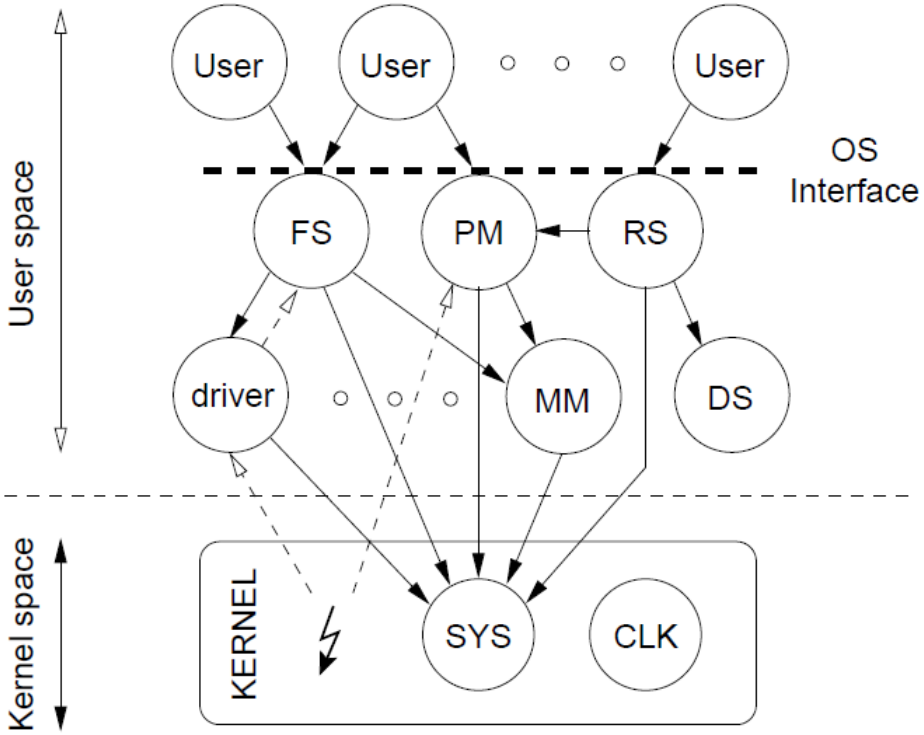

Figure 1. Native MINIX OS Inter-Process-Communication (IPC) Architecture

elevated features and known as RT-MINIX. This enhancement was performed by academic research project and they added few novel features in the below Table [2] areas. The research conducted by Pablo Regina and Gabriel Weiner proved that numerous control systems and real-time applications are using the BI in the area of fault-tolerance system. The elevated features of BI involvement uplifted the algorithm strength with hybrid distributed computing methods and proved to be different and better over traditional systems.

Feature	**Description**
Fault-Tolerance	Countering the errors and to provide non-fault results.
Isolation of errors	Dynamically identifying faulty signals and nodes in the system
Process scheduling	Managing faulty process

Table II: MINIX Elevation Features

The Figure [2] explains in detail about the new features of Brooks-Iyengar algorithm added to MINIX operating to shape up RT-MINIX by Gabriel and team. The MINIX source code [Junkil Park et al.] was programmed by adopting the Brooks-Iyengar algorithm features to offer real-time controls with hybrid computing capabilities. The team added new method of scheduling technique rate-monotonic to schedule process threads to reduce the failures and then another new technique called first process with earliest deadline is programmed into MINIX kernel to provide quick process threads for internal and external sources simultaneously. Further the key component fault-tolerance was programmed using the intelligence of BI. All these updates made

on MINIX kernel impacted data structure and architecture which are fixed during the testing of the updates. In specific, sensors, timers and scheduling were tested critically to match the native MINIX operation standards.

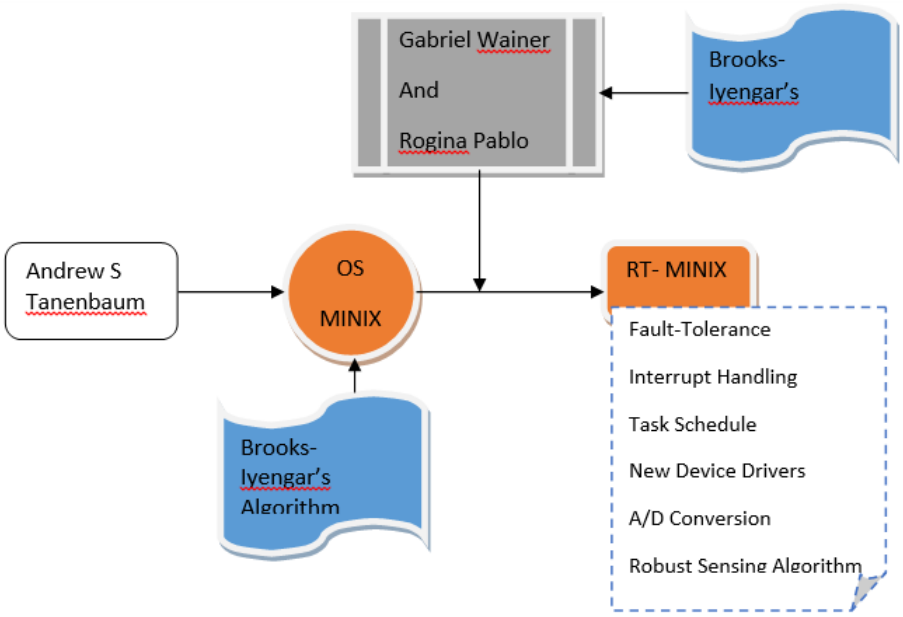

Figure 2. Schematic of the Proposed Framework

The second prime object discussed here is CLOCK control using which MINIX kernel communicates and handles process such CPU usage, cron tasks, hardware timers, scheduling of jobs and interrupt handlers. All interrupt timers trigger to capture data from the moment when kernel is on with process id initiated from the core. In addition to this MINIX allows external devices interfacing and device programs to run in process tray using special peripheral device drivers and these are coupled strongly with kernel to counter the failures with various levels of kernel and user communication. The Table [3] below illustrates few of the data structure modified while programming the intelligence of BI to native MINIX system.

Data Structure	**Parameters**	**Description**
struct rt_globstats {		
int actpertsk;	Acrta_petsk, Act_pertsk	Period and aperiodic real time tasks, total running tasks.
int actapetsk;		
int misperdln;		Total missed deadlines
int misapedln;	Mis_perdln, Mis_apedln	
int totperdln;		Total real-time task scheduled instance
int totapedln;		
int gratio;	Tot_petsk Gratio	Guarantee ratio between deadline and instances
clock_t idletime; };	Idletime	Computing Time in Second (clock Tick)

Table III: Modified Data Structures of MINIX under the influence of Brooks-Iyengar

The modified kernel is tested in all corners of MINIX functionalities to ensure the real-world challenges specifically, in real-time application and computing systems that are based of sensor computing, based on this further specific customization are made in the MINIX structure to

adopt isolation of errors and hybrid fault-tolerance methods. During this period, a newer version of MINIX came out to market with extended features with analog to digital conversion [Warrenedgar 2019], this update was specific to receive analog data as many legacy computing systems were adopted with analog functionalities specifically the production systems and chemical processing units. In this update further Brooks-Iyengar algorithm sensor intelligence is effectively implemented to reduce the noise and faulty data from various sensors employed. Further, the entertainment based computing system started using the sensors while offering the smart game pads, over there RT MINIX was found to be perfect implementation and the smart BI sensor computing is used in developing the device drivers of numerous game companies. According the researcher Pablo Regina the BI is also implemented in Fast Convergence Algorithm (FCA) to achieve the precision from fault-tolerant schemes and the algorithm [Kumar V. 2012] is powered with the help Byzantine methods effectively.

At the outset, the complete coding was performed based on Brooks-Iyengar algorithm intelligence with hybrid computing mechanism. The next immediate phase was to integrate the smart capability to make use of the real-time data and in order to do this, four potentiometers were used to sense the signals/data from analogic inputs from the joystick port. These sensor positions are arranged with actual positions for a simulation based robotic arm. An accurate and precise functionality of algorithm was noticed by providing an exclusive value from the simulated sensors in spite of the faulty units and at the same time users were offered open chance to modify the data by varying the potentiometers. At last, all the updated code is tested for various feasibility and real-time constraints and then the novel algorithm intelligence is united into MINIX kernel. The Figure [3] shows the RT-MINIX Kernel and new feature additions with Brook-Iyengar Algorithm.

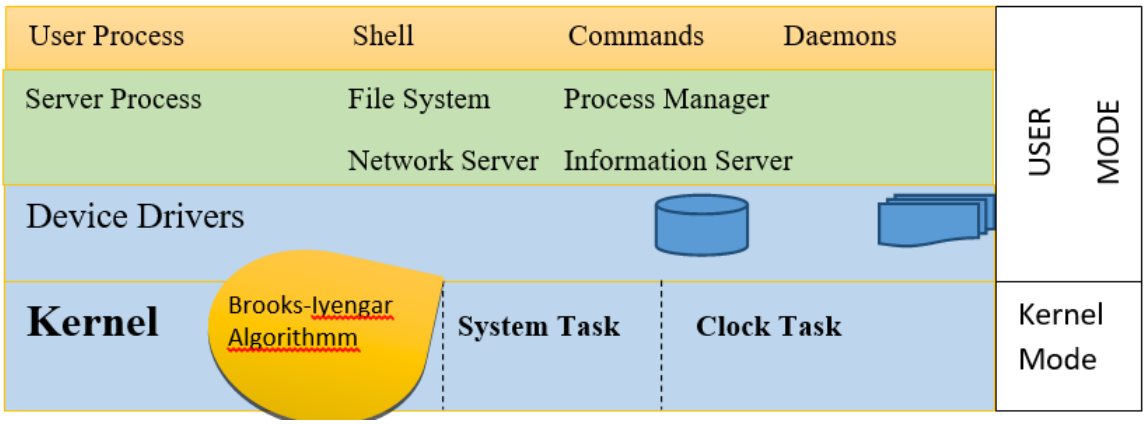

Figure 3. Brooks-Iyengars Algorithm Segment in MINIX Kernel

The software developers were given a set of functions to work with intellectual sensors, using these it was possible to generate many new services and devices like /dev/js0 and after that smart sensors were able to read data in the presence of faulty sensors. Once the operating system is enhanced with RT services, the demand ascended for various computing tools and applications. The Brook-Iyengars algorithm needed a test on the novel techniques applied on kernel, in order to evaluate the data structure through vivid system and library calls.

4. BI CASE STUDY ON MPI

In this paper we studied the Brooks-Iyengars adoption in various domains, in this case study we are describing the impact on MPI and virtualization with cited results. The BI is adopted in

OpenMPI [Ao, B. et al. 2016] project and in this project the researchers have created non-faulty channels to transfer messages from process to process on open source environment. This project was associated by academic researchers, industry partners and open source research community. The libraries and products developed by MPI offers numerous benefits to open source community, researchers and application users of computer science. Figure [4] depicts the Brooks-Iyengars algorithm intelligence transformed into coding practices in developing fault-tolerance system.

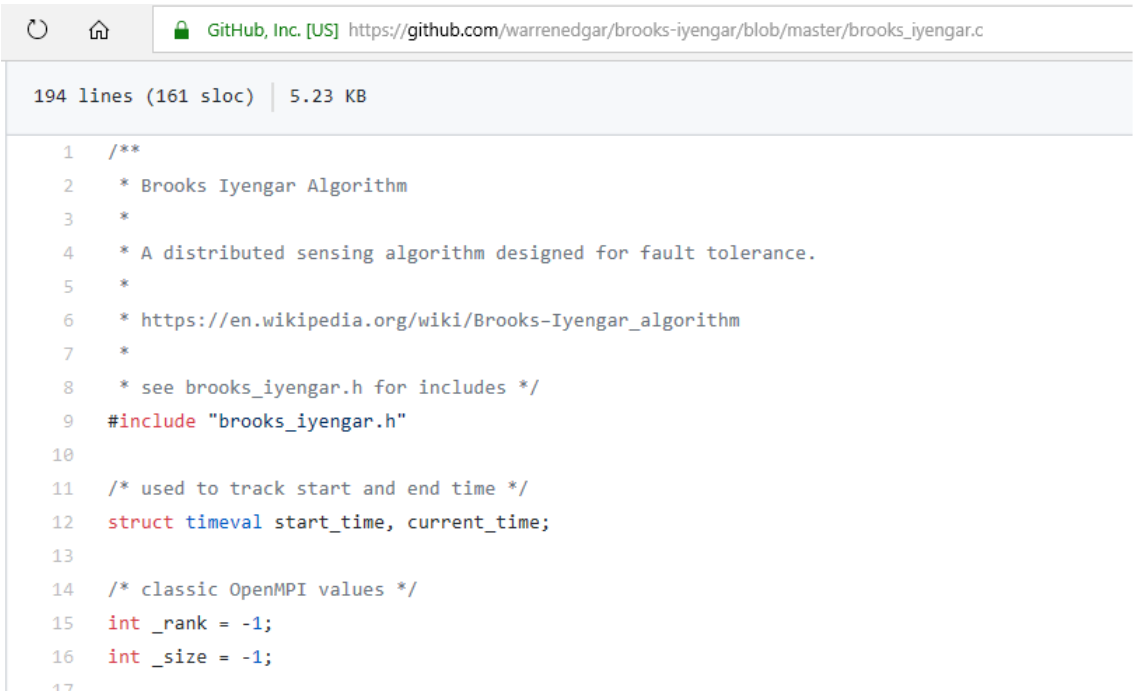

Figure 4. Brooks-Iyengars Algorithm Adoption in developing openMPI Library

A classical problem in distributed computing is Byzantine Generals' Problem, introduced in the 1982 white paper of the same name. It attempted to formalize a definition for faulty (traitorous) nodes in a cluster and how for the system to mitigate it and solutions such as majority voting or signed messages were suggested. Majority voting requires that all generals have the same information, a suggestion that isn't always possible. Signed messages are good to verify that it was the correct node in communication, even if it doesn't verify that the content itself is correct. The BI algorithm uses sensor fusion to mathematically eliminate the faulty sensors. In short, this is achieved by taking measurements over an interval and the measured interval is then shared between all sensors on the network. The fusion step then happens, by creating a weighted average of the midpoints of all the intervals. At this point you can eliminate any sensors with high variance or use heuristics to choose the reliable nodes. It runs in O (NlogN) time and can handle up to N/3 faulty sensors [M Hadi Amini et al. 2019]. The BI algorithm provided an intellectual fault-tolerant and error isolation sensor layer with timer is programmed and mapped into kernel system. The prime smart sensor is set with base interval value and each sensor node value is measured and if the measured value is not available from the subset node parameter value then the sensor considered to be faulty. Table [4] described major openMPI method involved with sensor programming function modules with explanation [Vijay Kumar 2013].

OpenMPI Methods	Description
Isend and Ireceive	Non-blocking sends and receives were used to communicate from sensor to sensor. In order to process the data each sensor needs the data from every other sensor in the network. This means there are a worst case of N$\hat{2}$ messages being passed at any given point. Due to this large number, it is best to use non-blocking communications.
Barrier	This acts as a sync step for all sensors. Barrier merely acts as a join for processes in the context of OpenMPI. It is very useful for a simulation as this to stop one process from being a front runner.
Broadcast	Seeing as this is a timed excution program, it is necessary for each sensor to kill itself after a fixed period of time. However, it is possible for one process to keep running if it gets to the check before all the others. To get around this one thread was designated with the resposibility to check the runtime, and then broadcasted the result to all others.

Table IV: Brooks-Iyengars features with openMPI

The Figure [5] explains the output obtained after implementing the BI intelligence using the openMPI programming procedures with methods listed in the Table [4] for removing the defective sensors. The output shows output curves from 0 to 6 across the X and Y axis 6 in three different colors, the data curve in green color indicates the output obtained by the implementation of BI algorithm with 100% accuracy after the elimination of defective sensors and data. However, the red curve indicates the unnecessary minor load time intervals of the final code procedure for removing the defective sensor. The below output includes faulty sensors with non-faulty sensors to measure the interval and data from the algorithm and evaluation of results discovered that defective sensors are controlled from spoiling the measurements by benefiting from payback error distribution.

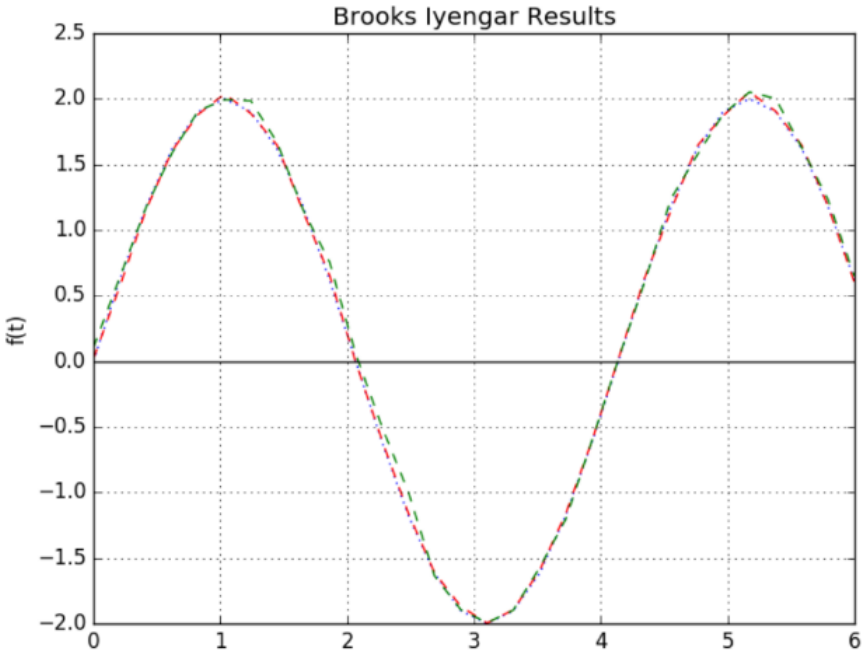

Figure 5. Brooks-Iyengars and openMPI exceptional results

To finalize the results we considered the dumb average data because faulty and non-faulty

sensors generated common scale of data in both channels created in the program. The red curve output would have been aggressive on green curve if proposed BI algorithm has failed to eliminate the noise and faulty data.The concluded results proved that Brooks-Iyengar algorithm is exceptional in fault-tolerance and scalable numerous computing system. The program was developed using MPI engine on Red hat 9 distribution and the code focused only to discover the faulty node and remove the noise from defective and clumsy sensor network setup.

5. USE CASES

The research study discovered the involvement of Brooks-Iyengars algorithm from 1996 till date in various domains as listed in below Table [5].

S.N	**Area/Technology**	**User**
1	Operating System MINIX	Andrew S Tanenbaum
2	IT Industry Software Systems	BBN, BAE Systems, Sense-IT
3	Research Labs	Penn State [20] Applied Research Lab(ARL)USC/ISI
4	Research Communities RT:MINIX	Pablo J. Rogina and Gabriel Wainer
5	Transportation Railway [9]	Buke Ao, BYD Company
6	Education- Training/Journal/Conference	Duke, Wisconsin, UCLA, Cornell, Purdue, Georgia Tech, Clemson, Maryland and LSU University
7	Defense- Thale Groups	UK Defense Manufacturer
8	Navy-Software	Maritime Domain Awareness Software
9	Technology	Cyber-physical systems, Data Fusion, Robot Convergence, High-Performance Computing, Artificial Intelligence Systems
10	Open Source	RT Linux, KURT, YARTOS, Spring
11	Medical (Virtualization) [10]	

Table V: Current Implementation Domains Brooks-Iyengars Algorithm

Furthermore, in a recent publication by R Baral in springers [Vijay Kumar 2013] sustainable interdependent networks II book chapter, we found the adoption of Brooks-Iyengars algorithm in interdependent Networks from societal perspective: Multi-context Influence Tracking on Social Network. This work focused on location information in influence maximization only, exploited location as a simple user property, and did not analyze the contextually dynamic user mobility behaviors. The Brooks-Iyengars algorithm is extensively influenced in finding the location grids [Krishnamachari et al. 2004] by eliminating the noisy nodes from the network.

6. CONCLUSION

In this paper prime we studied and narrated the exceptional impact of Brooks-Iyengar algorithm impact in various domains like Education, Research, Science, Technology, Transport, Medical and computing systems. The BI fault-tolerant technique is reliable and significantly increasing the margin with new computing environments without being modified. This algorithm provides the robust implementation and seamless scalability under faulty sensor conditions for various domains. Finally, the algorithm Stand the Test of Times from last two decades and hope it continues the successful journey further. In our next implementation we are making use of BI in Cyber Physical Security [Junkil Park et al. 2017] systems for augmentation of security devices to propose a sturdy security devise.

References

M Hadi Amini 2019. Sustainable Interdependent Networks II *Smart power grids to Intelligent Transportation Networks,Book Chapter- II, Springer Vol.186,*

Gabriel A. Wainer 1995. Implementing Real-Time services in MINIX. *ACM SIGOPS Operating Systems Review Vol.29,* pp. 75-84

Tanenbaum Andrew 1999. Operating systems (2nd ed.): design and implementation. *Prentice-Hall, Inc.*,ISBN:0-13-638677-6

Buke Ao. 2015 Robust Fault Tolerant Rail Door State Monitoring Systems: Applying the Brooks-Iyengar Sensing Algorithm to Transportation Applications *International Journal of Next-Generation Computing Vol.8,* No.2,pp.108-114.

Richard R. Brooks, and S. Sithrama Iyengar 1996. Robust Distributed Computing and Sensing Algorithm. *Computer Vol.29(6),*,ISSN 0018-9162,pp.53-60.

Krishnamachari, B., and Iyengar, S.S. 2004. Distributed Bayesian Algorithms for Fault-Tolerant Event Region Detection in Wireless Sensor Networks. *IEEE Transaction in Computing*

Buke Ao, Yongcai Wang,Iyengar S.S.,and Yu Lu 2016. On Precision Bound of Distributed Fault-Tolerant Sensor Fusion Algorithms. *ACM Computer Surv Vol.49(1),*

D. Dolev 1982. The Byzantine Generals Strike Again. In *.J. Algorithms*pp.14-30

Chakrabarty K, Iyengar, S.S.,H. Qi,and E.C. Cho 2002. Grid Coverage of Surveillance and Target Location in Distributed Sensor Networks. *IEEE Transactions on Computers Vol.51(12),*

D. Dolev1986. Reaching Approximate Agreement in the Presence of Faults. *Journal of the ACM Vol.33(3),*pp. 499516

S. Mahaney, and F. Schneider 1985. Inexact Agreement: Accuracy, Precision, and Graceful Degradation. *Proc. Fourth ACM Symp. Principles of Distributed Computing* pp. 237249

Mohammad Ilyas, and Imad Mahgoub 2004. Handbook of sensor networks: compact wireless and wired sensing systems. *Bit.csc.lsu.edu. CRC Press Vol.51(12),* pp 2529

L. Lamport, R. Shostak,and M. Pease 1982. The Byzantine Generals Problem. *ACM Transactions on Programming Languages and Systems Vol.4(3),*pp. 382401

Junkil Park , Radoslav Ivanov Philadelphia,James Weimer, Miroslav Pajic Sang Hyuk Son,and Insup Lee 2017. Security of Cyber-Physical Systems in the Presence of Transient Sensor Faults. *Journal ACM Transactions on Cyber-Physical Systems Vol.1(3),*Article No. 15

R. Brooks , and S. Iyengar 1996. Robust Distributed Computing and Sensing Algorithm. *IEEE Computer* pp. 53-60

Pablo J. Rogina, and Gabriel Wainer 1999. New Real-Time Extensions to the MINIX operating system. *Proc. Of 5th International Conference on Information System Analysis and Synthesis*

Kumar, V 2012. Computational and compressed sensing optimizations for information processing in sensor network. *International Journal of Next-Generation Computing*

Penn State University 2013. Reactive Sensor Networks. *AFRL-IF-RS-TR-2003-245, Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical, Information Service (NTIS), Defense Advanced Research Laboratory*

Warrenedgar 2019. An implementation of the Brooks-Iyengar algorithm using OpenMP. *Ihttps://github.com/warrenedgar/brooks-iyengar*

Dr. Latesh Kumar obtained his Ph.D from AeU university of Malaysia, prior to this he has served in Information Technology Company Lionbridge at Mumbai, India. He later associated to Hewlett Packard, Information Technology Company, California as Service Engineer and then Technical Solution Consultant in both California and North Carolina, CO. His clients included the Data Protection, Automobile, Airports and the U.S. Army. Few years later he moved to NetApp, IT storage company and served in California and Florida as Product Partner Manger. Dr. Latesh has published numerous papers in Journals, Conference and Technical articles of IT companies like Springer, IEEE, ACM, Elsevier and IT next, NetApp Technical Library worldwide. An active Research Consultant, he has investigated several projets (Data, Security and Technology) and conducting research in cyber security and machine learning. He received several awards including distinguished Customer Excellency for Technical Commitment and Content Training at Hewlett Packard, and Best International Journal award at PSRC, Indonesia. He is professional badminton Player.

Mrs Leena H.U. is currently working as full-time research scholar in the Department of Master of Computer Applications, Siddaganga Institute of Technology, B.H. Road, Tumakuru- 572103, Karnataka, (India). She has 11+ years of experience in IT industry and moved to academics. Her research interest is in the area of netwroks, spatial analytics and Cloud computing. A grant of Rs. 3 lakh received for the research project from ICT Skill Development Society, Department of IT, BT and S & T, Govt. of Karnataka, New Age Incubation Network(NAIN) during 2016.