A HEURISTIC APPROACH FOR OPTIMIZING PERFORMANCE IN SOFTWARE-DEFINED NETWORKS

[1]P.R.Rupashini, [2]A.Subashini, [3]L.Babitha, [4]V.Bhuvaneswari
[1]Assistant Professor, [2]Assistant Professor, [3]Associate Professor, [4]Assistant Professor
Department of Computer Science
United Institute of Technology, Coimbatore, India

Abstract - Another half breed clever approach for enhancing the execution of Software-Defined Networks (SDN), in light of heuristic advancement techniques coordinated with Artificial Neural Network (ANN) worldview is exhibited. Evolutionary Optimization techniques, such as Shuffled Frog Leap Algorithm (SFLA) and Genetic Algorithms (GA) are employed to find the best set of inputs that give the maximum performance of an SDN. The Neural Network model is trained and applied as an approximator of SDN behavior. An analytical investigation has been conducted to distinguish the optimal optimization approach based on SDN performance as an objective function as well as the computational time. After getting the general model of the Neural Network through testing it with unseen data, this model has been implemented with SFLA and GA to find the best performance of SDN. The SFLA approach combined with SDN, represented by ANN, is identified as a comparatively better configuration regarding its performance index as well as its computational efficiency.

Keywords–ANN, Evolutionary Optimisation, SDN, Genetic Algorithms, ESFLA

I. INTRODUCTION

In the present years, hybridisation or blend of different machine learning and modification techniques has been used for innumerable shrewd system diagrams. The major purpose of fusing these methodologies is to vanquish particular requirements and to achieve synergetic effects [1]. These techniques including, Artificial Neural Networks (ANNs), the Adaptive Network Fuzzy Inferences System (ANFIS) are used for mapping and modeling purposes. Whilst evolutionary based optimization approaches, such as the Genetic Algorithm (GA) and Shuffled Frog-Leap Algorithm (SFLA) have been applied widely to produce powerful and optimized intelligent systems. Recently, with the fast development of the Internet, network topology, and the number applications have changing gradually, which has led to more complex structures and functions.

A network based on the traditional TCP/IP architecture faces many challenges, especially the router, as the network core, takes on too many efforts to deal with. As a result, the validity and efficiency of data forwarding is being threatened. Hence, we need to find a new kind of network architecture to solve the existing problems. To this end, the study of future networks is proceeding all over the world, and the Software Defined Network (SDN) is one of the [3]. SDN is an architecture enabling rapid innovation, while hiding much of the complexity of the networking design. As such, it is a promising solution for network virtualization that decouples control from the forwarding or data plane [4]. In doing so, it can provide the capability of remote and centralized control of the network forwarding-plane through the network control-plane.

SDN, as a network platform, has been studied widely in the literature and many researchers have proposed soft computing methods to model and optimize the network. Yilan and et al. [5] provided a genetic algorithm for solving the bandwidth-constrained multi-path optimization problem in the SDN. Gao and et al. [6] contributed a PSO algorithm to solve the control placement problem that takes into consideration both the latency between controllers and their capacities. Zhang and Fumin [7] explored applications of the SDN technique in the direction of the automation of network management, the unified control of optical transmission and IP bearing, smooth switching in a wireless network as well as network virtualization and QoS assurance. Risdianto and et al. [4] evaluated the performance of an SDN-based virtual network on different virtualization environments, including operating system based virtualization, hardware-assisted virtualization, and par virtualization. Sgambelluri and et al. [8] proposed novel mechanisms that have been specifically introduced to maintain working and backup flows at different priorities and to guarantee effective network resource utilization when a failed link is recovered.

A heuristic approach is proposed in this paper to optimize the performance of the SDN. The proposed system includes ANN to model inputs-outputs of the network and evolutionary algorithms employed to find the optimal set of inputs that maximize the SDN performance. In this paper a new method is presented to lessen the burden of SDN switch by making the network controller poll the switches adaptively, instead of making the SDN switch wait for an event all the time.

II. SOFTWARE-DEFINED NETWORKS

It was beforehand noticed that a SDN, which is halfway sorted out around the guideline of the compartmentalization of the control plane and information plane, has developed as an alluring PC organizing gadget. As the SDN empowers the control-plane to be customized as to its programming in order to control the remote framework equip, those providing the organization can take part in adaptable framework operation.

The certified particular of SDN coordinates the SDN controller in the control plane and, taking after this, it likewise incorporates the SDN switch in the information plane. Figure 1 gives a sign of the route in which the interface specific is composed as to the two. The SDN controller serves to keep up the stream tables in order to take part in the administration of each stream that is passed on through the framework.

The OpenFlow strategy can be viewed as an open SDN detail that is constituted of a couple of foundational components: to begin with, the OpenFlow switch and, second, the controller. The previous encourages the execution of the information plane while the last understands the control plane. It ought to be noticed that the OpenFlow technique as a totality is utilized by the exchange that happens with respect to the controller and the switches by means of a protected channel.

The OpenFlow switch and controller convey by means of a sheltered divert that is acknowledged with regards to an unmistakable practice and, as indicated, this is additionally alluded to as OpenFlow. The controller transmits a Flow Table towards the OpenFlow switches and each stream section in the previous component is constituted of a direction created from an association of fields, in this manner planning the moving toward groups.

It is outstanding this is an action that describes the issue of how the planning packages can be readied, for instance, by transmitting on a particular yield port. Moreover, various counters are utilized with a specific end goal to gather data identifying with the stream. Likewise, every entry can be connected to option data, one instance of this being the prerequisite level and the hard and sit without moving timeout of the match of timing gadgets.

The package is caught and passed on to the controller that utilizes the protected direct in those examples where a switch gets a bundle that is not participating in the coordination of the sections that have been presented. Having caught the package, the movement of the OpenFlow switch is managed by the controller therefore of the upgrade of its Flow Tables, in this way transmitting the planning bundles to the end point [8]. One ramifications of this is the relative housings are not required to share in the controller some other time therefore of the potential start of the Flow Tables' reformulation [3].

III. ARTIFICIAL INTELLIGENCE

Neural Networks

A branch of Artificial intelligence, dispersed at a significant pace as of late inferable from its reasonableness with respect to the displaying and determining capacity it has in connection to dynamic frameworks, is the ANN. In light of their promising potential, ANNs have risen as a focal branch of research into Artificial intelligence.

The enrolment of the information yield connections of nonlinear and amalgamation frameworks have been recognized as one of the key preferences offered by ANNs and, strikingly, this relationship can be direct, quickly, and cost-and time-successfully perceived by bringing down the mistake with respect to the system output(s) and the real output(s).

A characterizing highlight is that, taking after the fitting planning of the system, yields can be assessed inside a short space of time (in particular, in a matter of seconds).

Structures that inside on simulated neural systems are at present observing powerful application in a scope of territories – a couple of illustrations being including versatile control, laser applications, nonlinear framework distinguishing proof, mechanical autonomy, picture and flag handling, medicinal regions, design acknowledgment, mistake identification, prepare logging, and inexhaustible and practical vitality ranges – with a specific end goal to surmount impediments confronted by architects.

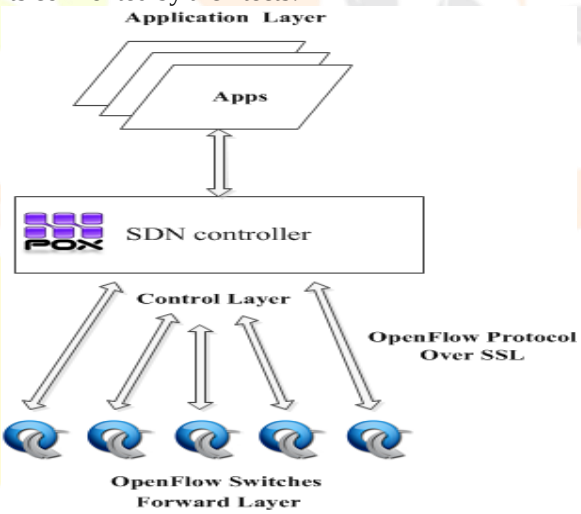

Figure 1.The software-defined network's architecture

Evolutionary Algorithms

Taking motivation and inspiration from natural procedures and, moreover, constructing the improvements in light of components identifying with iterative and probabilistic procedures, a scope of developmental calculations – including GAs, PSO, and reenacted strengthening – have been detailed lately. The main role of such improvements is for application in advancement issues. Two multi-reason and much of the time utilized calculations incorporate GA and PSO, and these are used in each space.

Genetic Algorithm

Figured by Holland in 1975, GAs are self-changing worldwide enhancement likelihood seek calculations, the essential idea of which was roused by the hereditary systems that shape the premise of the hypothesis of Darwinian characteristic determination and natural advancement. GAs work by mimicking the organic procedures that are seen in the normal world as driving the wonders of hereditary and transformative advancement; as indicated by the idea of regular determination, GAs give answers for profound issues by utilizing code strategy and proliferation forms. GAs has been broadly utilized in an assortment of areas with extensive viability as of late, and this is principally ascribed to their practically all inclusive pertinence and promising outcomes.

Enhanced Shuffled Frog Leaping Algorithm

In the shuffled frog-leaping algorithm, the population consists of frogs of a similar structure. Each frog represents a solution. The entire population is divided into many subgroups. Different subgroups can be regarded as different frog memes. Each subgroup performs a local search. They can communicate with each other and improve their memes among local individuals.

After a pre-defined number of memetic evolution steps, information is passed between memeplexes in a shuffling process. Shuffling ensures that the cultural evolution towards any particular interest is free from bias. The local search and the shuffling processes alternate until satisfying the stopping criteria.

Enhanced SFLA Algorithm Steps

1. Initialize parameters: m, n, p=m*n, etc.
Generate population (represented by P frogs) randomly;
2. Evaluate fitness of each frog;
3. While (convergence criteria is satisfied)
4. {
5. Sort P frogs in descending order;
(Shuffled all frogs (construct n groups and each groups has m frogs);
6. for groups
7. {
8. Get the worst frog Qw and the best frog Qb in this group;
9. Qtemp =Qw(t)+c.rand.(Qbest(t)-Qw(t));
10. for each Dimension
11. If Qtemp.fitness < Qw(t).fitness
12. Qw(t+1)= Qtemp ;
13. Else
14. Qw(t+1)= rand.*Qtemp ;
15. End ;
16. end
17. get new frog using mutation;
18. if Qnew.fitness<Qw(t).fitnes
19. Qw=Qnew;
20. End
21.}
22.For each frog in population
23.{
24. Get two frogs Qa, Qb randomly from population;
25. Q1=Qa(t)+rand.(Qb-Qa) ;
26. Q2= Qa(t)+rand.(Qa-Qb) ;
27. If Q1.fitness<Q(t).fitness
28. Q(t+1)=Q1;
29. Else
30. Q(t+1)=Q2;
31.End
32.}
33.}

IV. SIMULATION AND RESULTS

SDN Simulation

SDN simulation has been carried out using the Matlab code to gather all datasets of inputs and outputs. Similarly, simulation experiments have been executed the usage of POX controller and tracking of the flows turned into required for all events.

Concerning which, as a way to construct the learning gadget, the SDN controller offers the orders to SDN switch to display the flows, and the transfer continues on monitoring them to discover the occasions. while the flows are monitored the complete event, whether or not they occur most effective often or periodically, are stored in the database to be used in the ANN studying machine, and consequently the information that are collected from the ANN studying gadget is taken into consideration as being an efficient enter to the optimization algorithms. This paper offers a brand new approach to minimize the burden of the SDN transfer by using making it changing adaptively by using the controller, instead of await the event all of the time.

SDN Optimisation

GA and SFLA were included separately with trained ANN model to select the finest set of inputs that makes the network to work efficiently as viable. Fig. suggests the architecture of the system which includes the trained well known ANN model as well as SFLA and GA as an optimizer.

The simulation experiments have been done by way of MATLAB platform. SFLA was employed to find the optimal structure of the network and the best operational parameters of this and the GA set of rules have been selected after sizeable simulations.

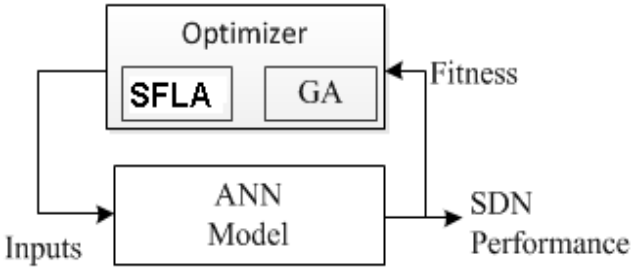

Figure 2. The architecture of the Proposed Method

ESFLA has outperformed GA regarding the overall performance and computational time, the convergence is quicker with fewer iterations and the obtained fitness is better.

Table 1. Performance and computation time comparisons for GA and ESFLA

Optimisation method	SDN Performance (%)	Computation Time (min)
GA	96.323	7.25
ESFLA	93.846	5.5

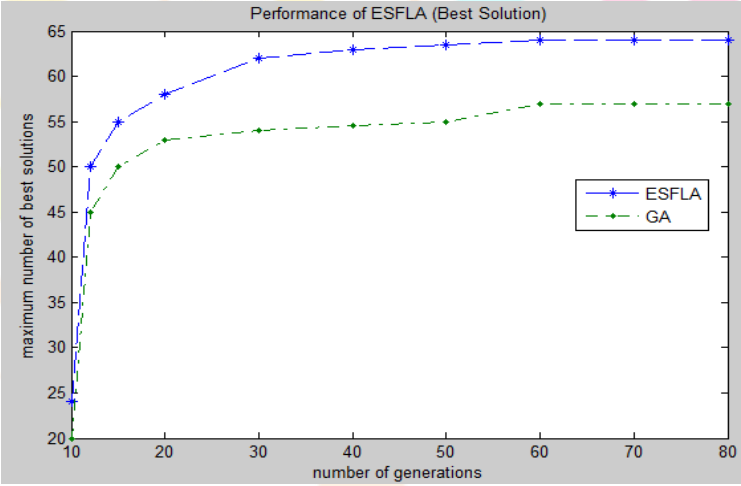

Figure 3.Convergence comparison of GA and ESFLA

CONCLUSION

The shuffled frog-leaping algorithm with local search can guarantee the feasibility of updated solution; but, the step size remains selected indiscriminately. To achieve an affordable step size, the adjustment factor and adjustment order are adopted. Moreover, associate external optimization is employed to exchange the data among individuals. The effectiveness of the enhanced shuffled frog leaping algorithm is evaluated employing a set of well-known instances. The results indicate that the results gained by ESFLA are usually an equivalent or slightly better than genetic algorithm. Moreover, the results conjointly show that the improved ways effectively improve the performance of the algorithm.

REFERENCES

[1] Y. Al-Dunainawi and M. F. Abbod, "Hybrid Intelligent Approach for Predicting Product Composition of Distillation column," Int. J. Adv. Res. Artif. Intell., vol. 5, no. 4, pp. 28–34, 2016.

[2] W. Xia, Y. Wen, C. H. Foh, D. Niyato, and H. Xie, "A Survey on Software-Defined Networking," IEEE Commun. Surv. Tutorials, vol. 17, no. 1, pp. 27–51, 2015.

[3] C. Gao, H. Wang, F. Zhu, L. Zhai, and S. Yi, "A Particle Swarm Optimization Algorithm for Controller Placement Problem in Software Defined Network," Springer International Publishing, 2015, pp. 44–54.

[4] Z. F. Zhang Shunmiao, "Survey on software defined network research," Appl. Res. Comput., vol. 30, pp. 2246–2251, 2013.

[5] A. Sgambelluri, A. Giorgetti, F. Cugini, F. Paolucci, and P. Castoldi, "OpenFlow-Based Segment Protection in Ethernet Networks," J. Opt. Commun. Netw., vol. 5, no. 9, p. 1066, Sep. 2013.

[6] S. Baik, Y. Lim, J. Kim, and Y. Lee, "Adaptive flow monitoring in SDN architecture," in 2015 17th Asia-Pacific Network Operations and Management Symposium (APNOMS), 2015, pp. 468–470.

[7] D. Kreutz, F. M. V. Ramos, P. Esteves Verissimo, C. Esteve Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-Defined Networking: A Comprehensive Survey," Proc. IEEE, vol. 103, no. 1, pp. 14–76, Jan. 2015.

[8] M. Azarkaman, M. Aghaabbasloo, and M. E. Salehi, "Evaluating GA and PSO evolutionary algorithms for humanoid walk pattern planning," in 2014 22nd Iranian Conference on Electrical Engineering (ICEE), 2014, pp. 868–873.