A MULTI-CRITERIA ROUTING ALGORITHM FOR SDN BASED DETERMINISTIC ADAPTIVE RENDERING TECHNIQUE

Ayotuyi Tosin Akinola[1], Matthew Olusegun Adigun[2], Pragasen Mudali[3]

Department of Computer Science, University of Zululand, P Bag X1001,
KwaDlangezwa, KwadZulu Nathal Province, Republic of South Africa.
ruthertosin@gmail.com.
http://<scholar.google.co.za/citations?user=c1XTnZEAAAAJ&hl=en>

Abstract

The sudden increase in the number of traffic flow in recent times is a pointer for the network engineers regarding the need for suitable mechanism that would take care of the network QoS demands. Such demands include the smooth running of big data, D2D video exchange, Voice over-IP and real-time multimedia which needed certain QoS requirements for optimal service. However, fewer research articles have reported on this challenges. We propose a multi-criteria routing algorithm that is based on deterministic Adaptive rendering technique called DART_MCP. Our DART_MCP algorithm deployed Dijkstra's algorithm to simplify the topology of the network before using multiple-criteria energy functions. The proposal was tested using a network simulator in comparison with other existing algorithms like, SA_MCP, H_MCOP and HSA_MCP. The experimental results show that our proposed algorithm recorded lower running time, higher success ratio rates and network interference reduction by 42.9%, thus achieving a faster network stability.

***Keywords*: Protocol; Multi-Criteria; ART; SDN; DART_MCP; Stability**

1. Introduction

Software-Defined Networking (SDN) has recently been a popularly recognised approach in the networking field as a better platform for a fast and easily deplorable networking system that addresses the most common challenges encountered in the networking field today [Zhu *et al.* (2015) and Zhu *et al.* (2006)]. Due to the spurious increase in the number of devices that access the internet especially mobile devices, the act of ubiquitous computing had been a common practice nowadays. Many real-time applications are commonly seen running concurrently which are highly resource consuming applications such as real-time multimedia, device-to-device video chat, and Voice over Internet Protocol applications [Akinola *et al.* (2018)]. Several existing Internet transmission approaches are still deploying Best-Effort single service which is not good enough or perhaps practically impossible to enhance the smooth running of the applications aforementioned. Similarly, OpenFlow has evolved as a core technology and a robust enabling approach for realizing a flexible control of network traffic in SDN thus, proving to be a reliable solution for the future Internet technology [Zhu *et al.* (2015), Nguyen *et al.* (2013)].

The marriage of the novel SDN architecture and OpenFlow have given an optimistic platform for a tailored networking service provisioning that is expected to meet up the need of future demands in terms of Quality of Service (QoS) requirements. By QoS, we mean the provisioning of several network QoS based on the best network state requirements for optimal performance of each application. Hence, QoS routing entails the consideration of both the network optimal path selection as well as the quality of the network based on the available resources to be able to make an appropriate decision for user demands [Akinola *et al.* (2015)]. This implies that the QoS routing for an SDN-based network environment needs to speak to both network extensions as well as the efficiency of the network flows in such a manner that the stability of the network is guaranteed over a specified range of time. Moreover, as the future network demands are gradually ascending its peak, the quest for fast deployable and efficient utilization of network resources becomes a challenge [Zhang *et al.* (2015)].

As several applications evolve and many are still yet to be released, it becomes more paramount that the QoS routing algorithm needs to be more equipped to be able to meet up the needs of network users. Asides, from the need to maintain good network qualities, several network traffic flow challenges also forestall the drop in the network qualities which can be checked through the routing algorithms. An example includes traffic flow interference which often results in network instability [Yeganeh *et al.* (2013), Song *et al.* (2017), and He *et al.* (2017)]. Furthermore, several kinds of literature have proposed various QoS routing algorithms which among the best include HAS_MCP, H_MCOP, and SA_MCP but most of these approaches are confined to a speculative

framework with the evaluations restricted only to algorithm evaluation concerning memory consumption and complexity computation [Nguyen *et al.* (2013) and Sheng *et al.* (2015)]. Moreover, none of these algorithms has featured the concern for network instability that arises from traffic flow interference within the networking environment [Sheng *et al.* (2015)]. It is easier to include several constraints into a routing algorithm however, the daunting task lies in how the bunches of network constraints (such as "*k*"), can figure out itself in addressing traffic flow interferences within the network channels.

These flow interferences are divided into two types which are intra or inter traffic interferences [Sridharan *et al.* (2017)]. Intra is within switches in the same slots while Inter is of different slots on the network. These interferences are caused by the sharing of the same network channels and or control flow paths. Thus, the QoS routing algorithm needs to be robust to be able to prevent the occurrence of network interferences thereby ensuring that the users' network optimal performance is guaranteed. This paper thus creatively introduces the adaptive rendering technique (ART)-enabled algorithm which is typically a computer graphics rendering approach coupled with deterministic multi-criteria energy function to enhance optimal QoS routing performance. This approach was typically deployed to be able to address the interference in network flow which impedes the network stability. Thus, this serves as one major contribution that our proposed QoS routing algorithm is unravelling to the body of knowledge in the Software Defined Networking field.
Thus, we highlighted the contributions of this article in this section through:

- Unravelling the impact of network interference in an SDN based network environment
- Critically analysing the problem of network instability as a bye product of network interference
- Analysing the impact and the possibility of addressing network instability with the use of the Adaptive Rendering Algorithm approach
- Deploying and evaluating the proposed approach in comparison with the earlier existing approaches.

The remainder of this article is organized as follows. Section II gives a brief explanation of the related works on the previous routing algorithms attempts while Section III provides some background investigation on the ART algorithm. Section IV discusses the ART algorithm with its brief details while section V depicts the explanation of the optimal solution and stability of the network. Section VI explains the detailed experimental simulation while section VII details the experimental results and discussions. Section VIII concludes the paper alongside its future works.

2. Related Works

SDN is one of the latest approaches for optimal network performance especially with increasing demands in network performance to meet up with the users' device requirements. Several attempts have been carried out by scholars to bring about the improvement in network users' experience. The work of Hilmi Enes Egilmez produces the LARAC algorithm [Egilmez (2012)]. The algorithm introduced a multiplier that represented two major concerned parameters being cost and delay to video streaming service. The algorithm was able to use the iterative method similar to Lagrange relaxation to select the optimal path for the network packets. It is one of the effective algorithms that address the constrained shortest path problem. In [Wang and Crowcroft (1996)], the authors propose an algorithm that uses a Multi-constrained approach to find the shortest path for routing in a network. However, the work does not discuss any optimization techniques to improve the proposed algorithm. The work presents three path computation algorithms and further opened up future works such as integrating the algorithm to address admission control and resource setup. The convergence speed after failure is also proposed to verify the degree of stability of the proposed algorithms.
The work of Chen and others in [Chen *et al.* (2014)] proposes Interference azimuth spectrum (IAS) and geometry-based stochastic models (GBSMs) as an important mechanism of an analytic framework that measures the network interference performances. Several numerical examples were depicted to demonstrate the usefulness of the mechanism, though the work proposes that other effects of interferences can result from specific multiple access and adaptive transmission schemes. The survey article in [Garroppo *et al.* (2010)] provides various routing algorithms varying from approximate to exact solutions. These algorithms derived their solutions using Multi-constrained optimal path problem (MCOP) that includes several metrics such as packet delay, available bandwidth, packet loss, and buffer overflow. Most of the review algorithms in the survey are restricted to a speculative framework and take little or no cognisance of the memory requirement or computational complexity of the proposed algorithms. Among such works include the work of Lee and others in [Lee *et al.* (1995)] that propose a Fallback algorithm of one main constrain. This algorithm serially considers other constraints if it meets the requirement else it continues the search. The major issue with this approach is that it does not guarantee the optimal routing path and also, it cannot be used for an autonomous network environment where attaining an optimal solution becomes non-negotiable.
Moreover, a novel QoS provisioning architecture called PRICER was developed to enhance QoS routing update (ROSE) as well as to promote effective pricing incentive for routing algorithm (PIRA) [Cheng *et al.* (2008)]. This

proper incentive was carried out by integrating an efficient pricing function into the QoS routing algorithm. Extensive simulations, as well as the conducted theoretical analysis, proved the better performance of the proposed architecture over the existing state-of-the-art approaches with an added contribution in terms of evaluating the staleness of the network link-state information.

Gang Liu and K. Ramakrishnan propose a heuristic algorithm called Heuristic Multi-constrained optimal path problem (H_MCOP) [Liu and Ramakrishnan (2010)]. This algorithm provided a cost function that helps to prune up the paths that violate the candidate path list thus achieving an optimal path solution. The algorithm also uses Dijkstra's shortest path approach to predict the pruning and ordering process thus making the algorithm somehow intelligent towards arriving at a precise solution. The enhancement of the prune algorithm with any of the established ϵ-approximate algorithms to achieve BA*Prune, therefore made this approach to be a comparable solution to some of the best known polynomial-time ϵ-approximate algorithms. However, the proposed algorithm is regarded as being classical and needed more improvements to achieve optimal performance.

Several other similar QoS routing solutions have left out the information of the link-state in the process of deriving an optimal network path by assuming the state is accurate and readily available. Examples of these algorithms include the general simulated annealing multi constrained path problem (SA_MCP) which happens to be an expansion of local search algorithm in [Sheng *et al.* (2015)] as well as the upgraded version of it called Heuristic simulated annealing multi constrained path problem (HSA_MCP). The upgraded version was able to enhance better optimization process to be executed faster than the former but however not tailored to enhance a stable network [Sheng *et al.* (2015)]. The work of Apostolopoulos and others in [Apostolopoulos *et al.* (1998)] incorporate different update policies to check the link state such as equal class, threshold, and exponential class policies. It was derived that with a given threshold value (τ), the update link signal is triggered under the threshold policy used such that when $|l_c - l_o|/l_o > \tau$, with the l_o being the least advertised value among the available latency and l_c appears to be the current latency of the link.

However, among all the highlighted literature, none of them had addressed the impact of network interference within a network hence there is a need to establish stability in a network that is designed to meet up the varying users' network requirements. Considering various approaches that try to optimize the network requirements that are needed by users, we try to extend the knowledge body along with network routing through integrating the network stability measures into the routing technique that we propose to optimise the QoS routing algorithm. To achieve this, we deploy Dijkstra's algorithm to simplify the network topology for easy analysis and then make use of the multi-criteria energy function to deploy the network requirement constraints while the stability evaluation concerning interference was implemented into the Adaptive Rendering Technique approach to optimise the users' need.

3. Background Investigation

We try to investigate the background of network topology as well as the concept behind the proposed topology resolution for network analysis. This lay the foundation for our proposed multi-criteria network user requirement.

3.1. *Network Description*

Let $G = (V,\ E)$ be a weighted digraph, with weight function w : E → **R** mapping edges to real-valued weights. If $e = (u, v)$, we write $w\ (u, v)$ for $w(e)$. With the above, we denote the path length p of a network as $p = < v_0, v_1, .., v_k >$ which gives the sum of the weights of its constituent edges which is represented in equation 1 [Korkmaz and Krunz (2001) and Zhang *et al.* (2014)] as:

$$L\ (p) = \sum_{i=1}^{k} w(v_i - 1, \qquad v_i) \tag{1}$$

Hence, the distance from u to v, denoted by $\delta(u, v)$ provided us with the minimum length path if there is a path from u to v, and it is ∞ if else. Thus, taking a reference estimate from $d[v]$ of the length $\delta(s, v)$ of the shortest path for each vertex v. We define $d[v] \geq \delta(s, v)$ and $d[v]$ always equal to the length of a known path. This implies that at the initial state, $d[s] = 0$, and all the $d[v]$ values are set to ∞. The validation of the real shortest distance is confirmed on a condition at which $d[v] = \delta(s, v)$. Therefore, combining the path from s to u with the edge (u, v), we obtain another path from s to v with length $d[u] + w(u, v)$. If $d[u] + w(u, v) < d[v]$, we therefore replace the old path $< s, \dots, w, v >$ with the latest short path $< s, \dots, u, v >$. Hence the path is updated.

We also introduced an energy constraint function which depicted the range of constraints that are combined for efficient path required to be used. Considering the path energy function (E) as expressed in equation 2:

$$f(E) = max_{k=1}^{K}\left(\frac{w_k(E)}{c_k}\right) \quad (2)$$

where $c = (c_1, c_2, \ldots c_k)$ represents the k QoS constraints for feasible path requirement. Several constraints like packet overheads, network delay, transmission failure, flow setup time, and many others depending on the network requirement for the particular application in question can be deployed.
Therefore, in streamlining the network topology concerning the energy function which inculcates the parameters of the constraint, we defined the shortest path from a source node to a sink node as shown in Figure 1 such that

3.1.1. $\forall p \in P', W_i(p) \leq C_i \; for \; i = 1, \ldots, m$

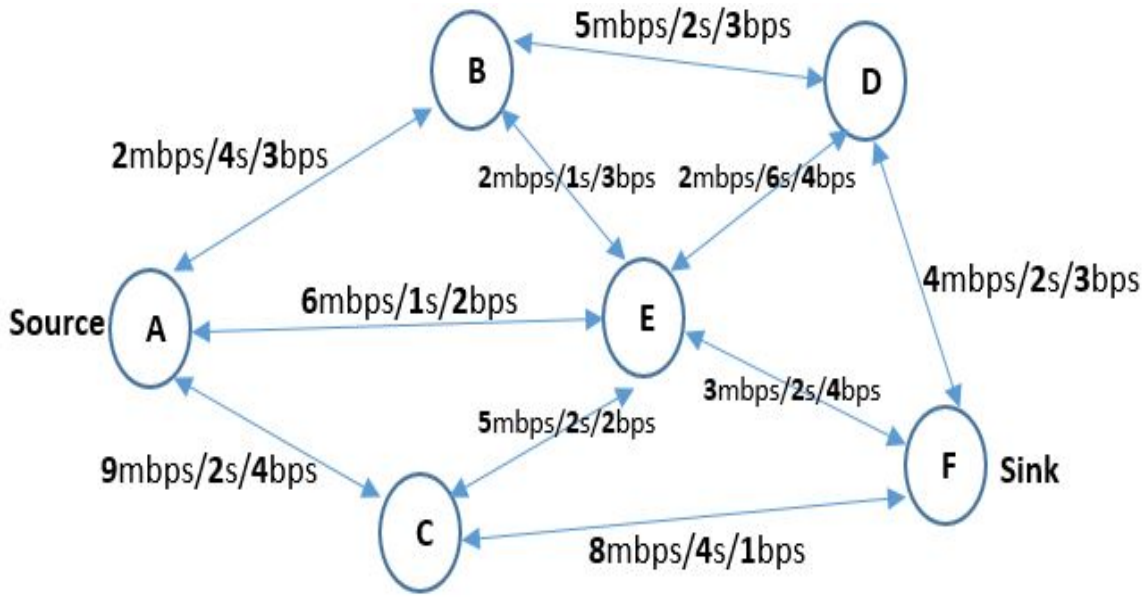

Fig.1: Skeletal topological network routes with varying requirements from source to sink. (space constraints)

3.1.2. $W_k(P)$ *the is minimized over all feasible paths concerning the condition in 3.1.1 above, thus:*

We have the expression in equation 3 as:

$$w(p) = \sum_{k=1}^{k} w_k(p) / C_k \quad (3)$$

Hence, the goal is to determine a path that considered all the constraints for the network user requirement in such a manner that the minimum requirement is not exceeded. In that case at any moment $(p) > k$.

3.2. *Adaptive Rendering Technique (ART)*

The adaptive rendering technique specifies the approach used in a graphic computing system to temper the quality of a picture to the required pixel and aspect ratio that is needed such that it looks the same or almost exactly as the real object. Several algorithms are deployed towards achieving the technique of which the steps within the algorithms were Transformation, Rasterization, Fragment shading and Frame stability. The process is termed adaptive because significant noises were removed from the graphics with almost similar fidelity maintained in the course of transformation and reconstruction. One of the main goals for deploying an adaptive rendering approach is to maintain the speed of frame generation while ensuring the realism of the object, on the other hand, thus introducing a measure of trade-off in the algorithms. There are a series of stages in which the algorithms undergo to achieve this maximal speed and realism that we deployed to address low latency networks with a degree of network success rates (throughput).

Firstly, we looked into various forms of physically-based rendering algorithms. These algorithms are used in solving the global illumination problem in computer graphics [Lafortune and Willems (1994)]. The algorithm determines the complex inter-reflection of light through a particular scene which is similar to the situation that we have in a typical network environment where network interference becomes a significant challenge in addressing user network requirements provisioning. The global problem is addressed by the physically-based rendering algorithms which are classified into two major types thus:

3.2.1. Object-Based and Image-Based Algorithms:

The object-based algorithms solve the global illumination problem independently of the actual parameters. Its key strength has to do with the deployment of the hierarchical algorithm, use of higher-order basis functions as well as some kind of wavelet algorithms for his implementation. On the other aspect, the image-based algorithms considered the actual parameters of the image like the stochastic ray tracing algorithm which considered each pixel for radiance value computation. However, based on performance, the speed of generation of the interactive frame is a bit slow in this classification thus its deployment in our solution approach will not be favourable.

3.2.2. Deterministic and Monte Carlo Algorithms:

The Deterministic approach uses classical numerical techniques to address the global illumination problem. It uses the combination of solution methods for large sets of linear equations and some cubature rules for integrating high-dimensional functions [Lafortune (1996)]. On the other hand, the Monte Carlo rendering algorithms deploy stochastic techniques to work out the high-dimensional integrals of the global illumination problem [Zhang *et al.* (2014)]. Generally, the rendering algorithms typically follow three major steps when used to address our type of problem through the rendering equation. These steps are:

- Identification of the physical (network) problem
- Formalizing the problem in a mathematical model and
- Developing the algorithm to solve the designed model.

4. The ART Algorithm

Both the integrity and the transparency of the local area network user requirements become important in the network system. This necessitates that the overall network maintenance is inevitable through having a global view of packet routing in SDN. This global view, therefore, enhances the maintenance of the network system via the network controller. The need for the global view for an SDN network, therefore, forms the bedrock for the deployment of the Adaptive Rendering Technique (ART) algorithm for optimal network packet routing. The ART that was used in this work is primarily the deterministic approach which was built on a classical numerical value technique. The development of this approach first analyses the network topology by using Dijkstra's algorithm as well as the multi-criteria energy function which introduces the constraints that were needed to be met for the user requirements. Thereafter, we applied the DART_MCP algorithm to address various network user requirements. This algorithm is as illustrated in Algorithm I.

DART_MCP Algorithm I.

```
Dijkstra(s, v, graph)
q ← empty
visited ← empty
for node ← graph.nodes do
        width[node] ← 0
        previous[node] ← null
end for
width[s] ← ∞
q.add(s)
while q is not empty do
        node ← q.dequeue()
        if visited.contains(node) then continue
        end if
visited.add(node)
        for link ← graph.connectedTo(node) do
        neighbor ← link.getDestination()
        altWidth ← max(width[neighbor],
```

```
                min(width[node],link.getWidth())
        if altWidth > width[neighbor] then
                width[neighbor] ⟵ altWidth
        previous[neighbor] ⟵ node
        q.add(neighbor)
                end if
        end for
end while;

getAnalysePath(packet)
token  ⟵ extractMPTCPToken(packet)
if exists(cachedPaths[token]) then
        p ⟵ dequeue(cachedPaths[token])
        install(p)
        return
end if
linkCapacity ⟵  topology.getAvailableCapacity()
for i ⟵ 0 to maxPaths-1 do
        p ⟵ dijkstra(flow.src, flow.dst,linkCapacity)
        w ⟵ width(p)
        for l ⟵ p.links do
                linkCapacity[l] ⟵ linkCapacity[l]-w
        end for
        cachedPaths[token].add(p)
end for
p ⟵ dequeue(cachedPaths[token])
install(p))

Include_Constraints (G(V, E))
For each node u in G
if (g(u)+r(u)>K)
        remove node u;
        remove link (u,v);
g* = min_{v∈NB} max_{k=1}^{K}([g_k(u) + d_k(u)]/c_k)
for each u in NB
E(u) = ma(g(u) + d(t) + g*(e)
Z = ∑_{u∈NB} [−E(u)/t]
X~uniform(0,1)
sum = 0
for each u in NB
sum = sum + [−E(u)/t]
if sum > x THEN return u;
Reduce(u, v)

DART_Optimalrequire()

procedure rendering takes s_k as Switch k
        source: Netstab
        pt: InterferenceStability
returns
        flows: globalstab
        stability: Isoefficiency
begin
        (α_k, β_k, γ_k) ⟵ Map(pt, Isospeed-Isoefficiency)
        (σ_x, σ_y) ⟵ InterferenceStab(source.Netsize)
        Netpos ⟵ (σ_x, σ_y, 1, 0)
        Netpos ⟵ Map(Netpos, Isospeed-Isoefficiency)
```

if source.at-stability **then**
interferencesq ← 1
else
interferencesq ← |Netpos − edge| ²
endif
stability ←Isoefficiency (Netpos − pt)
$(\alpha_p, \beta_p) \leftarrow (\alpha_k - \sigma_x * \gamma_k, \beta_k - \sigma_y * \gamma_k)$
list ← source.Monte-Carlo(zz)buffer$[\alpha_p, \beta_p]$
atten ←source.packetflow(stab.ility)/interferencesq
flows ← atten ∗ source.globalstab
while list ≠ nil
and not TooCongested(flows) **do**
tile ← list.first; obj ← tile.obj
if tile.γmin < $\gamma_k - \in$
and (tile.packdrop obj.HitTest$(\alpha_p, \beta_p, \sigma_x, \sigma_y)$)
then
γlist ← obj.Compute$\gamma_k(\alpha_p, \beta_p, \sigma_x, \sigma_y)$
for each γ'_k in γlist **do**
if $\gamma'_k < \gamma_k - \in$ **then**
$pt' \leftarrow (\alpha_p + \sigma_x * \gamma'_k, \beta_p + \sigma_y * \gamma'_k . \gamma'_k)$
$pt' \leftarrow$ Map(pt', Isospeed-Isoefficiency)
alpha ← obj.GetAlpha(pt', stability)
flows ← flows ∗ ((1, 1, 1) − alpha)
end if
end for
end if
end while
end procedure

Symbol	Meanings
α_k	Number of *Controller-to-Switch* messages sent from a controller to S_k
β_k	Number of *Asynchronous* messages sent from S_k to a controller
γ_k	Number of *Symmetric* messages sent between a controller and S_k
σ_x	Number of *stat-requests* messages periodically sent from a controller to S_k
σ_y	Number of *flow_mod* messages sent from a controller to S_k
α_p	Number of *flow_removed* messages sent from S_k to a controller
β_p	Number of packet in messages sent from S_k to a controller for local flows
γ'_k	Number of *packet_in* messages sent from S_k to a controller for global flows
$\in$	Number of *echo* messages sent between a controller and S_k
$E(u)$	Energy of the node
Z	The normal factor
$g*$	The local minimal energy
$g(u)$	Positive linear mark of node u
$g_k(u)$	The *k* constant parameter of the path from M and Z controllers
$d_k(u)$	The *k* constant parameter of the path from Z and L controllers
$r(u)$	The *k* constant parameter of the path between intra and inter switches

Table 1: Meaning of the Notations in DART_MCP Algorithm.

Thus, in the SDN network flows, three main types of messages are sent within the network environment which are transferred between the data plane elements and the network controllers. These are majorly symmetric

messages, asynchronous messages, and controller-to-switch messages [Karakus and Durresi (2016)]. These messages were deployed into the rendering algorithm 1. However, as their names imply, the symmetric messages are typically the echo and hello messages which do not need any solicitation from a controller. Asynchronous messages are typical messages that are sent to the controller in response to the reception of packets by the switches such as flow_removed and packet_in messages. The last and commonest type of message is the controller-to-switch message which is responsible for delivering messages to the switches without any necessary acknowledgment.

Furthermore, the DART_MCP algorithm contains the Monte-Carlo (ZZ) buffer which supposedly assists in establishing the stability of the network performance concerning the effect of network interferences. The buffer approaches the versatility of distributed networks as it can render a wide influx of transferred packets while maintaining the QoS of varying network users. This is carried out in two major phases which are scan conversion and the rendering phases [Zhang et al. (2014)].

4.1. *Scan Conversion*

The algorithm at this phase estimates the range of packets that are sent over the networks to harness the scheduling of the paths for preventing interference. Thus, considering the general deterministic Software-Defined enabled Network, which is measured from the source in the presence of n interferes at distanced $r_i > 0$ that are active with the probability p independently of each other. The path loss law is the standard power-law (considering normalized paths – edges (r)) i.e $e(r) = r^{-\alpha}$.
Considering the location of the controllers being predefined and fixed, we assumed that there is a path available for a switch to send packet-in messages whose paths are computed by using the earlier depicted Dijkstra algorithm. The major problem we are addressing is to alleviate the sharing of flow paths and network channels, thus the network channels are subject to Rayleigh fading which is exponential in power fading rate. Therefore the power is represented by P_{ri} that is caused by the rate of packet traffic i, with the distributed exponential mean $r^{-\alpha}$, then we express the probability of density function of P_{ri} as

$$fP_{ri}(x) = r_i^{\alpha} \exp(-r_i^{\alpha} x), \quad x \geq 0. \tag{4}$$

Therefore, the scanning conversion phase determines the overall interference by the inter and intra flow packets within the network as

$$I_n = \sum_{i=1}^{n} B_i P_{ri} \tag{5}$$

where the Bernoulli random variables B_i is with the varying parameter p. Hence, the Laplace transform of this exponential random variable of the transferred packets with mean 1/y is $\int(s) = \frac{y}{y+s}, s \geq 0,$ so we have,

$$\int_{In}(s) = \prod_{i=1}^{n} \left(\frac{p r_i^{\alpha}}{r_i^{\alpha} + s} + 1 - p \right) \tag{6}$$

$$= \prod_{i=1}^{n} \left(1 + \frac{p}{1 + \frac{r_i^{\alpha}}{s}} \right), s \geq 0 \tag{7}$$

in the above expressions, there is a need to maintain a proper network packet distribution especially when the number of network nodes n seems to be relatively infinite. To this effect, equation 7 portrays a uniform convergence as it converges to some positive limit such that for any s > 0 if and only if

$$\sum_{i=1}^{\infty} \frac{p}{1 + \frac{r_i^{\alpha}}{s}} < \infty \, (\forall s > 0) \iff \sum_{i=1}^{\infty} \frac{p}{r_i^{\alpha}} < \infty. \tag{8}$$

Thus, the equation in 8 gives an even distribution for the network packets but the interference of the network is infinite if we have

(9)

$$\sum_{i=1}^{\infty} \frac{p}{r_i^{\alpha}} = \infty,$$

Hence for a one-dimensional network setup, the scan conversation is determined while $r_i = i,\ i \in \mathrm{N}$, and $p > 0$, we achieved an interference of $\alpha > 1$ for a finite network and $\alpha \leq 1$ for an infinite network.

4.2. *Rendering*

For the rendering phase, considering an SDN-based network whose Rayleigh (amplitude) of fading due to interference varies toward infinite one-sided one-dimensional network path $r_i = i,\ i \in \mathrm{N}$, allowing a close expression for $\int_I(s)$ Laplace expression. Hence, for $\alpha = 2$ and more, the expression is thus,

$$\int_I(s) = \frac{\prod_{i=1}^{\infty}\left(1 + {(1-p)\,s}/{i^2}\right)}{\prod_{i=1}^{\infty}\left(1 + {s}/{i^2}\right)}, \qquad s \geq 0. \tag{10}$$

Recalling the Euler's product formula, this can be applied here to simplify further the Laplace expression in equation 10 thus,

$$\sin(\pi z) \equiv \pi z \prod_{i=1}^{\infty}\left(1 - {z^2}/{i^2}\right),\ z \in \mathrm{C}, \tag{11}$$

While the denominator carries z = $j\sqrt{s}$ and the numerator that is equal to z = $j\sqrt{(1-p)s}$, we can rewrite the denominator as

$$\prod_{i=1}^{\infty}(1 + s/i^2) = \frac{sin(\pi j\sqrt{s})}{\pi j\sqrt{s}} = \frac{sinh(\pi\sqrt{s})}{\pi\sqrt{s}}, s \geq 0, \tag{12}$$

thus, we derived the expression,

$$\int_I(s) = \frac{1}{\sqrt{1-p}} . \frac{\sinh\left(\pi\sqrt{s(1-p)}\right)}{\sinh(\pi\sqrt{s})}, \qquad s \geq 0. \tag{13}$$

Hence, for a higher degree of interference amplitude, we render a more reliable equation to enable listening nodes for packet scheduling accordingly with the expression:

$$\int_I(s) = \frac{\prod_{i=1}^{\infty}\left(1 + {(1-p)\,s}/{i^4}\right)}{\prod_{i=1}^{\infty}\left(1 + {s}/{i^4}\right)}, \tag{14}$$

And the value of α , in this case is, 4. By factorization, we can use Euler's product formula again since the expression in equation 14, states $(1 - z^4/i^4) = (1 - z^2/i^2)\,(1 + z^2/i^2)$. The factorization therefore derived these expressions $z = \sqrt{\pm j s^{1/4}}$ with $z = \sqrt{\pm j\left((1-p)s\right)^{1/4}}$ for the denominator and numerator respectively. Therefore, the resulting expressions give complex conjugates, resulting into

$$\int_I(s) = \frac{cosh^2\left(\sigma(1-p)^{1/4}\right) - cos^2\left(\sigma(1-p)^{1/4}\right)}{\sqrt{1-p}\,(cosh^2\,\sigma - cos^2\,\sigma)} \tag{15}$$

Thus, equation 15 gives the resultant rendering expression in any network scenario where the impact of interference is more experienced. This equation derived gives the solutions model that DART_MCP algorithm analyses the network environment to ensure its stability. The next section discusses various expressions used in determining the optimality of a typical SDN-based network and well as its stability through the network stability function (rendering equation).

5. Optimal Solution and Stability of the Network

One major goal of proposing the DART_MCP algorithm among the various available ones in the networking environment is to enhance an optimal performance in terms of rate of network stability when tailored user requirements are to be harnessed. We typically observed that little or no reports have been published on the challenge of network stability with reference to its routing algorithm. Therefore, it becomes paramount to reduce the number of packet failures and path redundancy in the course of packet transmission. The process of addressing this particular challenge brought about the conception of implementing a stable network via the routing algorithm model that addresses this issue [Akinola *et al.* (2020)].

In this section, we deploy a mechanism similar to the ticket-based probing TBP used in [Chen and Nahrstedt (1999)] however we deploy a network interference alleviation scheme to enhance network stability while meeting up with user requirements. This network interference alleviation was built on the rendering equation to input network stability. The equation was written thus:

$$L_o\,(x, w_o, \lambda, t) = L_e(X, w_o, \lambda, t) + \int_{\Omega}^{1} f_r\,(x, w_{i,} w_{o,} \lambda, t) L_i(x, w_{i,} \lambda, t)(w_i . n) dw_i \qquad (16)$$

Where $\boldsymbol{L_o\,(x, w_o, \lambda, t)}$ equals to the total outgoing packets from various network hosts with bandwidth **"λ"** directed in a Poisson distribution manner of "w_o" through time **"t"** on a path distance of **"x"** away.

$\boldsymbol{\lambda}$ represents the bandwidth
$\boldsymbol{w_o}$ represents the poisson distribution value
$\boldsymbol{t}$ represents the time for packet delivery
$\boldsymbol{L_e(X, w_o, \lambda, t)}$ represents outward packet distribution
$\boldsymbol{\Omega}$ represents units of packets transmitted through mean network n containing all possible values of w_o.
$\int_{\Omega}^{1} \ldots . dw_o$ is an integral value over Ω
$\boldsymbol{f_r(x, w_{i,} w_{o,} \lambda, t)}$ is the bidirectional poisson distribution of packets whose proportion varied from w_i to w_o over distance x, time t, and bandwidth λ.
$\boldsymbol{w_i}$ is the inverse packet flow from controllers to hosts
$\boldsymbol{n}$ is the mean controller equidistance apart
$\boldsymbol{w_i . n}$ is the weakening interference factor of packets as it transverse the network.

Hence, equation 16 expresses the solution to the DART_MCP algorithm model whose analysis enhances network stability through the elimination of network interference. Our idea is that once the solution to a single interference is determined, it is sufficient enough to address similar multi-objective optimization problem efficiently, then the summation of such singular solution gives the aggregate of the larger network interference problem, hence, the stability of such network can be guaranteed based on the aggregate solution.

We formulate the users' requests as a Poisson distribution requesting for network resources and we considered this as an optimization problem that needed to find a balance between the network resources and user requirements. We look at a Poisson cluster process which is motion-invariant requests, thus from a single user point process. For practical illustration using Figure 1, the request is sent from the source to the sink however with the best route at any point in time which meets the user requirements. The throughput, latency, and bandwidth of the routes were specified on the routes connecting one node to the other thus depicting the state of network routes at timeslot *t*.

6. Simulation Environment and Experiment

This work uses a simulation methodology to carry out the performance of the proposed DART_MCP algorithm. We deployed the gt-itm application that was used in [Modelling (2014)] to produce the required network topology. The performance was tested using the number of success rates achieved and the duration of time at which the algorithm performs the simulation (running time). In this particular experiment, we shall be considering the performance of the DART_MCP with the other earlier mentioned algorithms such as the H_MCOP, HAS_MCP, and SA_MCP. Using the diagram in Figure 1 as a typical network topology with various links whose current network states were specified. We, therefore, set the increasing network sizes to range from 300, 600, 900, 1200, and 1500 nodes.

We also evaluate the performances of these algorithms when there is an increasing number of packet transfers and the packet size was kept constant as well as when the packet size was varied with a constant packet transfer rate within the network. The success ratio is determined through the ratio of running time to the success times with the product of the total times. This helps in calculating the feasible paths through random selection of the 300

nodes in the considered network topology. Considering the number of QoS constraints *C* to be equal to 4 such as Latency, Response time, Throughput and Bandwidth thus, the performance of the algorithm is supposed to be able to respond to the requirement of the network node as expressed in various weight ratios in the network expressions as expressed in equation 17 thus:

$$g(u,v) = \sum_{k=1}^{K} a_k \,.\, w_k(u,v).$$

The highlighted constraints were deployed as conditions that are expressed in a binary notation whose total weights amounts to 4 different weights of 1:0:0:0, 1:1:0:0, 1:1:1:0, and 1:1:1:1. Thus, keeping the sample constraints to be 4, we also defined several iterations of a similar value of 4 to verify the level of stability of the results that are produced. We also considered the relationship that is existing with the success rate of the DART_MCP algorithm of various parameter weight ratios having the possibility of increasing the number of constraints if need be to be more than 4 by simply following the pattern used in 4 constraints. Finally, we determine the duration of delay experienced in attaining a stable state in the network.

7. Experimental Results and Discussions

The results of the tests carried out on the proposed algorithm were discussed in this section. We first carried out a test on the earlier specified number of network sizes to determine the success of various weight ratios 1:0:0:0, 1:1:0:0, 1:1:1:0, and 1:1:1:1. Considering the diagram in Figure 2, the success rate that was attained was purely 100% for the network size of 300 nodes. It was clear from the diagram that the success rate of the weight ratio 1:0:0:0 and that of 1:1:1:1 are 100%. All the weights were 100% when the network nodes were 300, showing the fact that all the algorithms were able to perform efficiently when the constraint was just one. However, as the number of considered constraints was 2 or 3, the performance of the SA_MCP and HAS_MCP both were not outrightly efficient due to increasing network sizes. Asides, this is also due to the fact that the weight ratio affects the performance of the algorithm, such that the summation of the constraints in terms of the energy function of each algorithm is directly related to the performance as well. This implies that the higher the energy function of the weight ratio combinations, the more efficient it is at achieving the tasks at hand.

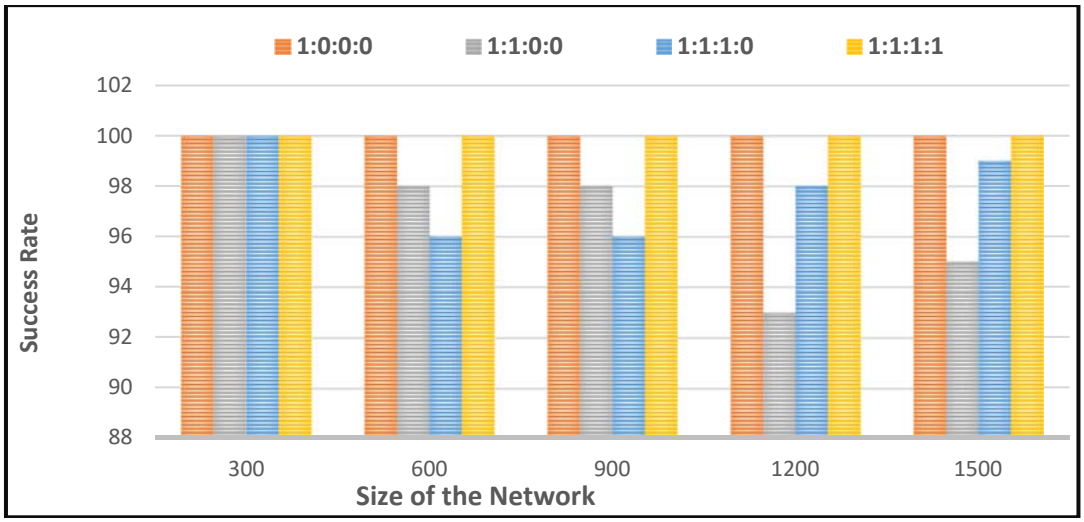

Fig.2: Success rate of the weight ratio of different network parameters.

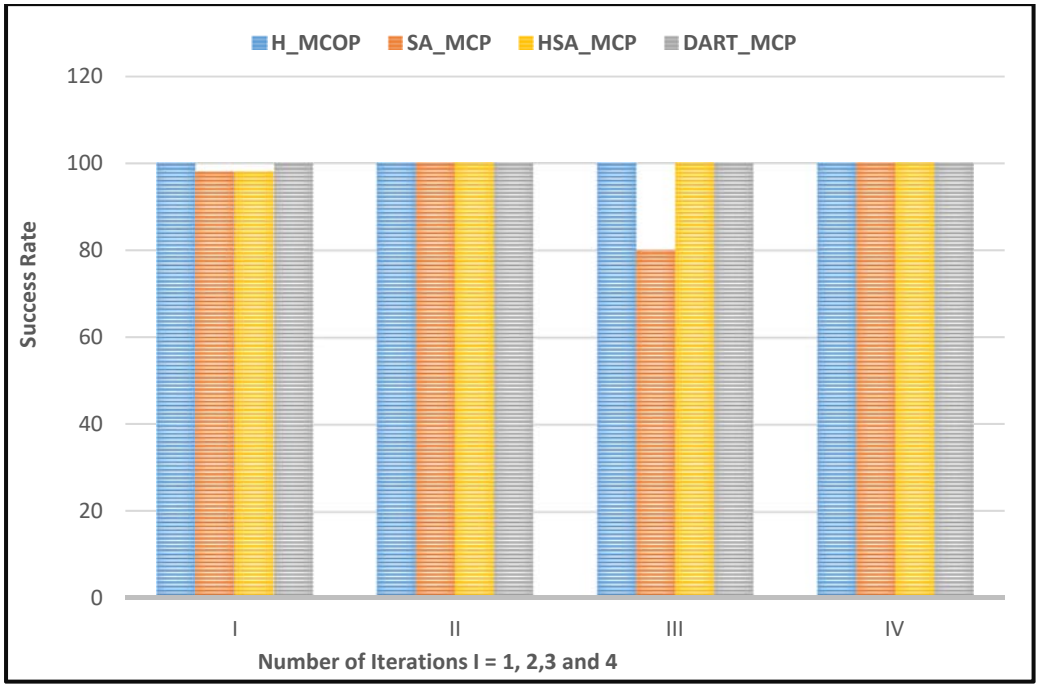

Fig.3: Output of Success rate of algorithms under 300 nodes.

However, we can briefly conclude that low constraints enable the efficiency experienced in 1:0:0:0 whereas dominant energy in 1:1:1:1 promoted the 100% efficiency that was recorded. Considering the goal of attaining better stability in networks, we have to consider varying constraints in terms of fluctuating user requirements as well as increasing network size which must be significantly spelt out, thus the 1:1:1:1 case was used for this study. The diagram in Figure 3. depicts the performance in terms of the success ratio of the various algorithms under the network size of 300 nodes. Having understood from Figure 2 that the network constraints and the weight ratio are both important in ensuring optimal performance in a network environment, we, therefore, test the following experiments from that in Figure 3 with weight ratio that included all the constraints as this seems to be the peak of the tasking scenario that could be evaluated based on this experimental setup. It can be deduced under this setup that the H_MCOP algorithm also attained 100% in all the cases involved. This happened as a result of the fact that we have a limited number of network nodes in terms of just 300 nodes only. We envisaged that as the number of network nodes increases, there might be a challenge with the performance of H_MCOP due to a lack of energy in-depth to handle multi-criteria network requests.

Moreover, at iteration III, the performance of SA_MCP accidentally falls to around 80% probably due to insufficient energy function needed to complete the tasks in the timeslot giving hence the reduced efficiency that was realised. Altogether, we can conclude based on the evidence that all the algorithms perform well with iterations II and IV attaining 100% for all the algorithms. Considering the diagram in Figure 4 under a network node of 900, the impact of the increased nodes promotes more network traffic flows to be experienced. This influences the need for the network to be more stable in handling the increased network flow. The obvious perception from Figure 4 typically showed that the performance in terms of the success rate of other algorithms is unstable and non-efficient. It is important to state at this point that the interference that exists in the network environment is one key factor that dictates the performance of the entire system. The DART_MCP algorithm was developed such that the rendering step enhance its best performance especially when the number of influx flows was increased. Even though none of the algorithms were able to attain a 100% success rate, we deduced that our proposed algorithm was able to manage the situation well and still record an appreciable number of the success rate of 98% above other algorithms. We finally put the algorithm to test with an increased number of network nodes up to 1500 nodes. This is really a large number of traffic flows in which the routing algorithms have to be able to adapt with the traffics as well as the demands depending on the combining quality of service constraints that are required. The success rate of DART_MCP was relatively stable from the third iteration as shown in Figure 5 and looking through the previous experiments, it is also clear that on the third iteration, the performance of all the algorithms is stable on average in an efficient manner. Therefore, we can conclusively say that while the actual performance of the algorithms is determined on the third iteration, the success rate of DART_MCP was outstanding amongst the other algorithms in general and this was due to its ability to address network control path and co-channel traffic interferences which were catered for in the algorithm development.

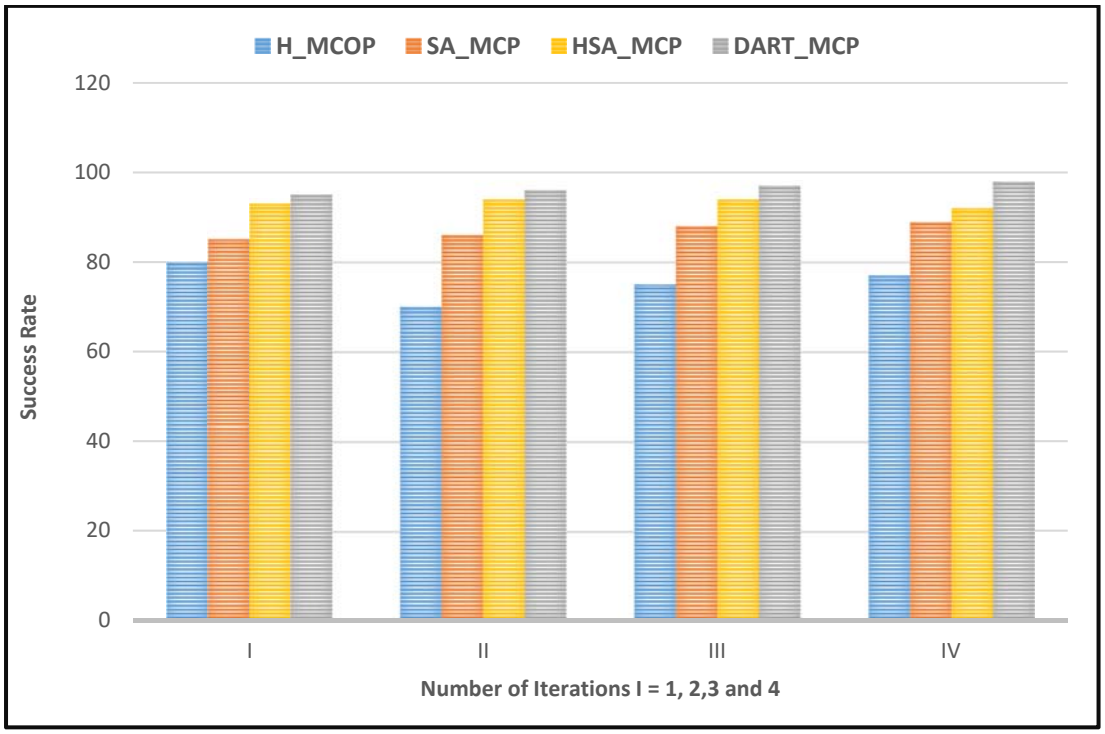

Fig.4: Output of Success rate of algorithms under 900 nodes.

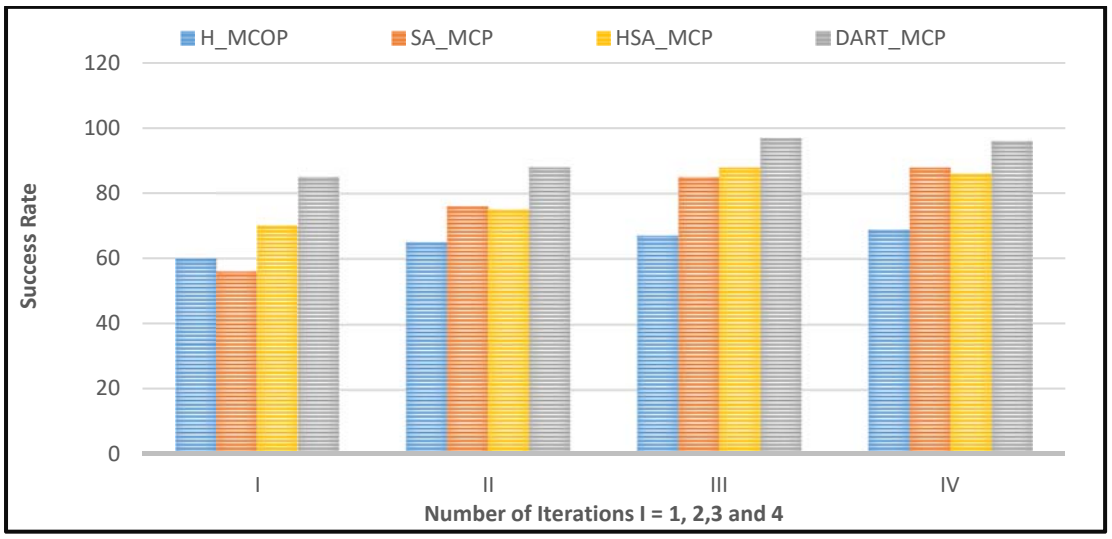

Fig.5: Output of Success rate of algorithms under 1500 nodes.

When determining the rate of a network running time, it becomes necessary to find out the varying time it takes between increasing packet transfer rate while the size of the packet is kept constant and vice versa. Figure 6 depicted the relationship between the running time for an increasing number of packet transfers and the size of the transferred packets. The results in the figure depicted that there is an increase in the running time when the rate of packet transfer is increased and the packet size is constant. This implies that there is more time spent to set up the network topology in this first scenario where the packet size was constant. This was due to the heavy congestion that takes place in the network as a result of the heavy transfer. On the contrary, the lesser running time was spent with the increasing packet size while the rate of packet transfer was kept constant. This is actually expected and it is proof that the network is in good condition for the running time test that is about to be carried out.

The running time depicted the time it takes the algorithms to complete the task of packet transfers. We performed (at least 4 iterations) on each of the results that were derived. Considering the result in Figure 7, under a network size of 300 nodes, due to the fewer network traffic, the H_MCOP gives a relatively low but consistent running

time. These results showed that the fewer network nodes enhance H_MCOP to perform better in the rate of running time consumption at almost all four iterations. DART_MCP often takes more time at the initial routing stage due to the computation complexity however, this becomes more of an advantage as the number of nodes increases.

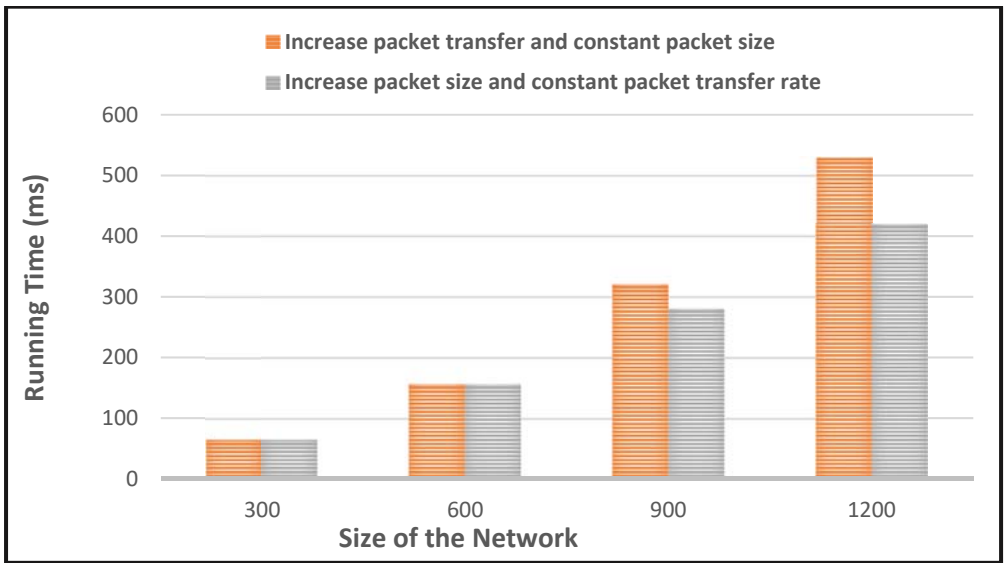

Fig.6: Determining the rate of running up the topology virtual network

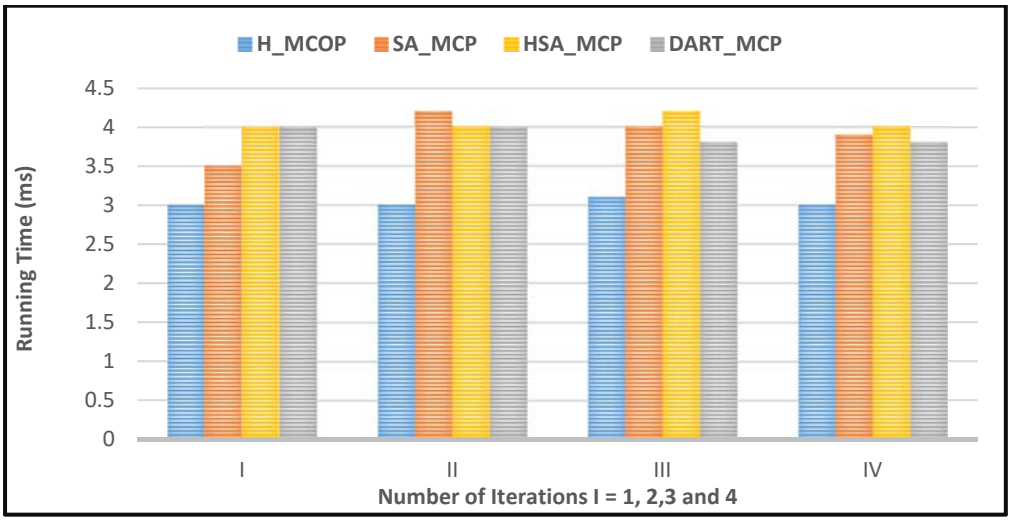

Fig.7: Output of the running time performance under 300 network nodes.

The diagram in Figure 8 depicted the performance of these algorithms under 900 network nodes samples. With these experiments all running under the same traffic flow rates, it brings along with it the benefit of the features of the DART_MCP algorithm. DART_MCP was able to attain an average of 87ms at both III and IV iterations in comparison with other algorithms.

The diagram in Figure 9 depicted a collection of 1500 network notes with four iterations. Overall, DART_MCP was able to quickly stabilize the performance to an average of 298ms over the iterations. Hence, we can conclude from the shreds of evidence that our proposed algorithm was able to optimize the performance of the network better especially with increasing network size. Although the effect of our proposed algorithm is better evaluated with a large number of nodes, it becomes paramount that DART_MCP maintains good performance through addressing the challenges of interference and control path issues as related to large traffic flows in SDN.

Moreover, the number of iterations does not really affect the performance of the algorithms. The first iteration is a little bit different sometimes due to the update of the routing table with the newly added number of network nodes, hence with the third iteration, a reliable result is determined from the algorithms. However, we can

conclude that DART_MCP performance was influenced by its ability to optimise the network requirements, address the control path issues, and as well resolved the possibility of co-channel network interferences within the network. These were depicted with the performance of the network in terms of lower running time, enhanced network success ratio as well as increased network stability performance.

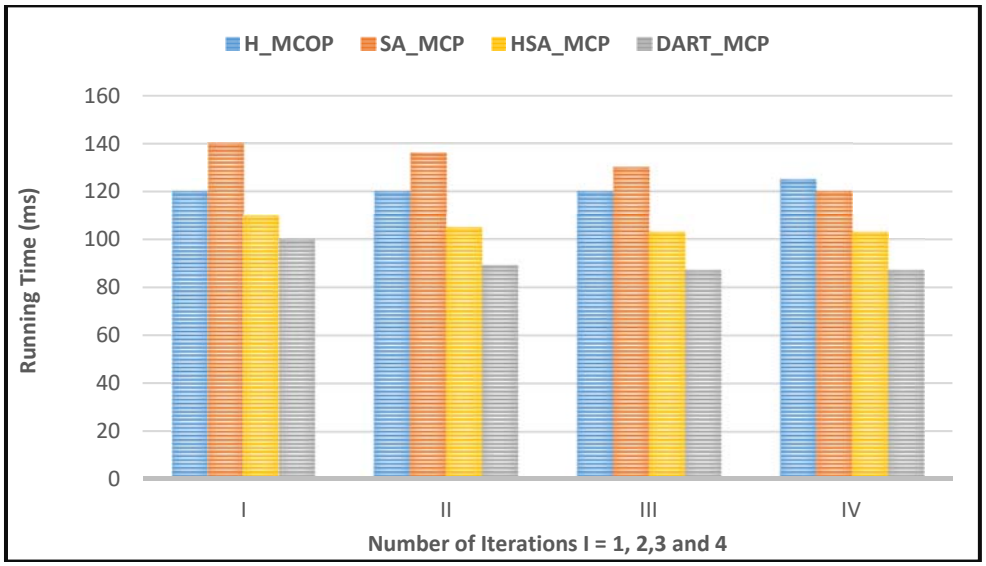

Fig.8: Output of the running time performance under 900 network nodes.

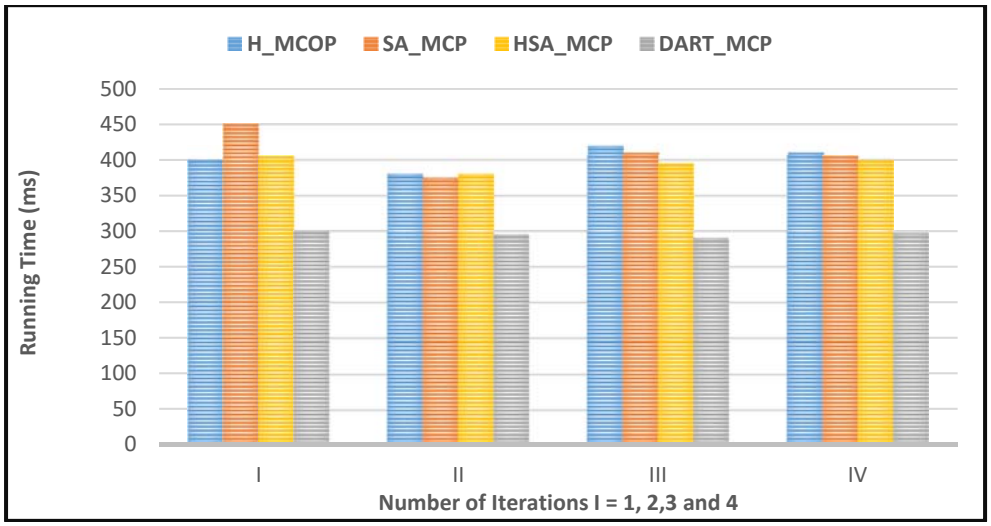

Fig.9: Output of the running time performance under 1500 network nodes.

The diagram in Figures 10 and 11 compares the performance of both HAS_MCP and DART_MCP algorithms especially in terms of the duration of delays before a stable state is reached. We selected only these two among others for this test because only HSA_MCP can be seen to perform at least in a comparable manner to our proposed algorithm. A total of 10 controllers and 1500 switches were used to investigate the impact of network interference on the stability that was attained in the course of packet routing. The delays were measured in milliseconds while the progressive results were derived in the course of sending packets. The three-dimensional representation was depicted in Figure 10 and Figure 11 for HSA_MCP and DART_MCP respectively.

The figures showed that for network nodes of 1500 running with 10 controllers, it took HAS_MCP approximately 245 ms to attain network stability while on the other hand, DART_MCP in Figure 11 only used 140 ms instead. The differences between these two values occurred as a result of network interferences as earlier highlighted. Under this experimental design, there are some cases in which an appropriate number of controllers would have enabled total avoidance of network interferences however, this was designed to see how much of the effects could be reduced by the deployed algorithms. Thus, a vast difference of 105 ms existed in the performance of the two algorithms when compared. It is noteworthy to understand that at each interval, the amount of delays could be

calculated to determine the corresponding values under similar conditions. However, to avoid unnecessary replication of graphs, we just selected the same number of network setups but running on different algorithms to compare the performances. The remaining results were depicted in Table II.

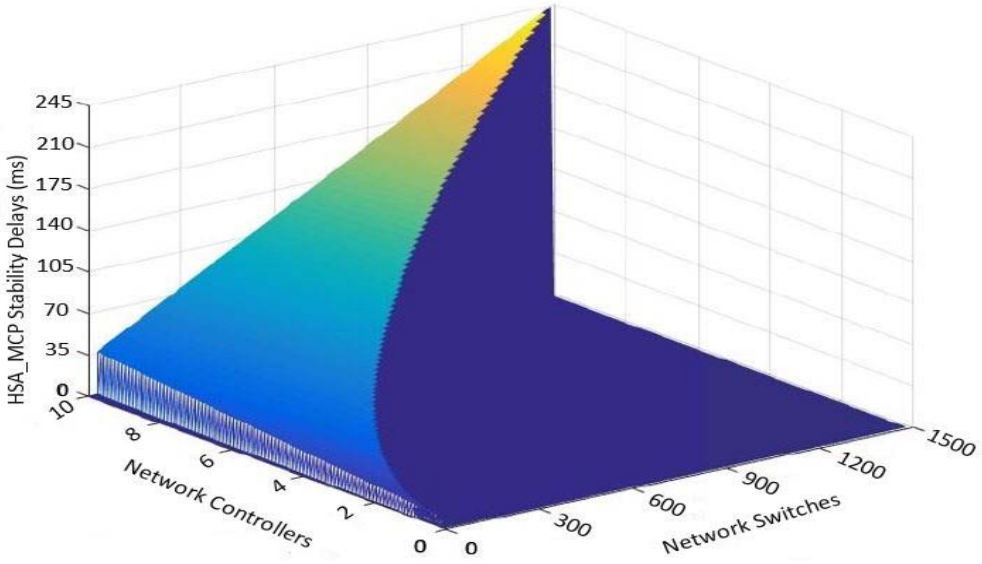

Fig.10: HAS_MCP Delay before attaining network stability.

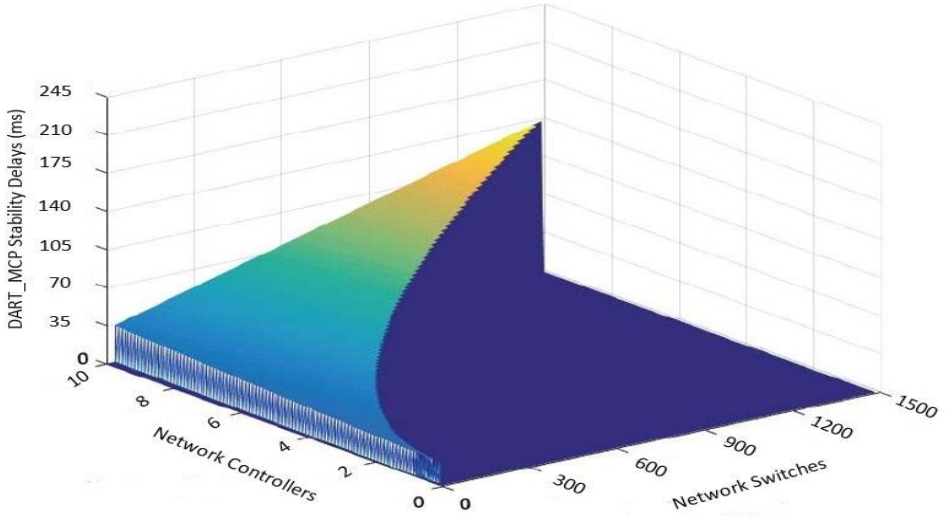

Fig.11: DART_MCP's Delay before attaining network stability.

SDN resources		Algorithm Delays			
Switches	***Controllers***	***H_MCOP***	***SA_MCP***	***HSA_MCP***	***DART_MCP***
300	2	128	88	47	26
600	4	255	176	96	55
900	6	388	267	145	83
1200	8	518	356	194	111
1500	10	650	450	245	140

Table 2. Comparison of Network Stability Delays

Hence, in a case where we are short of network resources especially controllers, and at the same time we are trying to accommodate more network hosts, a high stability routing protocol will be of importance which can help to reduce if not totally alleviate the effect of network interferences. Based on these four tested algorithms, DART_MCP provides us with the best network stability, causing a reduction in the network delays with the percentage of 42.9 as depicted by our experiments.

8. Conclusions and Future Works

This work was motivated by the need for an improved routing network in SDN that focuses on a stable network performance when larger network nodes were considered. We identified basically the impact of network interference in affecting the combination of user requirements which invariably reduces the success rate and the running time of the network. The success rate and running time directly speaks to the throughput and the latency experienced in the course of the network operations thus they are metrics that are needed to be optimised. We address these basic challenges by deploying a deterministic adaptive rendering technique (algorithm) called DART_MCP.

The DART_MCP algorithm contains the Monte-Carlo (ZZ) buffer which supposedly assists in establishing the stability of the network performance in relation to the effect of network interferences. The buffer approaches the versatility of distributed networks as it can render a wide influx of transferred packets while maintaining the QoS of varying network users. Hence, the proposed algorithm was able to optimize the network metrics requirements to a level that is optimal on the basics of this work in the domain of SDN. The evaluation of the algorithm against others showed that our proposed algorithm was able to maintain a lower network running time and higher success ratio to provoke low latency and higher throughput respectively. Further experiments also showed a faster convergence (with a reduction of 42.9% in-network delays) in attaining the network stability state for DART_MCP than the existing algorithms even though for the kind of design, interference could not be totally eradicated meaning that our proposed algorithms could assist in managing the network resources especially in a situation where the number of controllers available is fewer than what is needed.

We were also able to reduce the key inhibitors to network stability which is predominantly the control flow path issues and the co-channel interferences. The future work that we envisaged in this work is to introduce the deployment of machine learning algorithms or some AI mechanisms thereby introducing some level of intelligence to enhance the performance of the SDN system.

Acknowledgments

The authors acknowledge the funds received from the industry partners: Telkom SA Ltd, South Africa in support of this research.

References

[1] Akinola, A.T.; Adigun, M.O.; Mudali, P. (2020): Incorporating Stability Estimation into Quality of Service Routing in an SDN-Based Data Centre Network. In Proceedings of Fifth International Congress on Information and Communication Technology: ICICT, London, Volume 2 (pp. 406-416). Springer Singapore.

[2] Akinola A. T.; Adigun, M. O.; Mudali, P. (2018): Effects of Scalability measures on Control Plane Architectural Designs in different Data Centre Architectures, in *Southern African Telecommunication and Network Applications Conference*, Arabella, Hermanus, Western Cape, South Africa, pp. 198-203.

[3] Akinola, A. T.; Adigun, M. O.; Akingbesote, A. O.; Mba, I. N. (2015): Optimal Route Service Selection in Ad-Hoc Mobile E-Marketplaces with Dynamic Programming Algorithm using Tsp Approach, in International Conference on E-Learning Engineering and Computer Software, pp. 74-81.

[4] Apostolopoulos, G.; Guérin, R.; Kamat, S.; Tripathi, S. K. (1998): Quality of service based routing: A performance perspective, in SIGCOMM Computer Communication Review, 28(4), pp. 17-28: ACM.

[5] Chen, Y.; Mucchi, L.; Wang, R.; Huang, K. (2014): Modeling Network Interference in the Angular Domain: Interference Azimuth Spectrum, IEEE Trans. Communications, 62(6), pp. 2107-2120.Broder, A.; Kumar, R.; Maghoul, F.; Raghavan, P.; Rajagopalan, S.; Stata, R.; Tomkins, A.; Wiener, J. (2000): Graph structure in the Web. Computer Networks, **33**(1–6), pp. 309–320.

[6] Chen S.; Nahrstedt, K. (1999): Distributed quality-of-service routing in ad hoc networks, IEEE Journal on Selected areas in Communications, 17(8), pp. 1488-1505.

[7] Cheng, G.; Ansari, N.; Papavassiliou, S. (2008): Adaptive QoS provisioning by pricing incentive QoS routing for next generation networks, Computer Communications, 31(10), pp. 2308-2318.

[8] Egilmez, H. E. (2012): Adaptive video streaming over openflow networks with quality of service, Citeseer.

[9] Garroppo, R. G.; Giordano, S.; Tavanti, L. (2010): A survey on multi-constrained optimal path computation: Exact and approximate algorithms, Computer Networks, 54(17), pp. 3081-3107.

[10] He, M.; Basta, A.; Blenk, A.; Kellerer, W. (2017): Modeling flow setup time for controller placement in sdn: Evaluation for dynamic flows," in IEEE International Conference on Communications (ICC).

[11] Karakus M.; Durresi, A. (2016): A scalability metric for control planes in software defined networks (SDNs), in Advanced Information Networking and Applications (AINA), 2016 IEEE 30th International Conference on, pp. 282-289: IEEE.

[12] Korkmaz T.; Krunz, M. (2001): Multi-constrained optimal path selection, in IEEE INFOCOM, 10(2), April, pp. 834-843: Citeseer.

[13] Lafortune, E. (1996): Mathematical models and Monte Carlo algorithms for physically based rendering, Department of Computer Science, Faculty of Engineering, Katholieke Universiteit Leuven, pp. 74-79.

[14] Lafortune E. P.; Willems, Y. D. (1994): A theoretical framework for physically based rendering, in Computer Graphics Forum, 13(2), pp. 97-107: Wiley Online Library.

[15] Lee, W. C.; Hluchyi, M. G.; Humblet, P. A. (1995): Routing subject to quality of service constraints in integrated communication networks, IEEE Network, 9(4), pp. 46-55.

[16] Liu G.; Ramakrishnan, K. (2001): A* Prune: an algorithm for finding K shortest paths subject to multiple constraints, in Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies, 10(2), pp. 743-749: IEEE.

[17] Modeling Topology of Large internetworks, (2014): https://www.cc.gatech.edu/projects/gtitm/.

[18] Nguyen, V. D.; Begin, T.; Guérin-Lassous, I. (2013): Multi-constrained routing algorithm: A networking evaluation, in IEEE 37th Annual Computer Software and Applications Conference Workshops (COMPSACW), pp. 719-723.

[19] Sheng, L.; Song, Z.; Yang, J. (2015): A multi-constrained routing algorithm for software defined network based on nonlinear annealing," Journal of Networks, 10(6), pp. 376-385.
[20] Song, P.; Liu, Y.; Liu, T.; Qian, D. (2017): Controller-proxy: Scaling network management for large-scale SDN networks," Computer Communications, 108(2), pp. 52-63.
[21] Sridharan, V.; Gurusamy, M.; Truong-Huu, T. (2017): On multiple controller mapping in software defined networks with resilience constraints," IEEE Communications Letters, 21(8), pp. 1763-1766.
[22] Wang, Z.; Crowcroft, J. (1996): Quality-of-service routing for supporting multimedia applications, IEEE Journal on selected areas in communications, 14(7), pp. 1228-1234.
[23] Yeganeh, S. H.; Tootoonchian, A.; Ganjali, Y. (2013): On scalability of software-defined networking, IEEE Communications Magazine, 51(2), pp. 136-141.
[24] Zhang, B.; Hao, J.; Mouftah, H. T. (2014): Bidirectional Multi-Constrained Routing Algorithms, IEEE Trans. Computers, 63(9), pp. 2174-2186.
[25] Zhang, S. Q.; Zhang, Q.; Bannazadeh, H.; Leon-Garcia, A. (2015): Routing algorithms for network function virtualization enabled multicast topology on SDN, IEEE transactions on Network and Service Management, 12(4), pp. 580-594.
[26] Zhu, M.; Cao, J.; Pang, D.; He, Z.; M. Xu. (2015): SDN-based routing for efficient message propagation in VANET, in International Conference on Wireless Algorithms, Systems, and Applications, pp. 788-797: Springer.
[27] Zhu, W.; Song, M.; Olariu, S. (2006): Integrating Stability estimation into Quality of Service Routing in Mobile Ad-hoc Networks, in IEEE, pp. 122-129.

Authors Profile.

Ayotuyi Tosin AKINOLA, is currently a PhD student in the department of computer science at the University of Zululand, South Africa. He has submitted his Thesis and he is awaiting the examination. He has written several papers on web service selection in Ad-Hoc Mobile Cloud Computing. His area of specialization covers Mobile cloud computing, Ad-Hoc mobile cloud computing systems as well as Software Defined Networking deployment in Networking Environments.

Prof Matthew O. ADIGUN, is currently a Senior Professor of Computer Science at the University of Zululand, He obtained his doctorate degree in 1989 from Obafemi Awolowo University, Nigeria; having previously received both Masters in Computer Science (1984) and a Combined Honours degree in Computer Science and Economics (1979) from the same University (when it was known as University of Ife, Nigeria). A very active researcher in Software Engineering of the Wireless Internet, he has published widely in the specialised areas of reusability, software product lines, and the engineering of on-demand grid computing-based applications in Mobile Computing,

Dr. Pragasen Mudali, is a Senior Lecturer of the Department of Computer Science at the University of Zululand. He is also a member of the Institute of Electrical and Electronic Engineers (IEEE) and the South African Institute of Computer Scientists and Information Technologists (SAICSIT). His research interests revolve around energy-efficient networking for resource-constrained usage scenarios. He has focused on Wireless Mesh Networks in the past and is now applying the knowledge gained to the Internet-of-Things and Software-defined Networking. He holds a PhD in Computer Science from the University of Zululand that was awarded in 2017.