Synergy: Cross-Disciplinary Journal of Digital Investigation
(ISSN 2995-4827) VOLUME 02 ISSUE 9, 2024

PERFORMANCE EVALUATION OF AD-HOC NETWORKS USING NS2: A COMPARATIVE STUDY OF ROUTING PROTOCOLS

Pragya Devi
Department of Computer Science and Engineering, Parul Institute of Engineering and Technology, Parul University, Vadodara Gujarat, India

Abstract:

An Ad-hoc network is a type of wireless network where nodes communicate with one another using multi-hop links, without the reliance on any fixed infrastructure or centralized administration. In this dynamic network, wireless mobile nodes self-organize to create a communication framework that uses multi-hop point-to-point (P2P) routing to establish network connectivity. Given the limited range of wireless transmission, each node acts as a relay for other nodes, enabling communication between distant nodes by forming a cooperative network. The primary challenge in Ad-hoc networks lies in discovering and maintaining routes for data transmission between source and destination nodes, especially when they are separated by multiple intermediate nodes. This decentralized and infrastructure-less approach makes Ad-hoc networks highly flexible, but also presents unique challenges related to routing, scalability, and network stability.

Keywords: Ad-hoc Network, NS2, Wireless Networks, Routing Protocols, Dynamic Topology, Multi-Hop Communication, Network Simulation, Reactive Protocols.

1. Introduction

NS is an event driven network simulator developed at University of California at Berkeley, USA, as a REAL network simulator projects in 1989 and was developed at with cooperation of several organizations. Now, it is a VINT project maintained by DARPA. NS is not a complete tool that can handle all kinds of network model. It is really still an on-going endeavor of research and development. The users are dependable to authenticate that their network model simulation does not contain any bugs and the community should share their discovery with all [1-3]. There is a manual called NS manual for user guidance.

NS is a different event network simulator where the timing of events is maintained by a scheduler and able to simulate different types of network such as LAN and WPAN according to the programming scripts written by the user. Besides that, it also implements multiplicity of applications, and protocols such as TCP and UDP, and network elements such as signal power, traffic models such as FTP and CBR, the mechanisms of router queue management such as Drop Tail and many more [4-5].

There are two languages used in NS2 C++ and OTcl (an object oriented extension of Tcl). The compiled C++ programming hierarchy makes the simulation efficient and execution times faster. The OTcl script which is written by the users their own specific topology, protocols and all requirements need in the network models. The simulator is produce the output also can be set using OTcl. The OTcl script is written which creating an event scheduler objects and network component object with network setup helping modules. The simulation results produce after running the scripts can be use either for simulation analysis or as an input to graphical software called Network Animation (NAM) [6].

2. Literature Review

Ad-hoc networks have gained significant attention due to their decentralized nature and the absence of fixed infrastructure, which makes them highly adaptable in various applications such as military operations, disaster recovery, and mobile sensor networks. According to [1], Ad-hoc networks are composed of wireless mobile nodes that dynamically form a network without any central administrative control. This flexibility is advantageous in scenarios where traditional infrastructure cannot be established, but it also introduces several challenges, particularly in routing and maintaining network connectivity [7].

One of the core aspects of Ad-hoc networks is multi-hop point-to-point (P2P) communication, where data is transmitted between nodes through intermediate hops [2]. In traditional networks, communication typically relies on fixed infrastructure like routers or base stations, but in Ad-hoc networks, nodes must collaborate to forward data. As discussed by Johnson and Maltz [3], routing protocols play a crucial role in ensuring efficient communication in such networks. Early research introduced two main categories of routing protocols: proactive (table-driven) and reactive (on-demand). Proactive protocols, such as the Destination-Sequenced Distance-Vector (DSDV) routing, continuously maintain routing tables, while reactive protocols, like Ad-hoc On-Demand Distance Vector (AODV) and Dynamic Source Routing (DSR), discover routes only when necessary [4].

The dynamic topology and the limited transmission range of nodes in Ad-hoc networks create several routing challenges. The constant mobility of nodes leads to frequent topology changes, causing routes to break and requiring continuous updates. According to Perkins and Royer [5], maintaining robust and efficient routes in such a dynamic environment is essential for minimizing latency and ensuring data delivery. Additionally, factors like limited bandwidth, power consumption, and node mobility further complicate the design of effective routing protocols [8].

In summary, the literature highlights the potential of Ad-hoc networks for flexible and infrastructure-less communication, while also addressing the technical challenges that arise from dynamic topologies and the need for efficient routing mechanisms. Continued research on routing protocols and network optimization is essential to unlock the full potential of Ad-hoc networks in real-world applications [9-11].

3. Process of NS2

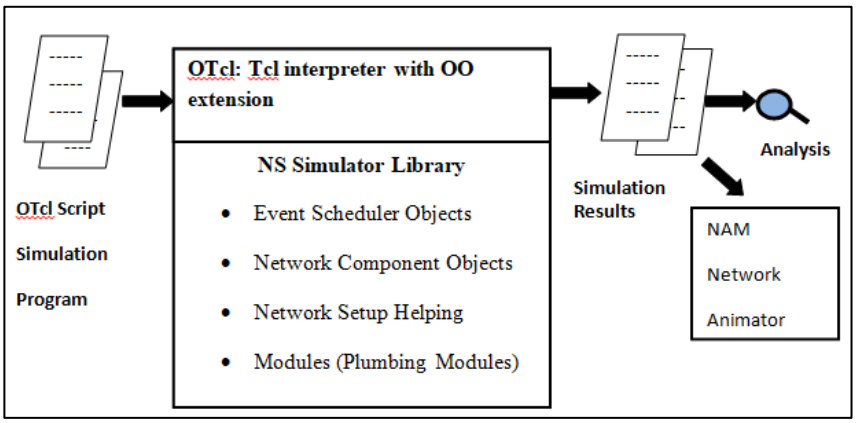

Fig. 1 Process of NS2

OTCL Helps in the Following Way:

- ✓ With the help of OTcl we can describe different network topologies
- ✓ It helps us to specify the protocols and their applications
- ✓ It allows fast development
- ✓ Tcl is compatible with many platforms and it is flexible for integration
- ✓ Tcl is very easy to use and it is available in free

Initialization

#To create a new simulator we write

```
set ns [new Simulator]
```

From the above command we get that a variable ns is being initialized by using the set command. Here the code [new Simulator] is a instantiation of the class Simulator which uses the reserved word new. So we can call all the methods present inside the class simulator by using the variable 'ns'.

- ➢ Creating The Output Files

#Create the trace files

```
set tracefile[open out.tr w]
$ns trace-all $tracefile
```

#Create the nam files

```
set namfile [open out.nam w]
$ns nametrace-all $namfile
```

In the above we create a output trace file 'out.tr' and a NAM visualization file 'out.nam'. But in the Tcl script they are not called by their names declared, while they are called by the pointers initialized for them such as 'tracefile' and 'namfile' respectively. The line which starts with # are commented. The next line opens the file 'out.tr' which is used for writing is declared 'w'. The next line uses a simulator method trace-all by which we will trace all the events in a particular format [12-13].

The termination program is done by using a 'finish' procedure

#defining the ‘finish procedure’

```
Proc finish {}
{
        Global ns tracfile namfile
        $ns flush-trace
        Close $tracefile
        Close $namfile
        Exit 0
}
```

In the above, the keyword proc is used to declare a procedure called 'finish'. The keyword global is used to tell what variables are being used outside the procedure.

Flush-trace is a simulator method that dumps the traces on the respective files. The command close is used to close the trace files and the command exec is used to execute the NAM visualization. The command exit closes the application and returns zero as default for clean exit.

In ns we end the program by calling the 'finish' procedure:

```
# End the program
        $ns at 125.0 “finish”
```

Thus the entire operation ends 125 seconds. To begin the simulation we will use the command

```
# Start the simulation process
        $ns run
```

So we are creating a bi-directional link between nodes n0 and n2 with a capacity of 10 Mb/sec and a propagation delay of 10 ms.

In NS an output queue of a node is implemented as a part of a link whose input is that node to handle the overflow at the queue. If the buffer capacity of the output queue is exceeded then the last packet arrived is dropped and here we will use a 'DropTail' option. There are other queue types such as RED (Random Early Discard) mechanism, FQ (Fair Queuing), DRR (Deficit Round Robin), SFQ (Stochastic Fair Queuing) also available [14-15].

4. User Datagram Protocol (UDP)

The User datagram Protocol is one of the main protocols of the Internet protocol suite. UDP helps the host to send send messages in the form of datagrams to another host which is present in a Internet protocol network without any kind of requirement for channel transmission setup. UDP provides a unreliable service and the datagrams may arrive out of order, appear duplicated, or go missing without notice. UDP assumes that error checking and correction is either not necessary or performed in the application, avoiding the overhead of such processing at the network interface level. Time-sensitive applications often use UDP because dropping packets is preferable to waiting for delayed packets, which may not be an option in a real-time system [16].

```
# Setup a UDP connection
set udp [new Agent/UDP]
$ns attach-agent $n1 $udp
$set null [new Agent/Null]
$ns attach-agent $n5 null
```

$ns connect $udp null

The command set udp [new Agent/UDP] gives a pointer called 'udp' which indicates the udp agent which is a object of ns. Then the command $ns attach-agent $n1 $udp defines the source node of UDP connection. Next the command set null [new Agent/Null] defines the destination of udp by a pointer called null. The next command $ns attach-agent $n5 $null defines the destination node as n5. Next, the command $ns connect $udp $null makes the UDP connection between the source and the destination i.e n1 and n5. To identify a particular flow we mark it using the command$udp set fid_2 [17].

5. Constant Bit Rate (CBR)

Constant Bit Rate (CBR) is a term used in telecommunications, relating to the quality of service. When referring to codecs, constant bit rate encoding means that the rate at which a codec's output data should be consumed is constant. CBR is useful for streaming multimedia content on limited capacity channels since it is the maximum bit rate that matters, not the average, so CBR would be used to take advantage of all of the capacity. CBR would not be the optimal choice for storage as it would not allocate enough data for complex sections (resulting in degraded quality) while wasting data on simple sections.We define a CBR connection over a UDP one. Well we have already defined the UDP source and UDP agent as same as TCP. Instead of defining the rate we define the time interval between the transmission of packets in the command $cbr set rate_ 0.01Mb. Next, with the help of the command $cbr set random_ false we can set random noise in cbr traffic. We can keep the noise by setting it to false or we can set the noise on by the command $cbr set random_ 1. We can set by packet size by using the command $cbr set packetSize_. The packet size is specified in bytes [15, 17, 18, 19].

6. Study analysis

The research paper presents a comprehensive overview of Ad-hoc networks, where mobile nodes form self-organizing, decentralized networks without relying on fixed infrastructure. The paper discusses the challenges of routing, dynamic topology, and scalability inherent in such networks, emphasizing the role of efficient routing protocols like DSDV, AODV, and DSR. To simulate and study the performance of these protocols, the paper introduces NS2, a network simulator that allows the evaluation of various network scenarios [18, 19, including Ad-hoc networks. Through the use of OTcl scripting and C++ programming, NS2 helps model network topologies, create traffic patterns [16-17], and analyze the behavior of wireless communication under different conditions. The outcomes of the research emphasize the importance of dynamic, reactive routing protocols in Ad-hoc networks and underscore the significance of simulation tools like NS2 in developing and refining these wireless systems.

7. Outcomes

1. **Routing Efficiency**: The study highlighted the performance of different routing protocols in Ad-hoc networks. It was observed that reactive protocols like AODV and DSR performed well under dynamic topology conditions due to their on-demand routing approach, minimizing overhead compared to proactive protocols like DSDV.
2. **Network Topology**: The flexibility of Ad-hoc networks allows for deployment in a wide range of applications, but the limited transmission range and mobility of nodes significantly impact the stability of routing paths. The NS2 simulations provided an understanding of how quickly routes break and the time it takes to establish new routes.
3. **Simulation Tools**: The study emphasized the utility of NS2 in simulating various types of network models, including LAN, WPAN, and Ad-hoc networks. By using C++ for efficient execution and OTcl for customizing network topologies and event scheduling, NS2 proved to be a versatile tool for studying wireless network behavior.

4. **Traffic Management**: Simulating traffic models such as Constant Bit Rate (CBR) and User Datagram Protocol (UDP) connections demonstrated how bandwidth constraints, packet sizes, and scheduling affect network performance. These simulations provide insights into optimizing traffic for real-time applications like video streaming or voice communication in Ad-hoc networks.

8. Conclusion

This research on Ad-hoc networks and their simulation using NS2 provides valuable insights into the design, implementation, and performance challenges of decentralized, infrastructure-less wireless networks. Ad-hoc networks, by their nature, offer flexible communication capabilities for environments where fixed infrastructure is unavailable or impractical. However, they also introduce unique issues such as dynamic topology, routing challenges, and scalability, which necessitate efficient routing protocols like DSDV, AODV, and DSR. Through simulations using the NS2 network simulator, various aspects of routing efficiency, bandwidth management, and packet loss can be evaluated, giving researchers tools to improve Ad-hoc network performance.

References

1. Reference to the description of Ad-hoc networks without a centralized administrator.
2. Point-to-point multi-hop communication in Ad-hoc networks.
3. Johnson, D. B., & Maltz, D. A. (1996). "Dynamic source routing in ad hoc wireless networks."
4. Perkins, C., & Bhagwat, P. (1994). "Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers."
5. Perkins, C. E., & Royer, E. M. (1999). "Ad-hoc On-Demand Distance Vector (AODV) Routing."
6. Pinotti CM, Betti Sorbelli F, Perazzo P, Dini G (2016) Localization with guaranteed bound on the position error using a drone. In: Proceedings of the 14th ACM international symposium on mobility management and wireless access, pp 147–154
7. Aslan S, Demirci S (2019) Solving UAV localization problem with artificial bee colony (ABC) algorithm. In: 2019 4th International conference on computer science and engineering (UBMK). IEEE, pp 735–738
8. C. Engineering, “Special Issue on Ubiquitous Computing Security Systems PERFORMANCE COMPARISON OF MULTIHOP WIRELESS MOBILE AH-HOC ROUTING PROTOCOLS,” vol. 4, no. 1, pp. 696–703.
9. Shreyaskumar Patel. (2021). Enhancing Image Quality in Wireless Transmission through Compression and De-noising Filters. In International Journal of Trend in Scientific Research and Development (Vol. 5, Number 3, pp. 1318–1323). IJTSRD. https://doi.org/10.5281/zenodo.11195294
10. Shreyaskumar Patel. (2022). Performance Analysis of Acoustic Echo Cancellation using Adaptive Filter Algorithms with Rician Fading Channel. In International Journal of Trend in Scientific Research and Development (Vol. 6, Number 2, pp. 1541–1547). IJTSRD. https://doi.org/10.5281/zenodo.11195267
11. M. de M. Lustosa and S. Singh, “Liowsn Project: An Operating System Remastered for Works with Simulation of Wireless Sensor Networks,” *International Journal of Computer Applications*, vol. 52, no. 12, pp. 32–37, 2012.
12. Prafulla Joshi, A. ., Jaiswal, R. C. ., More, P. ., Naik, S. ., & Dalal, R. (2024). Performance Evaluation and Comparative Analysis of AODV, DYMO, IARP, and IERP Routing Protocols in

Ad Hoc Networks. *International Journal of Intelligent Systems and Applications in Engineering*, *12*(18s), 354–363. Retrieved from https://ijisae.org/index.php/IJISAE/article/view/4980

13. Dhall, R., Dhongdi, S. Review of Protocol Stack Development of Flying Ad-hoc Networks for Disaster Monitoring Applications. *Arch Computat Methods Eng* 30, 37–68 (2023). https://doi.org/10.1007/s11831-022-09791-y
14. R. Yadav, P. Moghe, M. Patidar, V. Jain, M. Tembhurney and P. K. Patidar, "Performance Analysis of Side Lobe Reduction for Smart Antenna Systems Using Genetic Algorithms (GA)," *2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, Delhi, India, 2023, pp. 1-5, https://doi.org/10.1109/ICCCNT56998.2023.10306796
15. M. Patidar, G. Bhardwaj, A. Jain, B. Pant, D. Kumar Ray and S. Sharma, "An Empirical Study and Simulation Analysis of the MAC Layer Model Using the AWGN Channel on WiMAX Technology," *2022 2nd International Conference on Technological Advancements in Computational Sciences (ICTACS)*, Tashkent, Uzbekistan, 2022, pp. 658-662, https://doi.org/10.1109/ICTACS56270.2022.9988033
16. Shreyaskumar Patel. "Optimizing Wiring Harness Minimization through Integration of Internet of Vehicles (IOV) and Internet of Things (IoT) with ESP-32 Module: A Schematic Circuit Approach", International Journal of Science & Engineering Development Research (www.ijrti.org), ISSN:2455-2631, Vol.8, Issue 9, page no.95 - 103, September-2023, Available :http://www.ijrti.org/papers/IJRTI2309015.pdf
17. Fan Q, Ansari N (2018) Towards traffic load balancing in drone-assisted communications for IoT. IEEE Internet Things J 6(2):3633–3640
18. Yadav, R., Moghe, P. & et al. (2023). Performance Analysis of Side Lobe Reduction for Smart Antenna Systems Using Genetic Algorithms (GA). IEEEXplore 14th International Conference on Computing Communication and Networking Technologies (ICCCNT), Delhi, India, 1-5. https://doi.org/10.1109/ICCCNT56998.2023.10306796
19. Patidar, M. Bhardwaj, G. & et al. (2022). An Empirical Study and Simulation Analysis of the MAC Layer Model Using the AWGN Channel on WiMAX Technology, 2022 IEEEXplore 2nd International Conference on Technological Advancements in Computational Sciences (ICTACS), Tashkent, Uzbekistan, 658-662. https://doi.org/10.1109/ICTACS56270.2022.9988033