Journal of Computational Science 13 (2016) 26–36

Contents lists available at ScienceDirect

Journal of Computational Science

journal homepage: www.elsevier.com/locate/jocs

DEVS-M: A discrete event simulation framework for MANETs

Fatih Çelik

Department of Computer Engineering, Sakarya University, Esentepe Campus, Serdivan, Sakarya, Turkey

ARTICLE INFO

Article history:
Received 21 July 2015
Received in revised form 29 October 2015
Accepted 27 November 2015
Available online 18 December 2015

Keywords:
Distributed network simulator
MANET
DEVS
DEVS-M

ABSTRACT

The infrastructure required for mobile ad hoc networks (MANETs) should be continuously improved because of the increasing number of wireless devices in the world. Therefore, modelling and simulation is important for MANETs in order to detect problems that may arise when developing infrastructure solutions. Studies in literature show that simulation tools developed for cable networks have been transformed into MANET tools instead of developing MANET-specific simulation tools. In this paper, a simulation framework for MANETs is designed using the DEVS-Suite simulator tool, which is based on discrete event system specification (DEVS) formalism, and is used for the simulation of cable networks. A specific wireless node architecture for the nodes that form MANETs is developed for a simulation framework. In addition, the new framework includes a topography model to check the velocity and movements of the nodes, packet models roaming in MANET, and visualisation area to observe simulation events. A coupled model is generated by combining these two models. In addition, an ant colony-based load-balancing scheme is developed to test the model. As a result, the simulation tool developed for MANETs aims to keep memory consumption steady even when traffic and node intensity increases in simulators. The study also aims to undertake load-balancing tests to obtain results compatible with the values in literature.

1. Introduction

The MANET is a continuously self-configuring, small infrastructure network of mobile devices connected by wireless connections [1]. In recent years, MANETs have become widespread in all areas of life because devices, such as television, mobile phones, tablets, and computers are able to communicate with each other wirelessly. Therefore, the infrastructure required for MANETs should be constantly improved owing to the increasing number of nodes. Therefore, this development necessitated modelling and simulation tools to verify and validate the protocols that are developed for MANETs. The modelling and simulation tools developed for this purpose allow testing of protocols designed for MANETs.

Simulation tools used in the literature for MANET modelling and simulation were developed as MANET adaptations of simulation tools that were improved to model cable computer networks [2]. Therefore, while the links in these tools were adapted to antenna models, cable nodes were adapted to wireless nodes. One of the difficulties observed in wireless nodes is the placement of the nodes since wireless nodes are placed in networks randomly in contrast to cabled nodes. In order to overcome this problem, topology generators are adapted to fulfil wireless node placement requirements. Another difficulty faced in wireless nodes is the mobility of network nodes, whereas no mobility is possible in cable nodes. Hence, the simulator that will be developed requires the addition of a tool that does not exist in cable nodes. The developed tool will be able to check both location and velocity information for wireless nodes. Instantaneous change in the location of the mobile node during mobility should be identified by the simulator, depending on its velocity.

There are various simulators for MANETs in the literature and the most commonly used simulator tool is the ns-2 [3]. The ns-2 uses a pseudo random number generator (PRNG) as a topology generator. Some errors occur in the transmission range during node placement caused by the PRNG. This situation also shows the importance of a topology generator. Another commonly used simulator is OPNET [4]. OPNET is a simulator with commercial limitations and therefore generates a high accuracy expectation [2], which does not scientifically allow the formation of an experimental framework. There are also different simulator tools written by users themselves regarding specific experiments.

In this paper, a developed MANET simulation framework based on DEVS [5] is presented. This simulation framework can run distributed and parallel architectures as discrete events based on DEVS formalism [6]. This MANET simulation framework called 'DEVS-M' is developed using the DEVS-Suite network simulator [7].

The MANET simulation framework can serve parallel and distributed architectures. As opposed to other simulators, creation of scenarios and monitoring the results of these scenarios are easier

E-mail address: fatihc@sakarya.edu.tr

http://dx.doi.org/10.1016/j.jocs.2015.11.012
1877-7503/© 2015 Elsevier B.V. All rights reserved.

with the help of these developed tools. It is also possible to use it as a web-based MANET simulation framework in distance education since it is independent of platforms and is open source. It is different from other simulators since it does not change the memory consumption even when the load in simulators increases. AntODV [8] was inspired by an ant colony [9] load-balancing scheme based on the AODV [10] approach, which was developed to present the superior characteristics of the simulation framework.

Section 2 briefly reviews DEVS formalism. Section 3 explains the atomic node model, topography atomic model, packets, coupled model, topology generator developed for the MANET simulation framework, modelling processes, and architecture of visualisation area tools. Section 4 provides information on the simulation parameters necessary for the verification and validation of the developed MANET simulation framework. Section 5 graphically presents the results obtained in the experiments and provides performance assessments for them. Section 6 provides conclusions.

2. Discrete event modelling with DEVS

The modelling and simulation method used in this study was first presented for the modelling and analysis of discrete event systems by Dr. Bernard P. Zeigler in 1976 in his book titled *Theory of Modeling and Simulation*. The DEVS defines system behaviour in two different levels: atomic and coupled DEVS. At the lowest level, DEVS defines the autonomous behaviour of a discreet event system, such as the transitions between sequential cases, how the system responds to an external input (events), and how it evaluates the output (events) [5]. An atomic DEVS model is defined as follows.

$$M =< X, Y, S, ta, \delta_{\mathrm{int}}, \delta_{ext}, \lambda >$$

The set of input events is X, while Y is the set of output events, S is the set of sequential states and $ta : S \rightarrow T^{\infty}$ is the time advance function, which is used to determine the lifespan of a state. Moreover, $\delta_{ext} : Q \times X \rightarrow S$ is the external transition function, which defines how an input event changes the state of the system, where $Q = \{(s, t_e)|s \in S, t_e \in (T \cap [0, ta(s)])\}$ is the set of total states, and t_e is the elapsed time since the last event. In addition, $\delta_{int} : S \rightarrow S$ is the internal transition function, which defines how a state of the system changes internally (when the elapsed time reaches the lifetime of the state). Further, $\lambda : S \rightarrow Y^{\phi}$ is the output function where $Y^{\phi} = Y \cup \{\phi\}$ and $\phi \notin Y$ is a silent event or an unobserved event. This function defines how a state of the system generates an output event (when the elapsed time reaches the lifetime of the state) [5].

Coupled DEVS defines a system as a network of components at a higher level. The components may be atomic DEVS models or coupled DEVS models. Couplings show how components affect each other. Output events of a component may be the input events of another component. It is possible to design an atomic DEVS for each coupled DEVS, and atomic or coupled DEVS models can be displayed by an atomic DEVS [5]. A hierarchical modelling structure is supported since coupled DEVS can have other coupled DEVS components [11]. A coupled DEVS model is defined as follows.

$N =< X, Y, D, \{M_i\}, C_{xx}, C_{yx}, C_{yy}, Select >$

The set of input events is X, while Y is the set of output events, and D is the named set of sub-components. In addition, $\{M_i\}$ is the set of sub-components, where for each $i \in D$, M_i there can be either an atomic DEVS model or a coupled DEVS model. Moreover, $C_{xx} \subseteq X \times \bigcup_{i \in D} X_i$ is the set of external input couplings, and $C_{yx} \subseteq \bigcup_{i \in D} Y_i \times \bigcup_{i \in D} X_i$ is the set of internal couplings, while $C_{yy} : \bigcup_{i \in D} Y_i \rightarrow Y^{\phi}$ is the external output coupling function, and $Select$:$2^D \rightarrow D$ is the tie-breaking function, which defines how to select the event from the set of simultaneous events.

Advantages of DEVS formalism are strong connections between components, hierarchical design, event-based simulation, object-oriented adaptation, reduced design time, more developed tests, higher quality models, easier experimentation opportunities, autonomous opportunities for working in parallel/real time, ease of verification and validation, interoperability and reuse, modelling using more than one method, and high performance [12]. This study presents the performance achievements of the DEVS method, especially on parallel and distributed systems, such as networks, by utilising the advantages cited.

3. The components of the DEVS-M

Each node is designed in a way to process the packets in the formed topology and to route these packets to desired directions. Nodes are atomic models that can connect to and communicate with the nodes included in the coverage area in the system as in reality. Behavioural characteristics of the nodes include the bandwidth to process the traffic, processing velocity, and limited buffering size to process traffic. Network components of various capacities can be formed by changing the defined characteristics, and different network scenarios can be developed [11]. The wireless network model has two different types of atomic model: a network coupled model and experimental frame. The node atomic model and topography atomic model form coupled node models. Furthermore, the experimental frame model also has event generator and event transducer atomic models (see Fig. 1). Every wireless node in the wireless network model has an antenna model that allows traffic flow in a certain radius that is modelled as a wireless node with the ability to route them to suitable targets. The wireless nodes are atomic DEVS models connected to each other with wireless network connections.

3.1. Atomic node model

Wireless nodes are designed as atomic nodes, which can generate and route data packets and control packets. Fig. 2 presents the wireless node architecture in which the most important module is the routing layer. This module is composed of tables that store the shortest route for the data packet. Atomic nodes consist of energy, topography, processing time, queue, packaging, node IP, and routing. Fig. 3 displays the relationships between classes, which compose the architecture of an atomic node. The atomic DEVS model for a node is given as follows. Node=< X, Y, S, s_0, ta, δ_{ext}, δ_{int}, λ > such that

$$X = \{?IIZC\_receive, DATA\_receive, DATA\_received\}$$

$$Y = \{!IIZC\_send, GIZC\_send, DATA\_forward, DATA\_send, DATA\_updated\}$$

$$S = \{(d, \sigma)|d \in \{wait, IIZC\_send, GIZC\_send, DATA\_forward, DATA\_send, DATA\_updated\}, \sigma \in T^{\infty}\}$$

$$s_0 = (DATA\_send, 0.01)$$

$$t_a(s) = \sigma \text{ forall } s \in S$$

$$\delta_{ext}(((wait, \sigma), t_e), ?DATA\_receive) = (DATA\_forward, 0.01)$$

$$\delta_{\mathrm{int}}(DATA\_forward, \sigma) = (wait, \infty)$$

$$\delta_{\mathrm{int}}(wait, \sigma) = (DATA\_forward, 0.01)$$

$$\lambda(DATA\_forward, \sigma) = !DATA\_forward$$

$$\lambda(wait, \sigma) = \phi$$