BAMSim Simulator

Rafael F. Reale[2], Walter P. Neto[1], and Joberto S. B. Martins[1]

[1] Salvador University (UNIFACS), wcpneto@gmail.com, joberto.martins@unifacs.br
[2] Federal Institute of Bahia (IFBA), reale@ifba.edu.br

Abstract

Resource allocation is an essential design aspect for current systems and bandwidth allocation is an essential design aspect in multi-protocol label switched and OpenFlow/SDN network infrastructures. The bandwidth allocation models (BAMs) are an alternative to allocate and share bandwidth among network users. BAMs have an extensive number of parameters that need to be defined and tuned to achieve an expected network performance. This paper presents the BAMSim simulator to support the network manager decision process in choosing a set of BAM configuration parameters for network design or during network operation.

1 Introduction

The allocation of resources is an essential aspect in designing existing systems like the internet of things (IoT), smart cities, smart grid, and new generation 5G and 6G systems [5]. Bandwidth is a resource allocated for network substrates that use links between switching nodes to support user communications. Bandwidth allocation is an essential aspect of design in broadly used network infrastructures like multi-protocol label switching (MPLS) network and network infrastructure deployment approaches using OpenFlow/SDN to control level-2 switches or any other switching equipment [4].

The bandwidth allocation models (BAMs) are an alternative to allocate and share bandwidth among network users grouped in traffic classes representing their communication requirements [6]. BAMs are mostly used in networks with limited bandwidth in which there is no over-provisioning for the communication links [11].

This paper presents the BAMSim bandwidth allocation model (BAM) simulator to support the network manager decision process in choosing an appropriate set of BAM configuration parameters during network design and operation.

2 BAMSim Simulator

The BAMSim is specifically oriented to support the allocation of bandwidth on MPLS networks according to the DS-TE (DiffServ Aware - Traffic Engineering) style proposed by Faucher et al. [2]. In summary, the DS-TE style means that bandwidth is allocated by traffic classes (TC), with each TC having a bandwidth constraint (BC).

The BAMSim processes every new LSP demand for the network. The BAM model decides if the LSP is created (allocating bandwidth) or rejected. LSP's requests can be granted, blocked, or preempted depending on the traffic class bandwidth available. Abstractly, an LSP has its starting time, duration, TC, and required bandwidth. The execution module is optionally available to interface the BAMSim simulator with a physical or an emulated network like the MiniNet with OpenFlow-based network control.

BAMSim facilities include: i) The support of MPLS signaling protocol features like LSP establishment, LSP teardown, LSP block and LSP preemption; ii) Network topology definition; iii) Path computation with static matrix and Constrained Shortest Path First (CSPF) route computation; and iv) Input traffic generation (with poisson, exponential and uniform distribution and deterministic traffic).

BAMSim supports the basic BAM models Maximum Allocation Model (MAM) [2], Russian Dolls Model (RDM) [3], and AllocTC-Sharing (ATCS) [10] and the generalized BAM model (GBAM) [8]. The advantages of BAM dynamic model switching and reconfiguration are discussed in [11] [9].

The BAMSim has a modular internal structure implemented in Java. The BAMSim simulator is available at `https://github.com/rfreale/BAMSim`.

BAMSim related work includes the simulation of BAM models based on the NS (Network Simulator) presented in Adami et al. [1]. The NS module developed simulates MAM and RDM BAM models. Compared with Adami's NS-based module, the BAMSim simulator extends the simulation of BAM modules to all existing models. To the limit of our knowledge, no other BAM simulator as developed since then.

3 BAMSim Architecture and Operation

The BAMSim architecture and main modules are illustrated in Figure 1.

BAMSIM architecture is modular, configurable, and flexible. The simulator code allows the inclusion of functionalities by integrating new modules. The BAMSim allows the configuration of various simulation parameters like simulation time, the simulation runs, number of generated LSPs, LSP duration time, simulation stop conditions, and pseudo-aleatory seeds for traffic generation. BAMSim flexibility includes its capability to import topologies like NSFNet, NTT (Nippon Telegraph and Telephone Corporation), define file-based customized topologies and define routing in the MPLS network with static matrix or protocol computed routing based in protocols like the CSPF.

The MPLS modules abstractly implement the main functions of an MPLS network such as establishment, preemption, devolution, and blocking of LSPs by traffic class (TC) through its interaction with the BAM, topology, path and routing modules in a configurable way.

The BAM module implements the BAM models. A GBAM-based implementation was used in the BAMSim to allow the implementation of all existing BAM models with its operational characteristics and behavior. Another important aspect of using GBAM is that it allows the on-the-fly switching of BAM models to adequate input traffic dynamism, as discussed in [11].

The cognitive module supports the use of machine learning techniques in assisting BAM parameters configuration and BAM model switching according to network traffic and state conditions and according to the objectives defined by the network manager. In the current BAMSim implementation, a Case-based Reasoning (CBR) module is implemented using the jColibri framework [7].

The topology module represents the MPLS network routers and links in an abstract way. The module represents routers interconnection as well as link bandwidth and active traffic classes per link. Topologies can be configured manually or imported by text files.

The routing module enables path selection by integrating a path selection algorithm. The BAMSim can use the CSPF or manually configured path selection algorithms. The routing module is developed in rJava, which allows us to explore the R language resources.

The status and statistics module monitors the states of the network elements and generates statistics and graphics with a Round Robin Database (RRD) using the jRobin library. Statistics

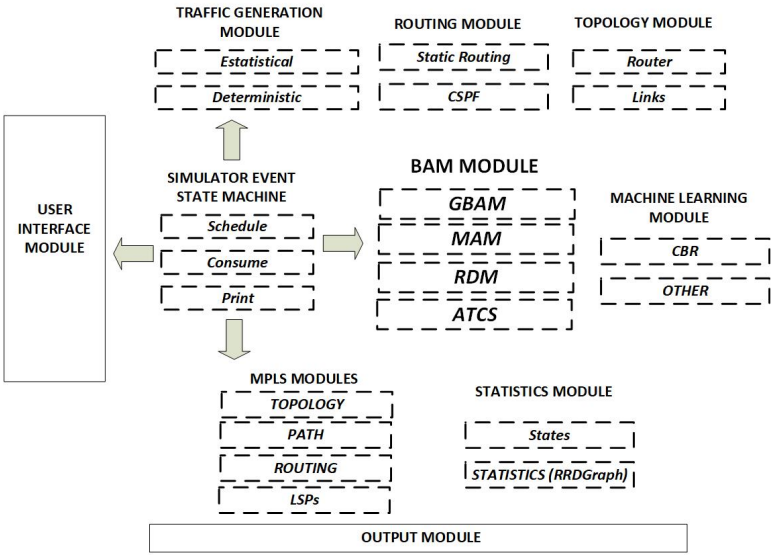

Figure 1: BAMSim Architecture and Internal Modules

are recorded in RRD files since RRD data and charts are intuitive for network managers and used in traditional network monitoring tools.

The traffic generator module allows BAMSim to generate random traffic profiles through probability functions such as uniform, Poisson, exponential, logarithmic, and deterministic traffic for DEBUG and traffic imported from real networks.

4 BAMSim Simulator Application Scenarios

Using BAMSim for BAM-based network design tuning means to evaluate how the configured input traffic impacts the MPLS network performance parameters like LSP preemption, LSP blocking, and link utilization for distinct BAM models. Table 1 illustrates a simulation scenario.

Table 1: Input Traffic - Simulation Pattern Example

Traffic Profile	**1**	**2**	**3**	**4**	**5**	**6**
TC0	High	Medium	Low	High		
TC1	Low	Low	Medium	High		
TC2	Low	High	High	High		
Link utilization	< 90%			>= 90%		

The simulation defines six input traffic profiles with 1 hour each. The first three traffic profile combinations alternate the need for sharing bandwidth between TCs and allows to assess what is the best BAM model option (MAM, RDM or ATCS) for each traffic profile. The last three input traffic profiles force link overload and allow to verify network performance parameters (blocking, preemption, other) and link utilization conditions for distinct BAM models and tune the one that best suits the network manager objectives.

Another BAM-based network-tuning possibility is to reconfigure the traffic class (TC) band-

width constraints (BC) and, by simulation, to evaluate how the network performance parameters behave for different BAM models.

BAMSim configuration is a straightforward task executed by command line (text file scripts) and GUI-based instructions to the simulator. For example, consider the simulation for a point-to-point MPLS topology with two routers connected by a network link. BAMSim configuration steps include the definition of the network topology (file scripts), input traffic generation characteristics, routing matrix, simulation stop criteria, BAM model and BAM configuration parameters. Script command syntax and semantics are obviously specific to the BAMSim. For example, $PTP-2n-1e$ is the topology name with two nodes and one link between routers 0 and 1. Additional syntax and semantics information is available with the BAMSim code.

The BAMSim simulator's utilization to facilitate the learning curve of the BAM configuration process is another relevant application scenario. The BAM teaching simulator's focus is now to allow the modification of the BAM configuration parameters and visualize their effect on the network performance. An application program, based on the BAMSim, was developed with this purpose and is available at https://github.com/rfreale/BAMSim/tree/Education.

5 Final Considerations

The BAMSim, in general, aims to facilitate the design process of bandwidth allocation in network infrastructures and foster the BAM-based bandwidth allocation learning curve. The BAMSim simulator facilitates the manager decision process in choosing the best BAM model and its operating parameters.

References

[1] D. Adami, C. Callegari, S. Giordano, and M. Pagano. A New NS2 Simulation Module for Bandwidth Constraints Models in DS-TE Networks. In *IEEE ICC*, pages 251–255, May 2008.

[2] F L Faucher and W Lai. Maximum Allocation Bandwidth Constraints Model for DiffServ-aware MPLS Traffic Engineering. Technical Report RFC 4125, June 2005.

[3] Le Faucheur, J. Boyle, T. Nadeau, and D. Skalecki. Russian Dolls Bandwidth Constraints Model for Diffserv-aware MPLS Traffic Engineering. Technical Report RFC 4127, 2005.

[4] Y. Li, X. Su, A. Ding, A. Lindgren, X. Liu, C. Prehofer, and P. Hui. Enhancing the Internet of Things with Knowledge-Driven Software-Defined Networking Technology. *Sensors*, 20(12), 2020.

[5] Jianhui Liu and Qi Zhang. Computation Resource Allocation for Heterogeneous Time-Critical IoT Services in MEC. *IEEE Wireless Communications and Networking Conference*, February 2020.

[6] Joberto Martins, Romildo Bezerra, Rafael Reale, and Gilvan Durães. Uma Visão Tutorial dos Modelos de Alocação de Banda como Mecanismo de Provisionamento de Recursos em Redes IP/MPLS. *Revista de Sistemas e Computação*, 5(2):144–155, December 2015.

[7] E. Oliveira, R. Reale, and J. Martins. Cognitive Management of Bandwidth Allocation Models with Case-Based Reasoning - Evidences Towards Dynamic BAM Reconfiguration. In *Proceedings of the IEEE Symposium on Computers and Communications*, pages 397–403, June 2018.

[8] Rafael Reale, Romildo Bezerra, and Joberto Martins. G-BAM: A Generalized Bandwidth Allocation Model for IP/MPLS/DS-TE Networks. *International Journal of Computer Information Systems and Industrial Management Applications*, 6:635–643, 2014.

[9] Rafael Reale, Romildo Bezerra, and Joberto S. B. Martins. Applying Autonomy with Bandwidth Allocation Models. *International Journal of Communication Systems*, 29(13):2028–2040, 2016.

[10] Rafael Reale, Walter Neto, and Joberto Martins. AllocTC-sharing: A New Bandwidth Allocation Model for DS-TE Networks. In *Proc. of the 7th Latin American Network Operations and Management Symposium*, pages 1–4, October 2011.

[11] Rafael Freitas Reale, Romildo Martins Bezerra, and Joberto S. B. Martins. Analysis of Bandwidth Allocation Models Reconfiguration Impacts. In *Proceedings of the III International Workshop on ICT Infrastructures and Services (ADVANCE)*, pages 67–76, 2014.