Statistical framework for ns-3: terminating simulation and regression analysis∗

Aleksandar Milenkoski†, Biljana Stojcevska†, Oliver Popov‡

1 SCSIT, University American College, Treta makedonska brigada, bb 1000 Skopje, Macedonia
2 DSV, SU and KTH Forum 100, Isafjordsgatan 39, SE 16440 Kista, Sweden

(Received September 3 2012, Accepted January 28 2014)

Abstract. The ns-3 simulator is constantly gaining popularity. It plays a central role in many recent research experiments. The accuracy of the reported simulation results of these experiments is an important concern. Thus, the support in ns-3 for simulation methodologies which guarantee the accuracy of these results is a necessity. Also, the simulation results are affected by numerous scenario parameters. The correlation between the results and the simulation parameters is a significant point of interest in many experiments. In this paper, we present a ns-3 statistical framework. It enables calculation of statistically accurate simulation results by applying the terminating simulation methodology. It features simultaneous execution of simulation scenarios in multi-processor and distributed environments. Also, we integrate support for regression analysis procedures. The proposed framework supports linear and polynomial regression analysis models. We consider simulation results as dependent variables and simulation parameters as independent variables. Regression analysis enables identification of simulation scenario parameters which are significantly correlated to given simulation results. Once a valid regression model is found, it is possible to estimatively predict metric values based on simulation parameter values. This may result in substantially reduced time and effort spent on simulation experimentation.

Keywords: simulation tool, simulation methodology, terminating simulation, regression analysis

1 Introduction

The constant change of the network protocols’ landscape is fueled by numerous research contributions. Their results are usually produced by various simulation tools[3–5]. The accuracy and the validity of these results are of great importance. Inaccurate simulation results may give a false impression of the performance characteristics of the network protocols. The used simulation methodologies and the validity of the reported simulation results have been frequently criticized in the past[13, 17]. This issue should be avoided in the current and future network research undertakings. Most of them use the popular network simulation tool ns-3[5]. Thus, we believe that development of a framework that enforces statistically sound evaluation for ns-3 is a priority.

The simulation scenarios contain numerous simulation scenario parameters. Some of them have significant impact on the simulation results of interest. The identification of these parameters could lead to focused and time-saving evaluation efforts. Also, an analytical relationship between simulation results and simulation scenario parameters might be used to estimate the results based on the simulation parameters values without spending time for actual simulation. Regression analysis is a method which can model and validate such a relationship. Regression analysis models may help to observe the statistical significance of various simulation parameters to a simulation output. They may also provide a formal abstraction of the relationship between simulation parameters as independent variables and simulation output metrics as dependent variables. That would

∗ Extensive gratitude to the reviewers and to Michal Papež for the thoughtful comments.
† Corresponding author. E-mail address: milenkoski@uacs.edu.mk

Published by World Academic Press, World Academic Union
make it possible to calculate metric values based on simulation parameter values. We suggest that analytical calculation of metrics of interest through regression models could save significant time spent on simulation. It might also provide basis for further analysis and high-level decision making. Previous works have already acknowledged the benefits of the use of regression analysis procedures in stochastic simulation\textsuperscript{[11, 20]}. However, to the best of our knowledge, no current ns-3 frameworks provide support for regression analysis.

In this paper, we present a framework which enables the terminating simulation methodology in the ns-3 network simulator. The framework executes numerous independent replications of a single simulation scenario. For each replication, it calculates the values of the metrics of interest. After certain number of replications, the means of these values are considered as statistically accurate metric values. We acknowledge that the repetitive execution of a simulation scenario could be a time-consuming process. Therefore, we provide simultaneous execution of the simulation replications. The framework offers simultaneous execution on a multi-processor (multi-core) machine, or on a physically distributed set of machines. Further, we implement support for linear and polynomial regression analysis.

The major design goals of the proposed framework are instant deployment and usability. The ns-3 simulator is consisting of a Python and a C++ codebase. The first is used for simulation execution, parametrization and control. The latter is used for implementation of the actual simulation scenarios. Our goal is to achieve minimal user involvement in the pre-execution set-up of the framework. To this end, the proposed framework resides completely in the Python codebase of ns-3. It provides simulation execution options and it does not require time-consuming modifications to the C++ code. In that way, the framework is blended in the usual ns-3 operational environment. Thus, it enables user-friendly use and support for generic ns-3 simulation scripts.

This paper is organized as follows: in Section 2, we give an overview of the related work. In Section 3, we briefly present the background theory, the implemented procedure of the terminating simulation method, and the simultaneous execution feature. In Section 4, we present on the applied regression analysis procedure. In Section 5, we consider an use-case scenario in which we demonstrate the use of regression analysis in stochastic simulation. Finally, in Section 6, we give our conclusions.

2 Related work

There are many statistical frameworks for the network simulators ns-3 and ns-2. Because of the many similarities in their designs and goals, we briefly discuss several such frameworks for the both network simulators.

An important work towards statistically accurate simulation is\textsuperscript{[16]}. Pawlikowski\textsuperscript{[16]} uses terminating simulation as a method to achieve statistically accurate results. Additionally, Pawlikowski\textsuperscript{[16]} uses parallelization to shorten the overall time spent on simulation. He distinguishes between the MRIP (Multiple Replications in Parallel) and the SRIP (Single Replication in Parallel) approach. The first approach assumes simultaneous execution of the whole simulation scenarios. The latter assumes partitioning each simulation scenario in sections which are executed in parallel manner. Pawlikowski\textsuperscript{[16]} applied the MRIP approach in his work because of several advantages. Some of them are effortless automatization and support of generic simulation scripts. Other noteworthy ns-2 statistical frameworks also exist. Such are ns2measure\textsuperscript{[7]} and simd\textsuperscript{[1]}. Their common goal is provisioning of data monitoring and simulation execution control abilities to the ns-2 users.

Amongst the most prominent statistical frameworks for ns-3 are the Statistical Framework for Network Simulation\textsuperscript{[3]} and the SAFE (Simulation Automation Framework for Experiments) framework\textsuperscript{[6]}. The Statistical Framework for Network Simulation integrates efficient data monitoring abilities in ns-3. It may provision data to tools for further statistical analysis and validation. One of the major advantages of this framework is the time-efficient data collection technique. It records simulation data on-line, i.e., without parsing simulation trace files. Consequently, the ns-3 tracing feature can be disabled. That could significantly speed up the simulation process due to reduced file input/output operations. The on-line data collection is achieved by modifying the C++ simulation scenario code in order to deploy data monitors. The framework also offers simulation execution control through a shell script. The shell script automates repetitive execution of a simulation scenario. At the same time, it may also vary user-specified simulation scenario parameters. The end result is an overview of the relationship between the collected simulation data and specific simulation parameters.
The SAFE framework is an ongoing project at the time of writing. Its goal is to provide an environment for statistically rigorous simulation experimentation. It is envisioned as a composition of four main components: data collection and processing framework, experiment execution manager, XML-like experiment descriptive languages, and steady-state/termination detectors. SAFE adopts the MRIP approach as a method to speed up replication execution. The experiment execution manager server takes as input a XML-based experiment description file. Then, it delegates simulation tasks to the experiment execution manager clients. Each such a client executes a simulation replication and collects the simulation data of interest. Then, it sends the data to the experiment execution server for further processing and storage. This process is iteratively repeated until a predefined confidence level is met. The functionalities of the framework are easily accessible through a web-based management system. That increases the value of the framework in terms of user-friendliness and overall usability.

3 Terminating simulation

When it comes to the use of the terminating simulation feature of our framework, the user specifies: a simulation scenario, a path to the simulation output trace file, a metric calculation approach for each metric of interest and statistical accuracy properties (i.e., confidence level and relative precision). The metric calculation approach specifies the way in which a specific metric is calculated from a trace file. It can take the form of a single text parsing command, e.g. the UNIX grep command. Alternatively, it can be a third-party script which performs complex metric calculations. Therefore, we provide effortless integration of various existing trace file processing scripts in the framework.

The statistical accuracy of each metric of interest is calculated according to a common terminating simulation procedure\cite{9,14}. After each replication, the framework estimates the mean $\bar{Y}_p$, denoted as $\hat{\bar{Y}}_p$. To this end, it calculates the average of the observations $y_1, y_2, \ldots, y_p$, where $y_q, \forall q \in [1, p]$, is the $q$-th value of the evaluation metric of interest, and $p$ is the number of the executed simulation replications.

Then the framework calculates the half-width of the desired confidence interval (1):

$$
\delta = t_{p-1,1-\frac{\alpha}{2}} \hat{\sigma}[\hat{\bar{Y}}_p],
$$

where $\alpha$ is the desired confidence level and $p-1$ is the degree of freedom of the Student’s $t$-distribution. $\hat{\sigma}[\hat{\bar{Y}}_p]$ is the estimation of the standard deviation of $\hat{\bar{Y}}_p$ (2):

$$
\hat{\sigma}^2[\hat{\bar{Y}}_p] = \frac{1}{p(p-1)} \sum_{q=1}^{p} (y_q - \hat{\bar{Y}}_p)^2.
$$

(2) holds because the observed samples are realizations of independent and normally distributed random variables. The independency of the random variables is guaranteed from the fact that $y_1, y_2, \ldots, y_p$ are obtained from independent replications. Each replication uses different statistically independent random numbers. To guarantee that there is no correlation between these random numbers, we use the ns-3 built-in MRG32k3a pseudo random number generator\cite{15}. We use an identical random seed for all replications, but different sub-streams of the randomly generated numbers. According to\cite{15}, each sub-stream has a period of $7.6 \times 10^2$. Thus, it provides the opportunity for execution of $2.3 \times 10^{15}$ independent simulation replications. Finally, the framework calculates the achieved relative precision\cite{9,16}:

$$
\epsilon = \frac{\delta}{\hat{\bar{Y}}_p}.
$$

If the desired relative precision is not yet reached, the framework simulates one more replication. To the contrary, the simulation is stopped and the value of $\hat{\bar{Y}}_p$ is considered as a statistically accurate value of the metric of interest.

WJMS email for contribution: submit@wjms.org.uk
3.1 Simultaneous execution

The number of replications needed to calculate statistically accurate results may be significantly big\textsuperscript{[16]}. We argue that a simultaneous execution of the required replications may reduce the overall simulation time. This approach is known as MRIP (Multiple Replications in Parallel). A possible alternative approach is parallelizing each replication itself, also known as SRIP (Single Replication in Parallel). We adopt the MRIP approach for the following reasons:

(1) The SRIP approach requires modification of the C++ code of the simulation scenario which is to be parallelized. Moreover, it requires decent knowledge in parallel programming. That goes against our basic design decisions of introducing modifications only in the Python ns-3 interface and relieving the user of framework set-up tasks;

(2) The SRIP approach is of great benefit when one executes complex and memory-consuming simulation tasks. However, effectively partitioning a complex simulation scenario may be an unfeasable or an error-prone task\textsuperscript{[16]}. Also, the SRIP approach is a strongly scenario-dependant approach. It requires knowledge of the implementational details of the simulation scenario to be parallelized. That characteristic does not go along with our design goal of ensuring support for generic ns-3 simulation scenarios.

Note that the framework can simultaneously execute simulation tasks in multi-processor (multi-core) and distributed computing environments. These functionalities are provided by the python-pp library\textsuperscript{[8]}.

As inter-process communication approach we adopt the traditional master-slave principle: the master simulation process assigns one replication to each worker process, i.e., slave. After each slave completes the assigned replication, it obtains the metric value $y_q$, $q \in [1, p]$ (Section 3) and sends it to the master process. If the master process does not receive a metric value from a slave after a certain time period, it considers the slave as unavailable. Consequently, it delegates the replication task to another slave. Finally, when it gathers the metric values from all slaves, the master process calculates the statistical accuracy of the simulation results. If the calculated relative precision does not satisfy the user’s requirements, the same procedure is repeated until the predefined criterion is met.

The MRIP approach is known to provide an average execution speed up according to the Truncated Amdahl Law\textsuperscript{[16, 18]}. This statement is valid under the assumption that the used machines have homogenous processors and that the available Input/Output (I/O) and memory bandwidth scale according to the number of simultaneous tasks. For terminating simulations, the Truncated Amdahl Law states that up to a certain limit, the average speed up equals the number of used machines.

4 Regression analysis

Regression analysis is a well-known technique for modeling the correlation between one dependent and one or more independent variables. The regression analysis method enables one to observe how the value of a dependent variable changes as values of different independent variables are varied. A regression model also makes it possible to calculate, i.e., to predict, dependent variable values for different values of independent variables. We incorporate regression analysis in the proposed framework with these goals in mind.

We assume the following:

(1) The simulation scenario parameters are realized as variables in the simulation scenario codespace - their values can be set by the simulator user before the simulation takes place. Therefore, we adopt scenario parameters as independent variables in the regression analysis model;

(2) The values of given metrics of interest are dependent on values of simulation parameters. Thus, we consider metrics as dependant variables. The previously mentioned dependence is our central concern in this work.

In Fig. 1, we depict the implemented regression analysis procedure in the proposed framework. It is consisting of two phases: (i) a preparation phase, and (ii) an analysis phase.
4.1 Preparation phase

The preparation phase is about defining the regression model, the dependent and the independent variables. It is consisting of the following steps:

Step 1 - Definition of a dependent variable: In this step, the user defines a dependent variable, i.e., a metric of interest. For that purpose, she should define a name of the metric, and a metric calculation approach.

Step 2 - Definition of independent variables: At this point, the user specifies independent variables, i.e., simulation parameters. Furthermore, she also defines a lower and an upper bound of the value range for each independent variable.

Step 3 - Definition of a regression model: In this step, the user defines a regression model. More on regression models is presented in Section 4.2.

Step 4 - Building the regression dataset: The framework needs a regression dataset to perform regression analysis. A regression dataset is a set of regression observations. We consider a single regression observation as a set of simulation parameter values (independent variables) and a metric value (dependent variable). The process of building the dataset is completely automated. First the framework randomly generates multiple values for each independent variable. The independent variables and the respective value ranges are defined in Step 2 of the preparation phase. For each independent variable, the user defines a number of generated values. A user may also define a number of generated values in specific sub-intervals of each value range. In that way, the framework generates values with smaller steps in those sub-intervals. That is useful when there is a higher variation rate of the dependent variable in those regions \[20\]. Then, the framework combines values
from the different value sets. Thus, it creates unique sets of independent variable values which we denote as observation points. The framework inserts the values from each observation point in a simulation scenario script.

For each observation point, the framework performs the terminating simulation procedure (Section 3). It iteratively simulates a simulation scenario and calculates the metric of interest from the simulation output files. For each replication, it records the simulation parameter values and the metric value as one regression observation. The framework replicates the simulation scenario until it calculates an appropriate value of ϵ (Section 3). After the framework records all regression observations, it considers the regression dataset as completed.

4.2 Analysis phase

The analysis phase takes place after the preparation phase. The goal of the analysis phase is to identify a relation between a dependent variable and independent variables, and to validate this relation.

Let $m_i$ be the value of the calculated metric of regression observation $i$ and $\hat{m}_i$ be the appropriate estimated value. Please note that $\hat{m}_i$ is a dependent variable, i.e., it represents an estimation of the value of the metric defined in Step 1 of the preparation phase. A multiple polynomial regression model of order $k, k > 0$, calculates $\hat{m}_i$ according to (4):

$$\hat{m}_i = \hat{\beta}_0 + \sum_{j=1}^{n} \sum_{z=1}^{k} \hat{\beta}_j \times x_{jz}^2 + \tau_i,$$

where $x_{jz}, \forall j \in [1, n], \forall i \in [1, r]$, is the value of the $j$-th independent variable of the $i$-th regression observation, $\hat{\beta}_0$ is the estimated intercept of the regression line, $\hat{\beta}_j$ is the estimated slope for the independent variable $x_j$, $\forall j, \forall i$. $n$ is the number of independent variables, $r$ is the total number of recorded regression observations and $\tau_i, \forall i$ is the error term of the $i$-th regression observation. Note that the independent variables $x_{jz}, \forall j, \forall i$, hold the randomly generated values of the simulation parameters. They are defined in Step 2 of the preparation phase. We define a set of randomly generated values $R_j$, so that $x_{jz} \in R_j, \forall j, \forall i$.

Each observation point, say $o$, represents a unique set of independent variable values denoted as $\{x_{o1}, x_{o2}, \ldots, x_{on}, \ldots, x_{no}\}$. Let for every observation point $o$, the recorded metric values from $r_o$ replications be $M_o : \{m_{1o}, m_{2o}, \ldots, m_{ro}\}$, $r_o < r$, with a mean of $\bar{m}_o$. It has been previously shown that fitting a regression model to individually recorded dependent variable values $m_i$ is equivalent to fitting such a model to the means $\bar{m}_o, \forall o$ [20]. Let $\hat{m}_o$ be the estimated value of $\bar{m}_o$. Then, the (4) can be transformed into (5):

$$\hat{m}_o = \hat{\beta}_0 + \sum_{j=1}^{n} \sum_{z=1}^{k} \hat{\beta}_j \times x_{oj}^2 + \tau_o.$$

The user may define a specific form of this generic regression model. The framework calculates the estimated values of the intercept and the slopes of the regression line by using the least squares method. The task of the least squares method is to minimize the sum $\sum_o (\bar{m}_o - \hat{m}_o)^2$. It does so by searching for the most fitting values of the intercept $\hat{\beta}_0$, as well as of the slopes $\hat{\beta}_j, \forall j$. The framework uses a least squares procedure that takes as input a Vandermonde matrix $M$ of dimensions $n \times k$ (6). Thus, this procedure is suitable for fitting polynomial regression models of an arbitrary order $k, k > 0$.

$$M = \begin{bmatrix} x_{11} & \cdots & x_{12} & x_{1} & 1 \\ \vdots & \ddots & \vdots & \vdots & \vdots \\ x_{n1} & \cdots & x_{n2} & x_{n} & 1 \end{bmatrix}$$

(6)

When the variance of the errors $\tau_o, \forall o$, are heterogeneous, we use the weighted least squares method that minimizes the sum $\sum_o w_o (\bar{m}_o - \hat{m}_o)^2$. In such case, the framework estimates the weights $w_o, \forall o$, as in $\hat{w}_o = \frac{1}{\sigma_o^2}$, where $\sigma_o^2$ is an estimation of the standard deviation of $M_o$.
After the framework calculates the estimated values of the intercept and the slopes, the user may verify the validity of the obtained model by performing residual analysis. For that purpose, the framework plots the residuals \( e_o = (m_o - \hat{m}_o), \forall o \), on three different charts:

1. Residuals vs. each estimated metric \( \hat{m}_o, \forall o \);
2. Residuals vs. each independent variable \( x_{jo}, \forall j, o \);
3. Histogram of residual values.

In case the residual analysis indicates an unfitting regression model, the user may define another one (Fig. 1). Otherwise, the framework tests the significance of the overall model with an overall \( F \)-test for fitness of the obtained regression line. The \( F \)-test checks for the validity of the logical expression \( F_{stat} \geq F_{\alpha} \). The value of \( F_{stat} \) is the \( F \)-test statistic \(^{[10]}\), while the value of \( F_{\alpha} \) is extracted from a built-in \( F \)-table. In case of a multiple regression, the individual contribution of each simulation parameter \( x_j, \forall j \), to the value of the metric of interest is determined with individual \( F \)-tests. If the results of the overall \( F \)-test show that the model is not significant, the user may revert to the preparation phase to redefine the regression model. Otherwise, the framework composes a data file which contains regression analysis results, such as:

1. The result of the overall \( F \)-test;
2. The results from the partial \( F \)-tests [in case of a multiple regression];
3. The slopes and the intercept of the regression line.

Note that for constant and non-constant variances of the errors \( \tau_o, \forall o \), different calculations of the \( F \)-statistics apply \(^{[20]}\).

5 Use-case study

We performed an use-case study in order to observe the benefits of using regression analysis in stochastic simulation. We considered the multirate example scenario as a simulation scenario for this use-case. It is implemented in the file \( \text{ns-3/examples/wireless/multirate.cc} \) in \( \text{ns-3.11} \). The simulation scenario simulates wireless ad-hoc network of 100 nodes with fixed predefined locations. They communicate through multiple simultaneous data flows by utilizing multi-hop routing protocol. The default inter-node distance is set to 50 meters.

We assume that of a research interest is the impact of the inter-node distance on the amount of re-transmitted data in the simulation scenario. We consider the inter-node distance as an independent variable and the amount of re-transmitted data as a dependent variable. Intuitive reasoning leads to the estimative conclusion that in the case of a small inter-node distance, the amount of re-transmitted data should be big. That is because of frequent data packet loss due to collision\(^{[19]}\). We expect the amount of re-transmitted data to linearly decrease as the inter-node distance is increased. Note that after a certain point, we expect the amount of re-transmitted data to rise due to attenuation packet loss\(^{[19]}\). For the sake of simplicity, we focus only on the expected linear decrease of packet loss.

We investigated the value range of the inter-node distance from 10 meter to 30 meters. We generated 15 different inter-node distances in that range. Since the inter-node distance is the only independent variable, \( j = 1, R_j = \{10, 10.4, 11, 11.3, 11.9, 12.7, 14, 16, 17, 19, 21, 22.3, 24, 25.8, 30\}, |R_j| = 15 \) (Section 4.2). We noticed that the variation rate of the metric of interest is higher for the inter-node distances in the range of 10 to 14 meters. Therefore, we generated distance values with smaller steps in that range. We performed 30 replications for each inter-node distance (\( r_o = 30, \forall o \)). That resulted in total number of 450 regression observations (\( r = 450 \)).

We performed residual analysis to confirm that the assumed linear regression model satisfied all the classical regression assumptions. The overall \( F \)-test showed that the regression model is statistically significant (\( F_{stat} = 314.938, F_{\alpha} = 3.92, \alpha = 0.05 \)). We present the obtained linear regression model in (7). \( A_d \) is the amount of re-transmitted data and \( D_n \) is the inter-node distance.

\[
A_d = 1409.26 - 33.532 \times D_n. \tag{7}
\]

In order to quantify the accuracy of the regression model, we performed experimental measurements. The inter-node distances should differ from the ones used for constructing the regression model\(^{[20]}\). Therefore, we

\[ WJMS email for contribution: \text{submit@wjms.org.uk} \]
measured the amount of re-transmitted data for 15 randomly generated inter-node distances in the value range of 10 to 30 meters. So, \( j = 1 \), \( R'_1 = \{10.2, 11.5, 12, 12.5, 13, 15, 16.5, 18, 19, 23, 24.4, 26, 28.2, 29, 29.6\} \), \(|R'_1| = 15 \). In Fig. 2, we present the predicted and the measured amount of re-transmitted data as function of the inter-node distance. The amount of re-transmitted data (predicted and measured) is expressed in Megabytes (MB). The inter-node distance is expressed in meters (m).

![Fig. 2. Measured and predicted amount of re-transmitted data](image)

We also calculated the Standard Error of the Estimate \( \text{SEE} \), the Pearson’s product correlation coefficient \( \rho \), and the difference between the means of the predicted and the measured amounts of re-transmitted data. The \( \text{SEE} \) is 31.77 MB, or 4.09% of the average measured amount of re-transmitted data. The Pearson’s product correlation coefficient is significant, i.e., \( \rho = +0.991, \alpha = 0.05 \). This indicates strong positive linear correlation between the predicted and the measured values. The difference between the means is 10.33 MB.

We consider an analytical model as valid if the predicted and the measured values have identical means and variances\(^\text{[12]}\). Let \( m_t, \forall t \in [1, |R'_1|] \) be the measured amount of re-transmitted data and \( \hat{m}_t, \forall t \) be the appropriate predicted value. We assume a given dependance between the differences \( D_t = m_t - \hat{m}_t \) and the sums \( S_t = m_t + \hat{m}_t, \forall t \) (8). We acknowledge that the predicted and the measured values have identical means and variances if the hypothesis \( H_0 \) holds (9)\(^\text{[12, 20]}\):

\[
D_t = \gamma_0 - \gamma_1 \times S_t + \tau_t,
\]

\[
H_0 : \gamma_0 = 0 \land \gamma_1 = 0.
\]

We regressed the differences \( D_t \) on the sums \( S_t, \forall t \in [1, |R'_1|] \), (10):

\[
D_t = -25.401 + 0.0097 \times S_t.
\]

We tested \( H_0 \) by performing an F-test. We use appropriate F-statistics when the variances of errors \( \tau_t, \forall t \) are constant or varying\(^\text{[12, 20]}\). The F-test checks for the validity of the logical expression \( F^* > F(2, |R'_1| - 2, 1 - \alpha), \alpha = 0.05, |R'_1| = 15 \), where \( F^* \) is the F-statistic. The hypothesis \( H_0 \) is rejected if the aforementioned logical expression holds. The F-test resulted in non rejection of \( H_0 \), since \( F^* = 0.0947 \) and \( F(2, 13, 0.95) = 3.81 \). Thereby, \( F^* > F(2, 13, 0.95) \) does not hold. Hence, we confirm that the regression model given in (7) provides a valid abstraction of the stochastic simulator output.

The presented results confirm that the inter-node distance significantly impacts the amount of re-transmitted data in a linear manner. Furthermore, that relationship is now modelled with an analytical formula which yields conforming values to the output provided by the simulator. This enables prompt calculation of the simulation output and an analytical insight into the results produced by the simulator.
6 Conclusion

In this paper we presented a ns-3 framework with three central features: (i) terminating simulation, (ii) simultaneous executions of simulation scenarios, and (iii) support for regression analysis.

The terminating simulation is a valid and a proved procedure for calculation of statistically accurate simulation results. The implemented terminating simulation feature in the proposed framework provides a fully customizable working environment, i.e., a user can define specific metrics of interest and concretize the way in which a metric is extracted from a trace file. Furthermore, she can specify the desired confidence level and relative precision. The framework automatically performs as many simulation replications as needed until the preset statistical accuracy criterion are met. Additionally, the framework provides simultaneous execution of simulation tasks in order to reduce simulation time. For that purpose, we adopted the MRIP approach.

Finally, we introduce support for regression analysis. We consider metrics of interest as dependent variables and simulation scenario parameters as independent variables. The user has complete control over the implemented regression analysis procedure. The regression analysis is beneficial to identify simulation parameters that have significant impact on given metrics of interest. Further, it may construct an analytical relationship between them. We argue that in some specific cases, one can use a regression model to replace a full-blown simulation procedure. This may lead to calculation of metric values in a considerably shorter time. Furthermore, such a model would enable making better design decisions and discovering optimal points of operation of the simulated entities.

There are several ways to extend this work. First, we plan to implement methods to increase the current fault tolerance of the simultaneous execution feature. Also, we consider the implementation of support for more regression analysis models as a priority.

References


WJMS email for contribution: submit@wjms.org.uk

