COMBINATION AND INTERWORKING OF TRAFFIC-FLOW ANALYSIS AND EVENT-DRIVEN DISCRETE EVENT SIMULATION

Gábor Lencse
Department of Telecommunications
Széchenyi István University
Egyetem tér 1.
H-9026 Győr, Hungary
e-mail: lencse@sze.hu

KEYWORDS

traffic-flow analysis, discrete event simulation, communication networks, performance analysis, traffic modelling

ABSTRACT

The traffic-flow analysis (TFA) is a novel method for quick performance estimation of communication systems. It gives approximate results and is potentially faster than event-driven discrete event simulation. The combination of the two methods seems to be a promising approach for the analysis of a system together with its environment. The combined method can be applied especially well for the performance analysis of the critical parts of communication networks.

INTRODUCTION

The traffic-flow analysis (TFA) (Lencse, 2001) is a simulation-like method for rapid investigation of the traffic conditions of communication networks. Unlike discrete event simulation, TFA does not model the travelling of each packet through the network individually, rather it uses statistics to model the networking load of applications. The method distributes the traffic (the statistics) in the network first, and calculates the specific traffic conditions for each line and switching node in the second step. The results are approximate but may characterize the traffic conditions of the network satisfactorily.

The event-driven discrete event simulation (DES) can be used for detailed and accurate analysis of a system, however if the system is large and complex, the number of the events may be so high that the execution requires unsuitably long time even using a supercomputer. Because of the algorithm of the event-driven DES, parallelisation is not an easy task and the achievable speed-up is often limited. (Fujimoto 1990)

This paper deals with the combination of the two methods. Several times we are faced with the following problem: there is a communication network that has a critical part. E.g. we have an X.25 network servicing ATM and POS terminals, and we would like to check what happens if an important link to the server fails. The common characteristic feature of these types of problems is that there is a critical part of the network (like the immediate neighbourhood of the server in the example above) that should be modelled accurately and there is the all remainder part of the network that cannot be omitted because it gives the load for the critical part. Using event-driven DES for the analysis of such kind of networks, we face with the following contradiction: even though we are interested in the critical part, the vast majority of the events in our simulation occur in the rest of the network (because that contains all but some of the nodes, lines, traffic generators). The basic idea of the suggested solution is the following: DES should be used for the precise analysis of the critical part only, and TFA is to be applied for the rest of the network. This solution has the following justification: the critical part is modelled accurately enough, but the computing power is not wasted for the execution of events that are irrelevant for us individually, only their certain statistical consequences influence the behaviour of the critical part of the network.

For the combination of the two methods, the methods should be taught to "speak the same language" to be able to interchange information with each other. It means that we need conversion methods between the *statistics* of TFA and the *events* of DES in both directions. Additionally, we are faced with the following problem: on one hand virtual time (model time) is very important in the event-driven DES, on the other hand TFA does not deal with the elapsing of time, TFA can be used for taking a snapshot of the system in a given state. Our implementation of TFA – a part of a network expert system called Iminet (Elassys, 2004) – was built on top of an event-driven discrete event simulation engine, and uses the simulator's virtual time for the internal purposes of TFA (for determining the spatial distribution of the traffic – will be explained later).

Let us discuss the issue of the conversion between the statistics and messages first, and solve the virtual time problem after that.

BIDIRECTIONAL CONVERSION BETWEEN STATISTICS AND MESSAGES

General Approach

Let us suppose that our network to be examined can be divided into two parts: one of them is to be simulated by DES and the other is to be analyzed by TFA. A system

built up by several DES and TFA segments can be traced back to this simple case.

The basic idea of the conversion between statistics and messages on the boundary of the two segments is shown in Figure 1. When the traffic information is travelling from the DES segment to the TFA segment, the representation mode is changed from messages to statistics. It means that we need to collect the appropriate statistical characteristics of the message flow, and perhaps it is necessary to transform the results to the kind of statistics that are used in the TFA segment. On the opposite direction, where the traffic travels from the TFA segment to the DES segment, messages should be generated on the basis of the statistics (called *traffic model* in TFA terminology) of the TFA segment. The characteristics of the message flow such as the distribution of the message length, inter-arrival time, the source and destination of the packets are representing the same information that is coded in the statistics travelling from the TFA segment to the DES segment.

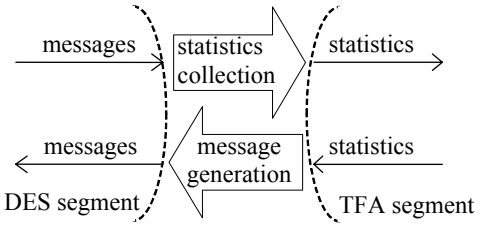

Figure 1. The Basic Idea of Connecting a DES and a TFA Segment

How can these conversions be done between the two representation forms of the traffic? Not too much can be said about it in general, as TFA is a general method which can work with various traffic models that comply with the *requirements for the traffic model* presented in the original paper. (Lencse, 2001) First, we shall examine the problems that the conversion between the traffic description methods of TFA and DES may cause in general, and then we deal with some of the issues in more specific cases.

When doing system modelling, we focus on some features of the system that are important for us and neglect some other details that we consider less important from our point of view. Let us consider the following example: we would like to examine the performance of a network and find its possible bottlenecks. To make a traffic model for DES, we probably carefully examine (for all traffic sources) the distribution of the packet length and of the inter-arrival time as well as the correlation of the two, (and perhaps some other features such us the burstiness of the traffic) but we are not really interested in the contents of the packets (the information they carry). To make a traffic model for TFA, the traffic sources could be characterized by their *packet throughput distribution* (a histogram of the number of packets per seconds) to model the load for the switching nodes and by their *bit throughput distribution* (a histogram of the number of bits per seconds) to model the load for the lines. Using either DES or TFA, we design a model, so that the results of the experiments with the model answer our questions. However, during modelling, a given portion of our efforts is done so that the model meet the requirements of the given simulation system. Such as we need to use the given topology description language, we have to code the activities of the elements in e.g. in C++, etc. It is also true in the case if we would like to combine TFA and DES. ***We must build both models so that they could be combined.*** As for the traffic exchange, it means that the traffic model of both the TFA segment and the DES segment need to contain enough information so that it could be converted to the other type and the result of the conversion will satisfy the requirements of the traffic model of the target segment. This is what we can say in general.

Requirements of a Typical Traffic Model

In the example above, we mentioned bit-throughput and packet-throughput as possible TFA traffic models. They were also proposed and described in details in the original paper on TFA (Lencse, 2001). As they are nothing else than *histograms*, if we choose them, we can use a lot of previous results. A good book in the topic of non-parametric density estimation is (Devroye and Győrfi 1985). The usage of histograms for statistics collection was examined in (Lencse 1998b). That research was done in the interest of the Statistical Synchronisation Method (SSM) (Pongor 1992), which is a less well known, but promising synchronisation method for parallel DES. We can borrow a lot of results of SSM research. A short summary of the *Statistical Synchronisation Method* is given here. See (Pongor 1992) for more information.

Similarly to other parallel discrete event simulation methods, the model to be simulated – which is more or less a precise representation of a real system – is divided into segments, where the segments usually describe the behaviour of functional units of the real system. The communication of the segments can be represented by sending and receiving various messages. For SSM, each segment is equipped with one or more input and output interfaces. The messages generated in a given segment and to be processed in a different segment are not transmitted there but the *output interfaces* (OIF) collect statistical data of them. The *input interfaces* (IIF) generate messages for the segments according to the statistical characteristics of the messages collected by the proper output interfaces.

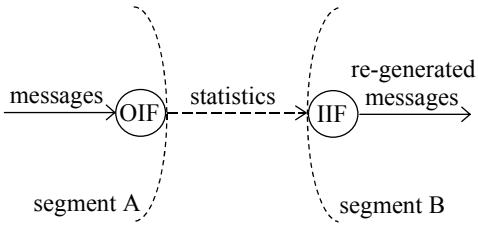

Figure 2. An OIF - IIF Pair

The segments with their input and output interfaces can be simulated separately on separate processors, giving statistically correct results. Events in one segment have not the same effect in other segments as in the original model, so

the results collected during SSM are not exact. The precision depends on the partitioning of the model, on the accuracy of statistics collection and regeneration, and on the frequency of the statistics exchange among the processors.

Histograms were used for representing statistics in the experiments testing the behaviour (Lencse 1997) and the achievable speedup (Lencse 1998a) of SSM.

Despite of the fundamental difference between the parallel DES using SSM and the combination of DES and TFA, they both can use the same or similar statistics collection and message generation mechanisms.

If we use histograms for the implementation of the traffic model of TFA, we can reuse the results achieved in (Lencse 1998b). In that paper, not only histograms, but a number of other statistics collection methods were compared and applied in the case of different distributions in order to determine what non-parametric density estimation methods should be used for statistics collection. The examined methods were: Barron estimate, equidistant histogram (histogram with equal bin/cell width) equi-probable bin histogram and also the relative frequency method for discrete distributions. Both their resource requirements and accuracy were examined. The L_1 error criterion was used for measuring the error of the estimation methods. The conclusions were following. Though theoretically, the equi-probable bin histogram should produce less L_1 error than the equidistant one, and it was also the experience in the case when too few bins were used, but if there were enough bins, the equidistant histogram was better. For continuous distributions, the equidistant histogram is the method of choice. The relative frequency method produced acceptable results for the examined real life discrete or quantized distributions.

It is very likely that histograms will satisfy our needs, and knowing the modelled system well enough we can determine the range and the cell size of the histogram if we are faced to one-dimensional distributions. However it might happen that in some cases we do not know enough a-priori information about some of the distributions or the number of observations is not enough to guarantee the accuracy of the results using histograms. In this case the messages-to-statistics conversion may be done in two steps: First, another method can be used for statistics collection and the results should be transformed to histograms after completing the statistics collection. A good candidate is the K-split method (Varga 1998). K-split maintains the number of cells (i.e. bins) optimal for the distribution and the number of observations by doing cell splits. If we have multi-dimensional distributions, this two step procedure (using K-split first and convert to histograms later) is strongly advised.

DEALING WITH THE DIFFERENT USAGE OF VIRTUAL TIME

The issues of Combination and Interworking

Under *combination* of TFA and DES we mean that we would like to use them for the same network, one method for one part and the other method for the other part of the system. The bidirectional conversion between the two kinds of traffic representations (discussed in the previous chapter) is a necessary precondition; otherwise the two parts can not exchange traffic information with each other. However if we would like TFA and DES to *interwork*, that is actively cooperate with one another, we need more. Before going into the details, let us consider how they use the virtual time – that will be the key issue.

The algorithm of the event-driven DES is the following (FES means the Future Event Set):

```
initialize, insert certain events into the FES;
repeat
  remove the first event from the FES;
  NOW:=the virtual time of the event removed from the FES;
  process the event, during this insert some event(s) into the
        FES if necessary;
until (FES is empty) or (NOW>limit) or (for other reason, we
        must stop)
```

Virtual time plays a key role in this algorithm.

TFA gives a snapshot of the traffic conditions of the examined system, that is, it analyses the network at a given point of virtual time. Though virtual time does not take part in TFA as in DES (as it would increase/elapse during the analysis) it is usually important and influences the results through the parameters of the system. For example, if we model POS (Point of Sale) or ATM (Automatic Teller Machine) terminals in a network, the frequency of the transactions highly depends on the hour of day (and also on the day of week).

Now, let us return to the definition of ***interworking of TFA and DES***. Under it we mean the following cooperation (one or more times it happens that):

1. DES sets up the virtual time of TFA
2. DES provides input parameters to TFA
3. DES calls TFA for the evaluation of the part of the system that was trusted for TFA
4. TFA runs
5. TFA returns parameters to DES

The way these steps are done highly depends of the architecture and communication scheme of the program(s). Let us see some possibilities:

A) The DES part and the TFA part are two processes and communicating with each other by inter-process communications (using PVM/MPI or named pipes, etc.).

B) TFA is a function/procedure within the DES program and is called sometimes, but it does not use the virtual time of the DES engine for its internal purposes.

C) TFA is a set of functions within the DES program and is called sometimes, and it uses the virtual time of the DES engine (and some of its services) for its internal purposes.

All of them could be useful, but may have some weaknesses too. Solutions "A" may not be feasible with commercial DES programs like OPNET Modeler (OPNET, 2004) that are usually not distributed with the full source code and therefore it would be inconvenient or sometimes practically impossible to use inter-process communications.

"B" can be the method of choice, if we use a commercial simulator that is not flexible enough for "C" or for some reason we prefer this solution. By choosing "B" we avoid all the issues to be discussed in the remainder of this paper, but also miss all the benefits of using the functionalities of the simulator. Then we are on our own and have to implement everything ourselves in the TFA part, including probably the topology description and routing of the network. We have no help from the simulator, no or little possibilities for reuse of the ready made elements. If we write a function that is once called, then works and finally returns we cannot use the event mechanism of the simulator, that is: we lose the chance to use all the models that are built up in the event-driven manner.

It is worth choosing "C", if we are able to use a lot a functionalities of the simulator in the TFA part and it will reduce our programming (also model building) efforts significantly. From now on, we concentrate on the different issues we are faced to when choosing "C".

What do the steps of interworking mean if we use the virtual time of the simulation system within TFA? (That is, we use the event mechanism of the simulator.) Let us consider them one by one.

1. DES sets up the virtual time of TFA by scheduling its starting event to the appropriate virtual time

2. DES provides input parameters to TFA by sending messages to the module(s) responsible for TFA

3. DES not at all calls TFA, it runs when its virtual time comes

4. when TFA runs, it may use the event scheduling system, but it needs some discussion (because of the contradictory usage of virtual time)

5. TFA returns parameters to DES by sending messages to the appropriate module(s) in the DES segment

All but no. 4 are quite clear. The issue of virtual time usage is discussed in the text subsection.

How to conciliate the virtual time usage in DES and TFA?

First, let us recall how TFA distributes the traffic in the network. It pursues the following procedure for all traffic sources: first, it divides the statistics (describing the traffic of the given source) into smaller ones, and then sends them to their target one by one. Their size is called the *size of routing unit* (S_{RU}) and they are routed one by one in the network according to the same algorithm that the nodes use for routing the packets in the network (in DES). Of course, S_{RU} have to be chosen so that the granularity should be smooth enough (to produce good enough results) on the one hand; however, the number of routing units should be at least one order of magnitude less then the number of packets would be in DES (to be faster then DES) on the other hand. As we mentioned, the statistics can be sent by the source to the destination as messages, and the switching nodes and lines can forward them as messages. The nodes can use the same routing algorithm for routing the statistics as DES would use for routing the packets. This is an important benefit that we do not need to write everything from scratch. In an object oriented environment we can reuse the granted elements even if we must slightly modify their behaviour. A very important question is that how the statistics that are coming from different sources and travelling through the same network can be interwoven well? This is crucial for the appropriate spatial distribution of the traffic in the network. There are two solutions that both reconcile the virtual time usage of TFA and DES, but in a different manner. With a little bit of irony, we can call the first one *model hacking* and the second one *kernel hacking*.

Under *model hacking* we mean that we have to construct the DES model so that it allow a given small but positive T virtual time for TFA to work. T should be chosen small enough so that the DES model can tolerate that even though it starts TFA at t virtual time, it can receive back the results from TFA at $t+T$ virtual time. For the interweaving of the traffic from different sources, let us consider two sources **A** and **B** that send N and M number of statistics respectively. The virtual times of the sending of the i-th statistics by **A** and **B** are:

$$t_{A,i} = t + i\frac{T}{N}, \qquad i = 0,\quad 1,\quad \ldots\quad N-1$$

$$t_{B,i} = t + i\frac{T}{M}, \qquad i = 0,\quad 1,\quad \ldots\quad M-1$$

When distributing the statistics, TFA does not model processing or transmission delay, so the statistics travel with the same time stamp through the network. A finishing event is scheduled for all the nodes and lines of the TFA model with the time stamp of $t+T$. This event causes that the nodes and lines transform the previously summed up statistics so that their finite capacities are considered. See the

algorithm in the original paper (Lencse, 2001), and the proof of its convergence in (Lencse and Muka, 2005).

The advantage of model hacking is that (unlike kernel hacking) this solution does not require any modifications of the simulation kernel. Hopefully our DES model will tolerate the small *T* long latency of the TFA segment. Otherwise we need to make some modifications on our DES model – remember, the method is called model hacking.

Kernel hacking provides a cleaner way for the interweaving of the traffic from different sources. However it requires some modifications in the DES simulation kernel. It can be done if either the kernel is our own property just like Elassys Consulting Ltd. owns the Iminet Network Expert System (Elassys, 2004) that contains a DES kernel or the simulator is open source such as the OMNeT++ Discrete Event Simulation System (Varga, 2004). The modification of the simulation kernel means the introduction of sub-time, that is, a second time stamp is added to all the events, and the events are ordered primarily on the basis of the original time stamp, but if the original time stamp of two messages are equal their order is determined on the basis of the second time stamp. Of course, the second time stamp of the normal DES events should be set to zero. Denoting the complete time stamp of the events by separating the original and the second time stamp with a comma, the scheduling times of the TFA statistics sending events in the previous example look like:

$$t_{A,i} = t, i\frac{1}{N}, \quad i = 0, \quad 1, \quad \ldots \quad N-1$$

$$t_{B,i} = t, i\frac{1}{M}, \quad i = 0, \quad 1, \quad \ldots \quad M-1$$

Of course, any other fixed constants could stand in the nominator instead of "1".

CONCLUSIONS

We have examined how the two methods, TFA and DES can be combined for more efficient performance estimation of communication networks.

We have shown how TFA and DES can interchange information using a bidirectional conversion between their traffic representation forms: statistics and messages. We found that our previous research results on statistics collection methods are reusable.

We have shown two ways for resolving the issue of different (and contradicting) virtual time usage of TFA and DES.

We conclude that the integration of TFA and DES is promising and building the TFA package on top of a flexible DES kernel is a practical way of implementation.

ACKNOWLEDGEMENT

This research was financed by the Hungarian Academy of Sciences under the János Bolyai research grant.

REFERENCES

Devroye, L.; L. Győrfi. 1985. *Nonparametric Density Estimation: The L1 view.* John Wiley, New York

Elassys Consulting Ltd. 2004. "Iminet Network Expert System" http://www.elassys.hu

Fujimoto, R. M. 1990. Parallel Discrete Event Simulation. *Communications of the ACM* **33** no. 10, 31-53

Lencse, G. 1997. "Efficient Simulation of Large Systems - Transient Behaviour and Accuracy" *Proceedings of the 1997 European Simulation Symposium (ESS'97)* (Passau, Germany, Oct. 19-23). SCS Europe, 660-665.

Lencse, G. 1998a. "Efficient Parallel Simulation with the Statistical Synchronization Method" *Proceedings of the Communication Networks and Distributed Systems Modeling and Simulation (CNDS'98)* (San Diego, CA. Jan. 11-14). SCS International, 3-8.

Lencse, G. 1998b. "Statistics Collection for the Statistical Synchronisation Method" *Proceedings of the 1998 European Simulation Symposium (ESS'98)* (Nottingham, UK. Oct. 26-28). SCS Europe, 46-51.

Lencse, G. 2001. "Traffic-Flow Analysis for Fast Performance Estimation of Communication Systems" *Journal of Computing and Information Technology* **9**, No. 1, 15-27

Lencse, G; L. Muka 2005. (expected) "Convergence of the Key Algorithm of Traffic-Flow Analysis" *Journal of Computing and Information Technology,* - under review,

OPNET 2004. "OPNET Modeler" http://www.opnet.com

Pongor, Gy. 1992. "Statistical Synchronization: a Different Approach of Parallel Discrete Event Simulation". *Proceedings of the 1992 European Simulation Symposium (ESS'92)* (Nov. 5-8, 1992, The Blockhaus, Dresden, Germany.) SCS Europe, 125-129.

Varga, A. 1998. "K-split – On-Line Density Estimation for Simulation Result Collection". *Proceedings of the 1998 European Simulation Symposium (ESS'98)* (Nottingham, UK. Oct. 26-28). SCS Europe, 41-45.

Varga, A. 2004. "OMNeT++ Discrete Event Simulation System" http://www.omnetpp.org

BIOGRAPHY

Gábor Lencse received his M.Sc. in electrical engineering and computer systems from the Technical University of Budapest in 1994 and his Ph.D. in 2000. The area of his research is (parallel) discrete event simulation methodology. He is interested in the acceleration of the simulation of communication systems. Since 1997, he works for the Széchenyi István University in Győr. He teaches computer networks and networking protocols. Now, he is an associate professor. He does research and development in the field of the simulation of communication systems for the Elassys Consulting Ltd since 1998.

He is a member of IEEE Hungary Section, IEEE Computer Society and IEEE Communications Society.