MOSEL-2 - A Compact But Versatile Model Description Language And Its Evaluation Environment*

Patrick Wüchner[1], Hermann de Meer[1], Jörg Barner[2], Gunter Bolch[2]

[1] Chair of Computer Networks and Computer Communications, Faculty of Mathematics and Computer Science, University of Passau, Innstr. 33, 94032 Passau, Germany. {patrick.wuechner, hermann.demeer}@uni-passau.de

[2] Research Group on Analytical Modeling, Chair of Distributed Systems and Operating Systems, Department of Computer Science, University of Erlangen-Nürnberg, Martensstr. 1, 91058 Erlangen, Germany. {barner, bolch}@informatik.uni-erlangen.de

Abstract: In this paper we present the current version of the MOdeling, Specification and Evaluation Language MOSEL-2 and show its applicability for performance and reliability modeling and evaluation of systems with Markovian and non-Markovian behavior. The tool MOSEL-2 consists of two major components: the *description language* and the *evaluation environment*.

The *description language* is the core element of MOSEL-2 and provides a high-level means for specifying models, performance measures, and the graphical presentation of the results of these measures.

MOSEL-2's *evaluation environment* includes a set of model translators that allow automatic translation of MOSEL-2 models to the model descriptions of several third-party performance evaluation tools. MOSEL-2 uses these tools to evaluate the model by numerical analysis or simulation. The results returned are collected by MOSEL-2 and presented in a unified textual and graphical form. This novel concept exempts the users from learning a new model specification language and rewriting all models each time they have to change a probability distribution within their models.

A new and unique concept has been added that allows the automatic approximation of non-Markovian distributions by Markovian constructs. The goal of our ongoing research is to enhance the modeling and evaluation power of MOSEL-2 steadily. This paper aims to give an introduction to the current version of the MOSEL-2 language and its evaluation environment demonstrating the most recent improvements.

Keywords: MOSEL-2, model description language, evaluation environment, performance modeling

1 Introduction

Performance and reliability modeling and the evaluation of these models play a major role in the design, development, testing, and maintenance of systems in many application areas. These include computer systems, communication systems and networks, manufacturing systems, and various others. In many cases, the following properties are characteristic for the real-world systems under investigation: Their dynamic evolution proceeds from one discrete state to another at arbitrary moments in time. Often they are composed of smaller subsystems which run in parallel and cooperate in order to fulfil a common task. They can be distributed and may react to stimuli which are triggered by the system's environment. One of the most frequently used techniques to model and evaluate the performance of such a system at the dynamic level is to represent it by a stochastic process which – roughly speaking – consists of all system states and all possible transitions between them. For an important class of stochastic processes, known as Continuous Time Markov Chains (CTMCs) [1], standard numerical algorithms can be used to compute the state probabilities. For systems whose dynamics can be described by only a few states, an experienced modeler is able to deduce the underlying stochastic process

*Parts of this work have been accomplished under support of the Euro-NGI - Network of Excellence, EC grant IST-50190293.

directly by inspection. Unfortunately, this is impossible for many interesting real-world systems since they tend to possess a large number of states at the stochastic process level. In this case, the investigator has to resort to a formal high-level modeling technique which is based on some mathematical theory and allows expressing some or all of the system properties mentioned above at a static level.

MOSEL-2 is a modeling environment which comprises a high-level modeling language that provides a very simple way for system description. In the design of the MOSEL-2 language, particular emphasis was placed on keeping the MOSEL-2 model specifications compact and still easy to understand. For example, models comprising repeating elements can be specified conveniently exploiting MOSEL-2's syntax pre-processor and MOSEL-2 supports the evaluation of models with different sets of system parameters.

In order to reuse existing tools for the system analysis, the environment is equipped with a set of translators which transform the MOSEL-2 model specification into the tool-specific system descriptions of various third-party performance evaluation tools. MOSEL-2 uses these tools to evaluate the model by numerical analysis or simulation. The results returned are collected by MOSEL-2 and presented in a unified textual and graphical form.

The benefit of MOSEL-2 is that the same MOSEL-2 model can be evaluated using different performance evaluation tools and their various methods. Currently, the tools MOSES, SPNP, and TimeNET are supported by MOSEL-2. The evaluation methods provided by these tools include numerical solvers for Markovian models, numerical solvers for a restricted class of models with non-exponentially distributed transitions, and DES for non-Markovian models.

The paper is organized as follows. In Section 2 work related to the MOSEL-2 project is summarized. In Section 3 the MOSEL-2 modeling and evaluation process is shown. Recent improvements of MOSEL-2 are presented in Section 4. Section 5 concludes this paper.

2 Related Work

In the last decades, several modeling formalisms have been developed and used for performance and reliability modeling in various problem domains. Among these, stochastic process algebras (SPAs) [2] and stochastic Petri nets (SPNs) with their numerous derivates [3] such as general stochastic Petri nets (GSPNs), deterministic and stochastic Petri nets (DSPNs), and extended stochastic Petri nets (ESPNs) turned out to be especially useful as they allow a combined description of the functional and temporal system behavior including the specification of phenomena like priorities, synchronization and preemption. Queueing network systems [4] is another widely used modeling formalism which does not reach the expressiveness of SPNs and SPAs but instead supports solution algorithms by which the desired performance measures can be calculated extremely fast.

SPNs and SPAs are high-level modeling formalisms in which the modeler gives a static description of the system structure and behavior using a textual (SPAs, SPNs) or graphical (SPNs) syntax. In order to evaluate the functional and temporal behavior the system has to be either simulated or analyzed depending on the ability to generate the whole state space.

Fortunately, this no longer has to be carried out manually. There exists a variety of (freeware and commercial) tools which are based on one of the high-level modeling formalisms mentioned above, e.g.: SPNP [5], TimeNET [6], GreatSPN [7], DSPNexpress [8, 9], and WebSPN [10] are based on Petri nets, whereas TIPP [11] and PEPA [12] are built around the SPA modeling formalism. Other packages, for example PEPSY [13], Win-PEPSY [14], and QNAP2 [15] favor the queueing theoretical approach, whereas SHARPE [16], Möbius [17], and MOSES [18] use a mixture of methods towards the generation and solution of the underlying CTMC.

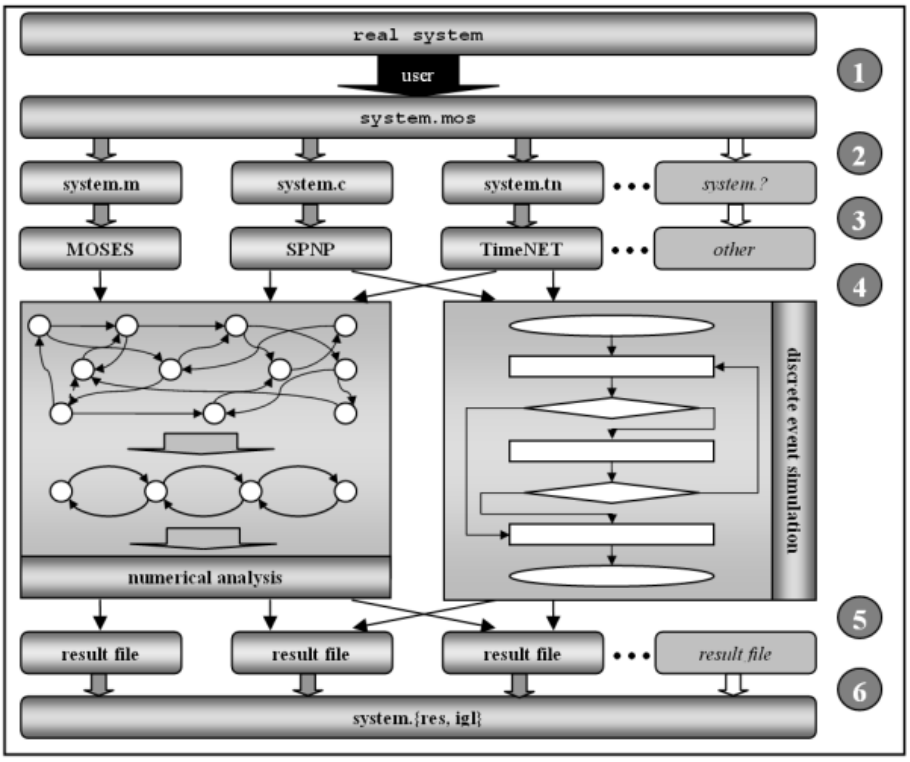

Figure 1: Modeling and evaluation process using MOSEL-2.

All these packages usually have their own textual or graphical specification language which depends largely on the underlying modeling formalism. The different syntax of the tool-specific modeling languages implies that once a tool has been chosen it will be difficult to switch to another one as the model has to be rewritten using a different syntax.

3 The MOSEL-2 Concept

Starting from these observations the development of MOSEL [19] and its successor MOSEL-2 [20] is based on the following idea: Instead of creating another tool with all the components needed for system description, state space generation, stochastic process derivation, and numerical solution or simulation, the focus was set on the formal system description part and on exploiting the power of various existing and well tested packages for the subsequent stages.

3.1 The Modeling and Evaluation Process

Figure 1 gives an overview of performance and reliability modeling and evaluation using the MOSEL-2 environment:

1. The modeler inspects the real-world system and generates a high-level system description using the MOSEL-2 specification language. He also specifies the desired performance and reliability measures using the syntax provided by MOSEL-2. He passes the model to the evaluation environment which then performs all following steps without user interaction.

2. The MOSEL-2 environment automatically translates the MOSEL-2 model into a tool-specific system description, for example a CSPL-file (C based Stochastic Petri net Language) suitable to serve as input for SPNP.

3. The appropriate tool (i.e. SPNP) is invoked by the MOSEL-2 environment.

4. The appropriate tool processes its input file in one of the two following ways:

   Numerical analysis: Out of the static model description the whole state space of the model is generated by the tool according to the semantic rules of its modeling formalism. This semantic model is mapped onto a stochastic process. The stochastic process is solved by one of the standard numerical solution algorithms which are part of the tool.

   Simulation: The model is evaluated by the tool without building the whole state space using discrete event simulation.

5. The results of the numerical analysis or discrete event simulation are saved in a file with a tool specific structure.

6. The MOSEL-2 environment parses the tool specific output and generates a textual result file (system.res) containing the performance and reliability measures which the user specified in the MOSEL-2 system description. If the modeler requested graphical representation of the results, a second file (system.igl) is generated by MOSEL-2.

Currently, the MOSEL-2 environment is able to translate models into system descriptions for the SPN-based packages SPNP and TimeNET and for the tool MOSES, whose model description language MOSLANG is a predecessor of MOSEL and MOSEL-2. As indicated in Figure 1, the connection of other performance modeling packages, e.g., SHARPE, Möbius, and DSPNexpress, to the MOSEL-2 environment is projected.

3.2 The MOSEL-2 Specification Language

During the integration of TimeNET, the MOSEL syntax was widely revised and enhanced [20]. The current version is called MOSEL-2.

The basic elements of a MOSEL-2 model are *nodes* and *rules*.

Nodes are used to describe the model's current state. In a specific state, each node has a certain value, which is an integer number in the range from 0 to a node-dependent capacity. Nodes are comparable to the places of SPNs.

Rules are used to describe the state changes. A rule may change the state by setting some nodes to a specific value or by incrementing or decrementing the values of some nodes. A rule is enabled in a subset of states. This subset is specified by the rule's explicit conditions, and implicitly by some of the rule's actions; for example, a rule may not set a node to a negative value or to a value that exceeds its capacity. Several rules may be enabled at the same time. Each rule is given a probabilistic firing time distribution. This firing time distribution specifies how much time will pass from the point when the rule has been enabled up to the point when the rule will be executed. Rules are comparable to the transitions of SPNs.

Exemplary, a MOSEL-2 specification of a closed tandem queueing network is shown in Figure 2. The line numbers in this example are given for referencing only. They are not part of the MOSEL-2 syntax.

A MOSEL-2 specification can be divided into six parts: the optional constant and parameter part (lines 1 to 6), the node part (lines 8 to 11), the optional function and condition part (unused in this example), the rule part (lines 13 to 16), the result part (lines 18 to 23), and the optional picture part (lines 25 to 30).

```
01 // Parameter declaration part
02
03 PARAMETER K := 1 .. 10;
04
05 CONST mue1 := 0.28;
06 CONST mue2 := 0.22;
07
08 // Component definition part
09
10 NODE N1[K] := K;
11 NODE N2[K] := 0;
12
13 // Transition definition part
14
15 FROM N1 TO N2 RATE mue1;
16 FROM N2 TO N1 RATE mue2;
17
18 // Result part
19
20 PRINT rho1 := UTIL (N1);
21 PRINT rho2 := UTIL (N2);
22 PRINT throughput := rho2 * mue2;
23 PRINT k1 := MEAN (N1);
24
25 // Picture part
26
27 PICTURE "utilization"
28 PARAMETER K
29 CURVE rho1
30 CURVE rho2;
```

Figure 2: MOSEL-2 model of a closed tandem queueing network.

A more detailed description of the MOSEL-2 syntax and semantics can be found in [20]. MOSEL-2 language extensions supporting non-Markovian distributions, that can be evaluated with the help of SPNP are provided in [21] and [22].

4 Recent Improvement: The Rule Pre-Processor

The most recent improvement is the introduction of a rule pre-processor by [22]. The rule pre-processor allows automatic approximation of rules with non-Markovian distributed firing times by phase-type constructs. These constructs comprise exponential distributions only and can thus be evaluated using Markovian analysis.

Currently, MOSEL-2 is able to pre-process arbitrarily distributed firing times that can be defined by their mean $\bar{t}$ and their variance σ^2. Due to the fact, that the mean and the variance can be obtained quite easily from measurements, this subclass of distributions is referred to as *empirical distributions*. The MOSEL-2 language element for such distributions is `EMP(`$\bar{t}$`, `σ^2`)`. In Figure 3 the corresponding SPN is shown. The dotted arcs in Figure 3 denote arbitrary input, output, and inhibitor arcs from and to the remaining model.

Knowing $\bar{t}$ and σ^2, the square coefficient of variation (SCV) $c^2 = \frac{\sigma^2}{\bar{t}^2}$ can be calculated. According to c^2 the pre-processor chooses the phase-type construct that will be used to approximate the rule with empirical distribution:

If $c^2 = 1$, then the empirical distribution is equivalent to an exponential distribution with rate parameter $\bar{t}^{-1}$. Therefore, the empirically distributed transition of Figure 3 can be substituted by the exponentially distributed transition shown in Figure 4.

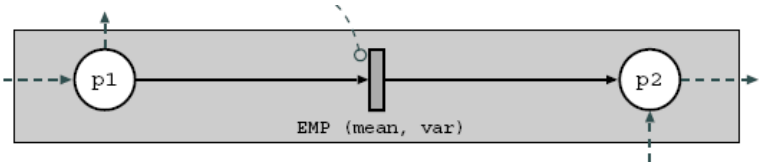

Figure 3: SPN model comprising an empirically distributed firing time.

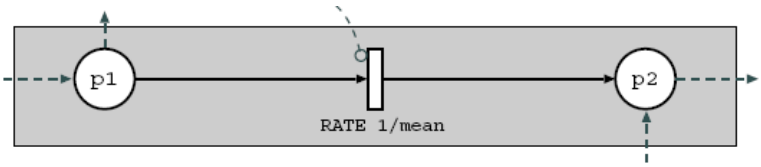

Figure 4: SPN model of the exponential substitution.

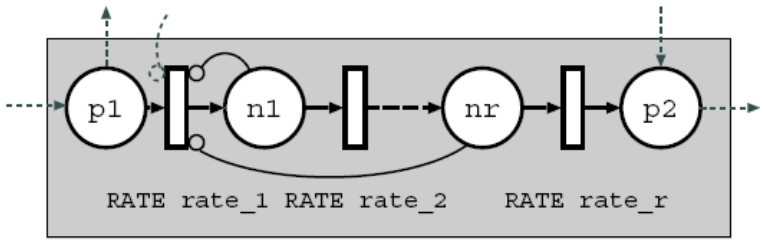

Figure 5: SPN model of the hypoexponential substitution.

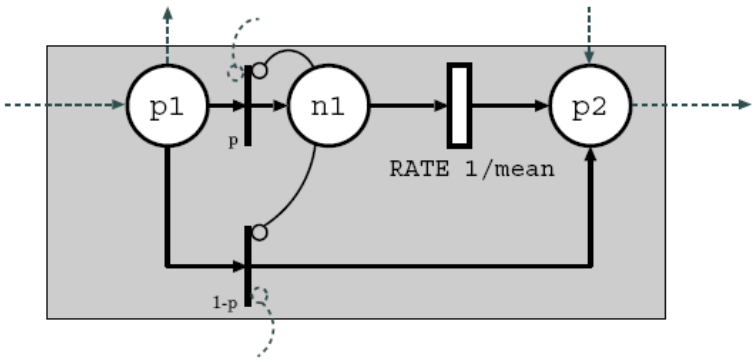

Figure 6: SPN model of the generalized exponential substitution.

If $c^2 < 1$, then the empirical distribution is substituted by a hypoexponential distribution (see Figure 5). The number of exponentially distributed stages needed can be calculated using $r = \lceil \frac{1}{c^2} \rceil$. The rates of these stages can be obtained using a recursive algorithm introduced in [23].

If $c^2 > 1$, then the generalized exponential (GE) distribution is chosen to approximate the empirical distribution (see Figure 6). The rate parameter of the GE distribution is set to $\bar{t}^{-1}$ and the transition probability can be calculated by $p = \frac{2}{c^2+1}$. Unlike the hypoexponential substitution and possible alternative Coxian substitutions, the GE distribution has the advantage that only one additional node has to be introduced – independent of the value c^2.

The details of these concepts and their implementation can be found in [22]. It can be shown that a MOSEL-2 model comprising an EMP-rule with an SCV of 0.0001 can be pre-processed and translated by MOSEL-2 to, e.g., SPNP's model description language CSPL. The translated model file has a size of about 2 MB. It contains over 10000 nodes and as many transitions. Unfortunately, the evaluation tools used by MOSEL-2 cannot handle models of this size. Tests showed, that SPNP is able to solve a simple model containing a single EMP distributed rule with SCVs down to 0.004675 (214 generated nodes) and that TimeNET can solve the same model with SCVs down to 0.017245 (58 generated nodes). MOSEL-2 itself does not restrict the parameters of the EMP distribution, because the smallest working SCV strongly depends on the complexity of the entire model and on the evaluation tool chosen. As a rule-of-thumb, SCVs smaller than 0.05 should be avoided. Nevertheless, a fixed lower limit with this value would be too restrictive.

5 Conclusion and Future Work

MOSEL-2 provides a textual model specification language, that is very compact and easy to learn. The model translators of MOSEL-2 give access to various performance evaluation tools and their different evaluation methods without forcing the user to learn the high-level description language of these tools. Several concepts of MOSEL-2 prevent the user from tedious tasks while modeling complex systems. As an example, the rule pre-processor was introduced in this paper.

Currently and in future we will be working on these topics:

More and more memory will be available in future computer systems for the state space generation and more and more algorithms are developed to allow the numerical analysis of non-Markovian models. Therefore, it will be desirable, to make the arising tools available to MOSEL-2 by providing appropriate translators.

The pre-processor will be extended to handle further phase-type distributions like the Erlang, hypoexponential, hyperexponential, or Cox distribution.

The result measures of some evaluation tools do not provide support for complex terms. We are currently rewriting MOSEL-2's tool specific input file generators (i.e. model translators) and the result parser in such a manner, that only the basic result measures (MEAN, PROB and UTIL) are requested from the evaluation tool. MOSEL-2's result parser can then use the results of this basic measures to compute the complex results that were desired by the user. This will also faciliate the introduction of new evaluation tools to MOSEL-2.

Improving MOSEL-2's documentation, which is currently available via several semester and diploma theses only.

Using MOSEL-2 for modeling and evaluation of complex systems with Markovian and non-Markovian behavior to prove its applicability and user friendliness.

References

[1] G. Bolch, S. Greiner, H. de Meer, and K. Trivedi, *Queueing Networks and Markov Chains*, John Wiley & Sons, New York, 1998.

[2] H. Hermanns, U. Herzog, and J.-P. Katoen, "Process algebra for performance evaluation," *Theoretical Computer Science*, vol. 24, no. 1-2, pp. 43–87, November 2000.

[3] G. Ciardo, R. German, and C. Lindemann, "A characterization of the stochastic process underlying a stochastic Petri net," *IEEE Transactions on Software Engineering*, vol. 20, pp. 506–515, 1994.

[4] J. Mehdi, *Stochastic Models in Queueing Theory*, Academic Press, 2 edition, 2003.

[5] C. Hirel, B. Tuffin, and K. S. Trivedi, "SPNP: Stochastic Petri Nets. version 6.0," in *Proceedings of the 11th International Conference on Computer Performance Evaluation: Modelling Techniques and Tools*. 2000, pp. 354–357, Springer-Verlag, London, UK.

[6] A. Zimmermann, J. Freiheit, R. German, and G. Hommel, "Petri Net modelling and performability evaluation with TimeNET 3.0," in *Proc. 11th Int. Conf. on Modelling Techniques and Tools for Computer Performance Evaluation (TOOLS'2000)*. 2000, vol. 1786 of *LNCS*, pp. 188–202, Springer.

[7] G. Chiola, G. Franceschinis, R. Gaeta, and M. Ribaudo, "GreatSPN 1.7: GRaphical Editor and Analyzer for Timed and Stochastic Petri Nets," *Performance Evaluation*, vol. 24, no. 1,2, pp. 137–159, November 1995.

[8] C. Lindemann, *Stochastic Modeling using DSPNexpress*, Oldenburg, Munich, 1994.

[9] C. Lindemann, A. Thuemmler, A. Klemm, M. Lohmann, and O. Waldhorst, "Quantitative system evaluation with DSPNexpress 2000," in *Proc. 2nd Int. Workshop on Software and Performance (WOSP), Ottawa, Canada*, 2000, pp. 12–17.

[10] A. Bobbio, A. Puliafito, M. Scarpa, and M. Telek, "WebSPN: Non-Markovian Stochastic Petri Net tool," in *Int. Conf on WEB-based Modeling and Simulation*, January 1998.

[11] N. Götz, U. Herzog, and M. Rettelbach, "TIPP – Introduction and application to protocol performance analysis," in *Formale Beschreibungstechniken für verteilte Systeme*, H. König, Ed. Saur, 1993.

[12] J. Hillston, "PEPA: Performance Enhanced Process Algebra," Tech. Rep. CSR-24-93, University of Edinburgh, March 1993.

[13] G. Bolch and M. Kirschnick, "PEPSY-QNS - Performance Evaluation and Prediction SYstem for Queueing NetworkS," Technical report TR-I4-21-92, Department of Computer Science, University of Erlangen, Germany, Oct. 1992.

[14] P. Bazan, B. Bolch, and R. German, "WinPEPSY-QNS performance evaluation and prediction system for queueing networks," in *Proc. of the 11th International Conference on Analytical and Stochastic Modelling Techniques and Applications (ASMTA'04)*, K. Al-Begain and G. Bolch, Eds. June 2004, pp. 147–150, SCS-European Publishing House, Erlangen, Germany.

[15] D. Potier and M. Veran, "Qnap2: A portable environment for queueing systems modelling," in *Modelling Techniques and Tools for Performance Analysis*, pp. 25–63. North-Holland, Amsterdam, 1985.

[16] R. Sahner, K. S. Trivedi, and A. Puliafito, *Performance and Reliability Analysis of Computer Systems – An Example-Based Approach Using the SHARPE Software Package*, Kluwer Academic Publishers, Boston, M.A., 1996.

[17] D.D. Deavours, G. Clark, T. Courtney, D. Daly, S. Derisavi, J.M. Doyle, W.H. Sanders, and P.G. Webster, "The Moebius framework and its implementation," *IEEE Trans. on Soft. Eng.*, vol. 28, no. 10, pp. 956–969, Oct. 2002.

[18] S. Greiner and G. Bolch, "Modeling production lines with blocking, batch processing and unreliable machines using the Markov analyzer MOSES," in *Proc. European Simulation Symp. - ESS '95*, 1995, pp. 303–309.

[19] K. Begain, G. Bolch, and H. Herold, *Practical Performance Modeling – Application of the MOSEL language*, Kluwer Academic Publishers, 2001.

[20] B. Beutel, "Integration of the Petri Net tool TimeNET into the MOSEL modelling environment," M.S. thesis, Department of Computer Science, University of Erlangen, Germany, 2003.

[21] P. Wuechner, "Extending the interface between the modeling languages MOSEL and CSPL by adding simulation constructs," Semester Thesis SA-I4-2003-06, Department of Computer Science, University of Erlangen, Germany, 2003.

[22] P. Wuechner, "Performance modelling of mobile networks using MOSEL-2," M.S. thesis, Department of Computer Science, University of Erlangen, Germany, 2004.

[23] K. Begain, "Using subclasses of PH distributions for the modeling of empirical distributions," in *Proc. 18th ICSCS*, 1993, pp. 141–152.