Stochastic simulation and modelling

Augustin PRODAN and Rodica PRODAN
Iuliu Hațieganu University
Str. Emil Isac 13, 3400 Cluj-Napoca, Romania
e-mail:<aprodan@umfcluj.ro>

Abstract: The paper demonstrates the advantages of stochastic models for representation of real world activities. The Department of Mathematics and Informatics from Iuliu Hațieganu University Cluj-Napoca has created and implemented a collection of Java class libraries for stochastic simulation and modelling. The paper reports on the incremental development of an object-oriented Java framework, based on theoretical fundamentals in simulation and stochastic modelling, that supports the creation of the main elements for building and implementing stochastic models. The stochastic models constructed accurately represent real world phenomena and processes particularly in health care and patient monitoring. One application modelled the patient flow through chronic diseases departments. The model uses a Poisson process with parameter λ estimated by using the inter-arrival times. The model covers in-patient care time as a mixed-exponential phase-type distribution. Geriatricians and hospital administrators both agreed that such a model can be applied to optimise the use of hospital resources in order to improve hospital care.

Keywords: distributional model, exponential distribution, stochastic process, simulation, polymorphic method.

1. Introduction

Previous research has shown that stochastic models are advantageous tools for representation of the real world. Due to actual spread of fast and inexpensive computational power everywhere in the world, the best approach is to model a real phenomenon as faithfully as possible, and then rely on a simulation study to analyse it. Based on theoretical fundamentals in stochastic modelling and simulation defined by Ross (1990), an incremental development of an object-oriented Java framework containing the main elements for building and implementing stochastic models is realized in the Department of Mathematics and Informatics of Iuliu Hațieganu University, Cluj-Napoca. The main result of this research is a collection of Java class libraries, which are used to model and to simulate classical distributions, stochastic processes and Monte Carlo methods. The basic design philosophy of our object-oriented approach to simulation of the random variables by means of classical distributions is presented in Prodan et al. (1999). The object-oriented Java framework containing the set of base-line classes for stochastic modelling and simulation is presented in Prodan et al. (2000). This framework is used to create stochastic models, which accurately represent real world phenomena and processes, particularly in health care and patient monitoring. We apply this framework to study the flow of patients around departments of geriatric medicine. Geriatric departments generally consist into two main different compartments: acute care and long-stay care. Patients are initially admitted into acute care consisting of diagnosis, assessment and rehabilitation. The majority of patients is either released and therefore re-enters the community or die following such a period of acute care. A certain number, however, may be considered to be unable to look after themselves, and therefore pass from acute into long-stay care where they may remain for a considerable amount of time or they will eventually die. A two-stage discrete-time deterministic model to describe such movements of patients has been developed by Harrison and Millard (1991) and shown to explain the empirical result that the distribution of length of stay of patients in a geriatric department is described by a mixed-exponential distribution. An extension of this model to its continuous-time stochastic analogue is to be found in Irvine and McClean (1994).

2. Levels of simulation

There are three levels of simulation to be considered (see Fig.1). The *first level* consists of simulating *random numbers*, as they are the basis of any stochastic simulation study. The so-called *pseudo-random numbers* generated via Java's built in linear congruential generator are used to simulate random numbers. If Java's simple linear congruential generator is non-random in certain situations by generating undesirable regularities, it is necessary to build a class as an improvement over the one supplied by Java [1, 8].

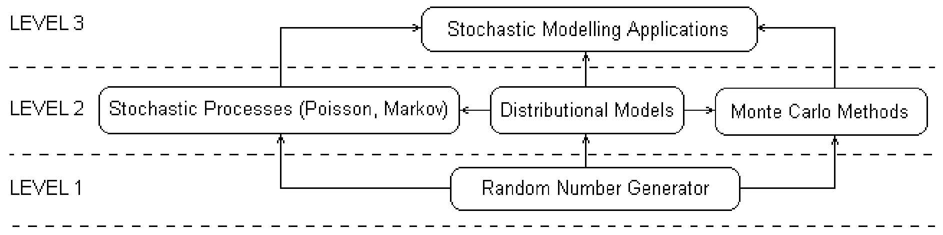

Figure 1. The simulation levels

Based on the first level, the *second level* of simulation applied for *distributional models* is built. A hierarchy of Java classes for modelling classical distributions is implemented. Each distribution is determined by a distribution function and a set of parameters. Based on these elements, a specific simulation algorithm is implemented for each distribution via a polymorphic method, each distribution class encapsulating a specific method, which generates a specific value. Simulations for *stochastic processes*, and their use in the context of queuing systems, are implemented at this level. First we considered *Poisson processes* and we have some applications where Poisson processes are simulated. Then we began to implement simulations for *renewal processes* and *Markov processes*. Also, methods for the *Monte Carlo* approach to solving some problems (approximating simple or multiple integrals, finding a solution for an equation system, finding the reverse, the eigenvectors and the eigenvalues for a matrix) are implemented at this level.

The *third level* of simulation is devoted to applications. As an application, in section 4 we modelled the patient flow through chronic diseases departments. Admissions are modelled as a Poisson process with parameter λ (the arrival rate) estimated by using the observed inter-arrival times. The in-patient care time is modelled as a mixed-exponential phase-type distribution.

3. Distributional models, stochastic processes and Monte Carlo methods

The classical random variables are the simplest stochastic models, also called *distributional models*, which enter into the composition of other complex models. A hierarchy of Java classes for modelling the classical distributions is proposed (see Fig. 2). Each distribution is determined by a set parameters and a distribution function. Based on these elements a Java class is defined for each distribution. This Java class is used to create objects, that is instances with particular values for parameters. Also, a simulation algorithm is defined for each class, which is able to generate a value for corresponding distribution. So, there is a set of simulation algorithms belonging to the whole hierarchy of Java classes, these algorithms being implemented via a polymorphic method called simValue(). This way, a particular simValue() must be specified by each distribution class in turn, in order to simulate a specific value. An instance of a particular class can be used to simulate a sequence of values for the corresponding random variable, by calling simValue() as many times as needed. The particular implementation of the simulation algorithm for each class is based on one or more of the following techniques: the *Inverse Transform Technique*, the *Acceptance-Rejection Technique* and the *Composition Technique* [6].

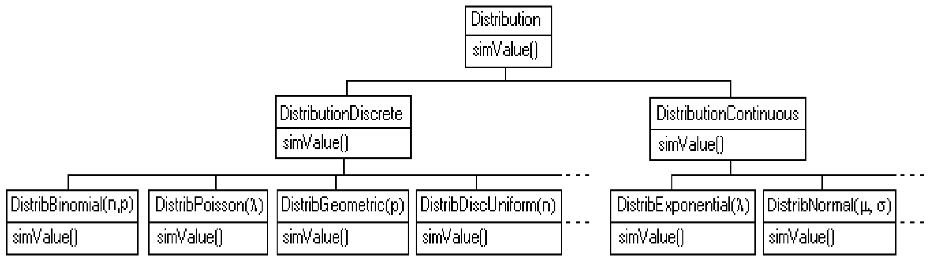

Figure 2. The hierarchy of Java classes for distributional models

The basic Java constructors for discrete and continuous distribution classes are given in Tab. 1 and Tab. 2, respectively.

Table 1. The discrete distribution classes

DISTRIBUTION	PUBLIC CLASS CONSTRUCTOR	PROBABILITY MASS FUNCTION
Binomial	DistribBinomial(int n, double p)	$p_i = C_n^i p^i (1-p)^{n-i}$, $i = 0, 1, \ldots, n$
Poisson	DistribPoisson(double lambda)	$p_i = e^{-\lambda}\lambda^i/i!$, $i = 0, 1, 2, \ldots$
Discrete uniform	DistribDiscUniform(int n)	$p_i = 1/n$, $i = 0, 1, \ldots, n-1$
Discrete general	DistribDiscrete(int n, double[ ] p)	$p_i = p[i]$, $i = 0, 1, \ldots, n-1$
Geometric	DistribGeometric(double p)	$p_i = p(1-p)^{i-1}$, $i \geq 1$

Table 2. The continuous distribution classes

DISTRIBUTION	PUBLIC CLASS CONSTRUCTOR	PROBABILITY DENSITY FUNCTION
Exponential	DistribExponential(double lambda)	$f(x) = \lambda e^{-\lambda x}, x > 0$
Gamma	DistribGamma(double lambda, int n)	$f(x) = \lambda e^{-\lambda x}(\lambda x)^{n-1}/(n-1)!, x > 0$
Normal standard	DistribNormalS()	$f(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2}, x \in R$
Normal	DistribNormal(double miu, double sigma)	$f(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{-(x-\mu)^2/(2\sigma^2)}, x \in R$
Weibull	DistribWeibull(double alfa, double beta)	$f(x) = \alpha\beta(\alpha\beta)^{\beta-1} e^{-(\alpha x)^{\beta}}, x \ge 0$

Simulations for *stochastic processes* are implemented at this level. A stochastic process is a family of random variables $\{X(t), t \in T\}$, where $X(t)$ describes the law of probability at time $t \in T$. First we approached the so-called *counting processes* $\{N(t), t \ge 0\}$, where $N(t)$ denotes the number of events occurred by time t. We implemented a simple category of counting processes, namely the *Poisson processes*. To generate a Poisson process, means in fact to generate the first n event times, or to generate the inter-arrival times. Making use of the fact that the times between successive events are independent exponential random variables (each with parameter λ, the rate of the Poisson process), a Poisson process is implemented simply based on Inverse Transform Technique used to implement an exponential random variable. To generate the first n inter-arrival times of a Poisson process, it is enough to generate n random numbers $U_1, U_2, ..., U_n$, and then to set $X_i = -(1/\lambda)\ln(U_i)$, $i = 1,2,...,n$. Facilities for generating arrival times (or waiting times) for the first n events are implemented. The arrival time of the k-th event is a Gamma distribution with parameters k and λ, the expected value being k/λ. Also, *Monte Carlo methods* are implemented at this level. We approached a lot of specific problems, such as approximating simple or multiple integrals, finding a solution for an equation system, finding the reverse, the eigenvectors and eigenvalues for a matrix, etc.

4. An application for patient flow simulation

The planning of medical service within a chronic healthcare department is a complex problem the staff has to face, because patients of long-term services occupy the beds for long periods of time and a high quality medical care costs a lot of money. Under these circumstances, a balanced policy between a high quality service measured by the number of beds and suitable costs becomes a necessity for the administration in order to get full value for the money they have spent. With this end in view, using the simulation of distributional models and stochastic processes, we intend to model the patient flow through chronic diseases departments. The use of stochastic compartmental analysis [10], which assumes probabilistic behaviour of the patients around the system, is considered a more realistic representation of an actual situation rather than simpler deterministic model. In order to simulate the model, we have split it into two parts: the arrival of patients and the in-patient care. Patients are initially admitted into acute care consisting of diagnosis, assessment and rehabilitation. The majority of patients is either released and therefore re-enters the community or die following such a period of acute care. However, a certain number of patients may be considered to be unable to look after themselves, and therefore pass from acute into long-stay care where they may remain for a considerable amount of time, or they will eventually die.

The arrival of patients is modelled as a Poisson process with a parameter λ estimated by using the inter-arrival times [6]. These times are independent exponential random variables, each with parameter λ and with the corresponding density $f(t) = \lambda e^{-\lambda t}$, $t \geq 0$. To simulate the first T time units of a Poisson process, it is necessary to simulate the successive inter-arrival times, stopping when their sum exceeds T. The results of a simulation, considering the arrival of patients as a Poisson process at rate $\lambda = 2.75$ patients per day, are presented in Fig. 3.

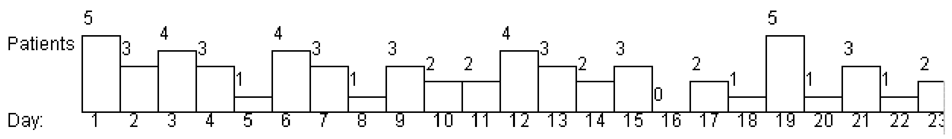

Figure 3. The Poisson arrivals at rate λ = 2.75 patients per day.

The care time is modelled by the application of a mixed-exponential distribution, where the number of terms in the mixture corresponds to the number of stages of patient care. A common scenario is that there are two stages for in-patient care: *acute* and *long-stay*, composing in this case two exponential distributions with parameters λ_1 and λ_2, representing the corresponding access rate for each stage. In this case, the mixed-exponential phase-type distribution [6, 7] has the probability density function $f(x) = \rho\lambda_1 e^{-\lambda_1 t} + (1-\rho)\lambda_2 e^{-\lambda_2 t}$, which implies a mean care time of $\rho/\lambda_1 + (1-\rho)/\lambda_2$ days per patient. Fig. 4 shows the results of a simulation with parameters $\rho = 0.07$, $\lambda_1 = 1/77.18$ and $\lambda_2 = 1/33.3$.

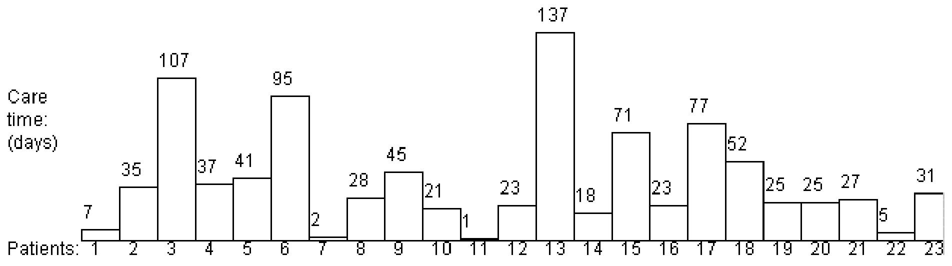

Figure 4. The simulated results for in-patient care time

5. Future work

As a future work, we intend to approach and implement models for stochastic processes (Markov, semi-Markov, renewal), with applications in medicine and pharmacy. A continuous-time Markov model for the flow of patients around compartments of geriatric medicine will be proposed. This model will enable us to study the whole system of geriatric care and will be used to look either at the time patients spend in hospital and also the subsequent time patients spend in the community.

Also, we intend to implement the so-called *bootstrap technique*, which is quite useful in analysing a simulation. One may bootstrap the observed data (randomly generate new plausible data based on the observed ones) to create a pseudo-dataset, and then fit the model to that pseudo-data to obtain a set of parameter pseudo-estimates that provide another estimate of that individual parameter values. The latter is expected to provide predictions that are as plausible as those obtained from the original experimental fit. Such predictions can, upon repetition of the process, reflect the sampling variability that is believed to be inherent in the original data. Bootstrapping strategies will be applied in pharmaceutical research, education and industry with the purpose to optimise the use of substances and reactants.

6. Conclusions

A Java framework containing the base line classes for distributional models was introduced. This framework is used to create stochastic models, which accurately represent real world phenomena and processes, particularly in health care and patient monitoring. We modelled the patient flow trough chronic diseases departments. Admissions are modelled as a Poisson process with parameter λ estimated by using the inter-arrival times. The in-patient care time is modelled as a mixed-exponential phase-type distribution. Both geriatricians and hospital administrators agreed that such a model is useful to be applied for optimising the use of hospital resources in order to improve hospital care. As a future work we intend to approach and implement new models for stochastic processes, with applications in medicine and pharmacy. Also, we intend to implement the new *bootstrap technique*, which is very useful in analysing a simulation.

References

[1] ECKEL, B. (1998): *Thinking in Java*, MindView Inc., Prentice Hall PTR.

[2] CÂMPEAN, R. and PRODAN, A. (2000): *A Java Environment for Stochastic Simulation,* Clujul Medical Vol. LXXIII, No. 2, 312-317.

[3] GORUNESCU, F., PRODAN, A. and GORUNESCU, M. (2001): *Optimising the Chronic Healthcare Department Occupancy Using the Queuing Modelling*, Clujul Medical Vol. LXXIII, No. 3, (to appear).

[4] HARRISON, G. W. and MILLARD, P. H. (1991): *Balancing Acute and Long-stay Care: The Mathematics of Throughput in Departments of Geriatric Medicine*, Methods of Information in Medicine 30, 221-228.

[5] IRVINE, V., MCCLEAN, S. and MILLARD P. (1994): *Stochastic Models for Geriatric In-Patient Behaviuor*, IMA J. Math. Appl. Medicine and Biology 11, 207-216.

[6] PRODAN, A., GORUNESCU, F. and PRODAN, R. (1999): *Simulating and Modelling in Java*, Proceedings of the Workshop on Parallel / High-Performance Object-Oriented Scientific Computing, June 1999, Lisbon, Technical Report FZJ-ZAM-IB-9906, Forschungszentrum Jűlich GmbH, 55-64.

[7] PRODAN, A., GORUNESCU, F., PRODAN, R. and CÂMPEAN, R. (2000): *A Java Framework for Stochastic Modelling*, Proceedings of the 16th IMACS World Congress 2000 on Scientific Computation, Applied Mathematics and Simulation, 21-25 August 2000, Lausanne, Switzerland.

[8] PRODAN, A. and PRODAN, M. (1997): *Java Environment for Internet*, Ed. ProMedia-plus, Cluj-Napoca, ISBN 973-9275-07-9.

[9] ROSS, S. M. (1990): *A Course in Simulation*, Macmillan Publishing Company, New York.

[10] TAYLOR, G. J., MCCLEAN, S. and MILLARD, P. (1998): *Continuous-time Markov Models for Geriatric Patient Behaviuor*, Appl. Stochastic Models and Data Analysis 13, 315-323.