PLISKA
STUDIA MATHEMATICA
BULGARICA

ПЛИСКА
БЪЛГАРСКИ
МАТЕМАТИЧЕСКИ
СТУДИИ

For further information on
Pliska Studia Mathematica Bulgarica
visit the website of the journal http://www.math.bas.bg/~pliska/
or contact: Editorial Office
Pliska Studia Mathematica Bulgarica
Institute of Mathematics and Informatics
Bulgarian Academy of Sciences
Telephone: (+359-2)9792818, FAX:(+359-2)971-36-49
e-mail: pliska@math.bas.bg

Pliska Stud. Math. Bulgar. **13** (2000), 117-132

PLISKA
STUDIA MATHEMATICA
BULGARICA

MONTE CARLO ALGORITHM FOR SOLVING INTEGRAL EQUATIONS WITH POLYNOMIAL NON-LINEARITY. PARALLEL IMPLEMENTATION *

Ivan T. Dimov, Todor V. Gurov

An iterative Monte Carlo algorithm for evaluating linear functionals of the solution of integral equations with polynomial non-linearity is proposed and studied. The method uses a simulation of branching stochastic processes. It is proved that the mathematical expectation of the introduced random variable is equal to a linear functional of the solution. The algorithm uses the so-called *almost optimal density function.*

Numerical examples are considered. Parallel implementation of the algorithm is also realized using the package ATHAPASCAN as an environment for parallel realization. The computational results demonstrate high parallel efficiency of the presented algorithm and give a good solution when almost optimal density function is used as a transition density.

1 Introduction

The paper deals with realizations of Monte Carlo algorithms for evaluating of integral equations. It is known that these algorithms usually can be applied for calculating functionals of the solution. Consider the following functional:

$$J(u) \equiv (g,u) = \int_G g(x)u(x)dx, \tag{1}$$

where the domain $G \subset \mathbb{R}^d$ and a point $x \equiv (x_1, x_2, \ldots, x_d) \in G$ is a point in the Euclidean space $\mathbb{R}^d$. The functions $u(x)$ and $g(x)$ belong to any Banach space X and to the adjoint space X^*, respectively, and $u(x)$ is an unique solution of the following Fredholm integral equation in an operator form:

*Supported by the Ministry of Education, Science and Technology of Bulgaria under Grants # MM 449/94 and # I 501/95 as well as by EC under INCO-Copernicus Project #960237 - STABLE.

$$u = I\!K(u) + f. \tag{2}$$

The problem under consideration can be formulated as estimation the functional $J(u)$. The case when $g = \delta(x - x_0)$ is a delta - function is of special interest, because we are interested in calculating the value of u at x_0, where $x_0 \in G$ is a fixed point.

It is known, that Monte Carlo methods give statistical estimates for the solution of the problem by performing random sampling of a certain chance variable whose mathematical expectation is the desired solution [8], [11].

Moreover, the same algorithmic complexity is needed for estimating any linear functional of the solution (for example, an inner product of a given function with the solution of the Fredholm integral equation of the second kind) [2], [3].

Consider a general description of the Monte Carlo method. Let $f = f(x) \in X$ and $u_l = u(x_l) \in X$ be defined in $\mathbb{R}^d$ and $I\!K = I\!K(u)$ be a linear operator $I\!K(u) \in [X \to X]$.

We consider the *first order stationary linear iterative process* for the integral equation (2)

$$u_l = I\!K(u_{l-1}) + f, \quad l = 1, 2, \ldots, \tag{3}$$

where l is the number of iterations. In fact (3) defines a *Neumann series*

$$u_l = f + I\!K(f) + \ldots + I\!K^{l-1}(f) + I\!K^l(u_0), \qquad l > 1, \tag{4}$$

where $I\!K^l$ means the l-th iteration of $I\!K$.

A special case is when the corresponding infinite series converges. Then the sum is an element u from the space X which satisfies the equation (2).

From (2) and (4) one can get the value of the truncation error. If $u_0 = f$, then

$$u_l - u = I\!K^l(f - u). \tag{5}$$

Consider the spaces

$$T_{i+1} = \underbrace{\mathbb{R}^d \times \mathbb{R}^d \times \ldots \times \mathbb{R}^d}_{i \text{ times}}, \qquad i = 1, 2, \ldots, l, \tag{6}$$

where "$\times$" denotes Cartesian product of spaces and let $J(u_l)$ be a linear functional that is to be calculated.

Random variables Θ_i, $i = 0, 1, \ldots, l$ are defined on the respective product spaces T_{i+1} and have conditional mathematical expectation:

$$E\Theta_0 = J(u_0), \; E(\Theta_1/\Theta_0) = J(u_1), \ldots, E(\Theta_l/\Theta_0) = J(u_l). \tag{7}$$

The computational problem then becomes one of calculating repeated realizations of Θ_l and combining them into an appropriate statistical estimator for $J(u_l)$. As an approximate value of the linear functional $J(u_l)$ set up

$$J(u_l) \approx \frac{1}{n}\sum_{i=1}^{n}\{\Theta_l\}_i = \hat{\Theta}_{n_l}, \tag{8}$$

where $\{\Theta_l\}_i$ is the i-th realization of the random variable Θ_l.

The probable error is defined as a value r_n (see [5], [11]) for which the following condition holds:

$$Pr\{|J - \hat{\Theta}_{n_l}| \leq r_n\} \approx \frac{1}{2} \approx Pr\{|J - \hat{\Theta}_{n_l}| > r_n\}.$$

For the usual Monte Carlo method (which does not use any a priori information about the smoothness of the functions)

$$r_n = c\sigma(\Theta_l)n^{-\frac{1}{2}}, \tag{9}$$

where $c \approx 0.6745$; $\sigma(\Theta_l)$ is the standard deviation of the random variable Θ_l; n is the number of trials. Our approach corresponds to a special case of the operator $\mathbb{K}$:

$$\mathbb{K}(u^{(m)}) = \int_G \dots \int_G k(x, y_1, \dots, y_m) \prod_{i=1}^{m} u(y_i) \prod_{i=1}^{m} dy_i,$$

which leads to a *non-stationary iterative process.*

2 Formulation of the Problem

Let us take $X = L_1(G)$ then $X^* = L_\infty(G)$ ([5], p.148). Consider the following special case of the operator:

$$\mathbb{K}(u^{(m)}) = \int_G \dots \int_G k(x, y_1, y_2, \dots, y_m) \prod_{i=1}^{m} u(y_i) \prod_{i=1}^{m} dy_i \quad m \geq 2, \tag{10}$$

where $\mathbb{K}$ is an integral transform with polynomial nonlinearity.

Thus, equation (2) becomes :

$$u(x) = \int_G \dots \int_G k(x, y_1, y_2, \dots, y_m) \prod_{i=1}^{m} u(y_i) \prod_{i=1}^{m} dy_i + f(x), \quad m \geq 2. \tag{11}$$

In this case the problem under consideration can be formulated as follows: for a given $k(x, y_1, y_2, \dots, y_m) \in L_1 \underbrace{(G \times \dots \times G)}_{m+1}$ and $f(x) \in L_1(G)$, compute the functional (1), where the function $u(x)$ is a solution of an integral equation with polynomial non-linearity (11).

Solving the integral equation (11) by the method of simple iterations one can obtain the iterative process:

$$(12)\quad u_l(x) = \int_G \dots \int_G k(x, y_1, y_2, \dots, y_m) \prod_{i=1}^{m} u_{l-1}(y_i) \prod_{i=1}^{m} dy_i + f(x) \quad l = 1, 2, \dots,$$

$$u_0(x) = f(x),$$

or

$$u_l = I\!K(u_{l-1}^{(m)}) + f, \; u_{l-1}^{(m)} = \prod_{i=1}^{m} u_{l-1}(y_i), \quad u_0 = f.$$

Consider the equation:

$$(13)\qquad u(x) = \int_G \dots \int_G |k(x, y_1, y_2, \dots, y_m)| \prod_{i=1}^{m} u(y_i) \prod_{i=1}^{m} dy_i + |f(x)|, \quad m \geq 2.$$

It is clear that if the following iterative process

$$(14) u_l(x) = \int_G \dots \int_G |k(x, y_1, y_2, \dots, y_m)| \prod_{i=1}^{m} u_{l-1}(y_i) \prod_{i=1}^{m} dy_i + |f(x)|, \quad l = 1, 2, \dots,$$

$$u_0(x) = |f(x)|$$

converges then the iterative process (12) converges.

Suppose, the conditions providing existence of the unique solution of the problem under consideration are fulfilled. Let us note (as an example) that for the existing of an unique solution of the equation

$$u(x) = \int_G k(x, y) u^m(y) dy + f(x), \quad m \geq 2$$

the following condition

$$(15)\qquad K_m < \min\left(\frac{D - F}{D^m}, \frac{1}{mD^{m-1}}\right)$$

has to be fulfilled. In the last expression the following notations are used:

$$K_m = \max_{x \in G} \int_G |k(x, y)| dy, \qquad F = \max_{x \in G} |f(x)|$$

and the constant D satisfies the inequalities:

$$D > F, \quad \| u_0 \| \leq D.$$

The problem of constructing of a random variable which mathematical expectation is equal to the functional (1) is considered in the next section. A branching stochastic process [5] which corresponds to the above iterative process is used. The above mentioned iterative process is non-stationary.

3 The Monte Carlo Method

In this section we consider an approach for constructing iterative Monte Carlo algorithms for evaluating of functional (1), where $u(x)$ is the solution of the non-linear integral equation of Fredholm type (11).

Suppose that any particle distributed with initial density function

$$p_0(x) \geq 0, \text{ and } \int p_0(x)dx = 1$$

is born in the domain $G \in \mathbb{R}^d$ in a random point x_0.

In the next moment this particle either dies out with probability $h(x_0)$, where $(0 \leq h(x_0) < 1)$ or generates posterity of m, $(m \geq 2)$ analogical particles in the next random points $x_{00}, x_{01}, \ldots, x_{0m-1}$ with probability

$$p_m(x_0) = 1 - h(x_0)$$

and transition density function

$$p(x_0, x_{00}, \ldots, x_{0m-1}) \geq 0,$$

where

$$\int \ldots \int p(x_0, x_{00}, x_{01}, \ldots, x_{0m-1}) \prod_{i=0}^{m-1} dx_{0i} = 1.$$

The generated particles behave in the next moment as the initial one and etc. The traces of such a process is a tree from the type sketched in Figure 1.

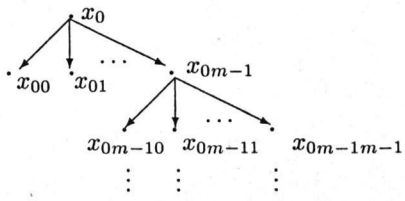


Figure 1

The used index numbers in Figure 1 are called *multi-indices*. The particle which belongs to zero generation is enumerated with zero index, i.e. x_0. Its direct inheritors are enumerated with the indices $00, 01, \ldots, 0m-1$, i.e. the next points $x_{00}, x_{01}, \ldots, x_{0m-1}$ are first generation. If a particle (or a point) from the q-th generation of the tree γ has

the multi-index $\gamma[q]$, then the multi-index of the s-th inheritor of this particle has the following form $\gamma[q+1] = \gamma[q]s$, where in this case the corresponding multi-index is a number written in m-th numerical system.

Consider the first two iterations of the iterative process (12), ([5], p. 254) in the simple case when $m = 2$. The branching stochastic process which corresponds to these two iterations is presented on Figure 2.

$$u_0(x_0) = f(x_0),$$

$$u_1(x_0) = f(x_0) + \iint k(x_0, x_{00}, x_{01}) f(x_{00}) f(x_{01}) dx_{00} dx_{01},$$

$$u_2(x_0) = f(x_0) + \iint k(x_0, x_{00}, x_{01}) f(x_{00}) f(x_{01}) dx_{00} dx_{01} + \tag{16}$$

$$+\iint k(x_0, x_{00}, x_{01}) f(x_{01}) \left(\iint k(x_{00}, x_{000}, x_{001}) f(x_{000}) f(x_{001}) dx_{000} dx_{001} \right) dx_{00} dx_{01} +$$

$$+\iint k(x_0, x_{00}, x_{01}) f(x_{00}) \left(\iint k(x_{01}, x_{010}, x_{011}) f(x_{010}) f(x_{011}) dx_{010} dx_{011} \right) dx_{00} dx_{01} +$$

$$+ \iint k(x_0, x_{00}, x_{01}) \left(\iint k(x_{01}, x_{010}, x_{011}) f(x_{010}) f(x_{011}) dx_{010} dx_{011} \right) \times$$

$$\times \left(\iint k(x_{00}, x_{000}, x_{001}) f(x_{000}) f(x_{001}) dx_{000} dx_{001} \right) dx_{00} dx_{01}.$$

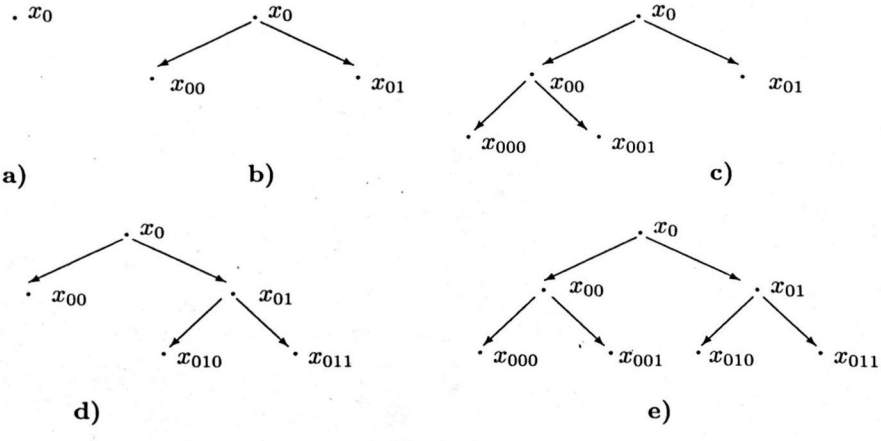


Figure 2

Obviously the structure of u_1 is linked with all trees which appear untill the first generation, (see Figure 2a and Figure 2b). The structure of u_2 is linked with all trees which appear untill the second generation.

Definition 3.1 *A full tree with l generations is called the tree Γ_l where the dying out of particles is not visible from zero to the $l-1$-st generation but all the generated particles of the l-th generation die out.*

Example: In fact Γ_2 is the tree γ_0 from Figure 2e and Γ_1 is the tree from Figure 2b. The next density function corresponds to the tree γ_0 from Figure 2e:

$$p_{\gamma_0} = p_0(x_0)p_2(x_0)p(x_0,x_{00},x_{01})p_2(x_{00})p(x_{00},x_{000},x_{001})p_2(x_{01})p(x_{01},x_{010},x_{011})\times$$

$$\times h(x_{000})h(x_{001})h(x_{010})h(x_{011}). \tag{17}$$

The random variable which corresponds to γ_0 is:

$$\Theta_{[g]}(\gamma_0) = \frac{g(x_0)}{p_0(x_0)}\frac{k(x_0,x_{00},x_{01})}{p_2(x_0)p(x_0,x_{00},x_{01})}\frac{k(x_{00},x_{000},x_{001})}{p_2(x_{00})p(x_{00},x_{000},x_{001})}\times \tag{18}$$

$$\times\frac{k(x_{01},x_{010},x_{011})}{p_2(x_{01})p(x_{01},x_{010},x_{011})}\frac{f(x_{000})f(x_{001})f(x_{010})f(x_{011})}{h(x_{000})h(x_{000})h(x_{000})h(x_{000})}.$$

This random variable estimates the last of the terms in (16) if the realization of the random process is a tree of kind γ_0. Thus, random variables are constructed which correspond to trees of another type.

Denote by A the set of points, generating new points and denote by B the set of points, which die out [7].

Consider a branching stochastic process with l generations in the general case $m \geq 2$. It corresponds to the truncated iterative process with l iterations (12).

There is a one-to-one correspondence between the number of the subtrees of the full tree Γ_l and the number of the terms of the truncated iterative process with l iterations (see, for example, [5]).

This correspondence allows to construct a procedure for a random choice of the tree and to calculate the values of a random variable which corresponds to this tree. Thus, when we construct the branching stochastic process we shall receive arbitrary trees γ. In this case we can associate to every one of them the random variable $\Theta_{[g]}(\gamma/\Gamma_l)$ in the following way:

$$\Theta_{[g]}(\gamma/\Gamma_l)) = \frac{g(x_0)}{p_0(x_0)}\frac{f(x_0)}{h(x_0)}, \tag{19}$$

if the tree consists of the initial point only.

If the tree consists of other points, then $\Theta_{[g]}(\gamma)/\Gamma_l)$ is constructed simultaneously with the construction of γ. When a transition from the random point $x_{\gamma[q]}$ to points

$$x_{\gamma[q]0}, x_{\gamma[q]1}, \ldots, x_{\gamma[q]m-1}$$

take place then it corresponds to multiplication by

$$\frac{k(x_{\gamma[q]}, x_{\gamma[q]0}, \ldots, x_{\gamma[q]m-1})}{p_m(x_{\gamma[q]})p(x_{\gamma[q]}, x_{\gamma[q]0}, \ldots, x_{\gamma[q]m-1})}.$$

When the point $x_{\gamma[r]}$ dies out it corresponds to multiplying by

$$\frac{f(x_{\gamma[r]})}{h(x_{\gamma[r]})}.$$

In this case the random variable $\Theta_{[g]}(\gamma/\Gamma_l)$ which corresponds to γ is:

$$\Theta_{[g]}(\gamma/\Gamma_l)) = \frac{g(x_0)}{p_0(x_0)} \prod_{x_{\gamma[q]} \in A} \frac{K(x_{\gamma[q]})}{P(x_{\gamma[q]})} \prod_{x_{\gamma[r]} \in B} \frac{f(x_{\gamma[r]})}{h(x_{\gamma[r]})} \tag{20}$$

with the density function

$$p_\gamma = p_0(x_0) \prod_{x_{\gamma[q]} \in A} P(x_{\gamma[q]}) \prod_{x_{\gamma[r]} \in B} h(x_{\gamma[r]}), \tag{21}$$

$$1 \le q \le l-1, \quad 1 < r \le l,$$

where

$$K(x_{\gamma[q]}) = k(x_{\gamma[q]}, x_{\gamma[q]0}, \ldots, x_{\gamma[q]m-1})$$

and

$$P(x_{\gamma[q]}) = p_m(x_{\gamma[q]})p(x_{\gamma[q]}, x_{\gamma[q]0}, \ldots, x_{\gamma[q]m-1}).$$

Thus, we obtain random variable for arbitrary trees γ which estimate l-th iteration, u_l, of the iterative process (12).

Theorem 3.1. *The mathematical expectation of the random variable* $\Theta_{[g]}(\gamma/\Gamma_l)$ *is equal to the functional* $J(u_l)$, *i.e.*

$$E\Theta_{[g]}(\gamma/\Gamma_l) = J(u_l) = (g, u_l).$$

Proof. Suppose that the subtree γ_i is fixed and it has $s_1 + s_2 + 1$ points, such that $x_0, x_1, \ldots, x_{s_1} \in A$ and $x_{s_1+1}, x_{s_1+2}, \ldots, x_{s_1+s_2} \in B$. It is clear that this can not be considered as a restriction of the generality.

Consider the following density function

$$p_{\gamma_i} = p_{\gamma_i}(x_0, x_1, \ldots, x_{s_1}, x_{s_1+1}, \ldots, x_{s_1+s_2}) = p_0(x_0) \prod_{i=0}^{s_1} P(x_i) \prod_{x=1}^{s_2} h(x_{s_1+i})$$

and the probability to obtain the subtree $\gamma = \gamma_i$

$$Pr\{\gamma = \gamma_i\} = \int_G \ldots \int_G p_{\gamma_i} dx_0 \ldots dx_{s_1+s_2}.$$

The random variable

$$\Theta_{[g]}(\gamma_i) = \frac{g(x_0)}{p_0(x_0)} \prod_{j=0}^{s_1} \frac{K(x_j)}{P(x_j)} \prod_{j=1}^{s_2} \frac{f(x_{s_1+j})}{h(x_{s_1+j})}$$

has the following condition density function

$$p_\gamma(x_0, \ldots, x_{s_1+s_2} | \gamma = \gamma_i) = \frac{p_{\gamma_i}(x_0, x_1, \ldots, x_{s_1}, x_{s_1+1}, \ldots, x_{s_1+s_2})}{Pr\{\gamma = \gamma_i\}},$$

where

$$K(x_j) = k(x_j, x_{j0}, \ldots, x_{jm-1}) \quad \text{and} \quad P(x_j) = p_m(x_j, x_{j0}, \ldots, x_{jm-1}).$$

Let us calculate the mathematical expectation:

$$E\Theta_{[g]}(\gamma_i) = E\{\Theta_{[g]}(\gamma/\Gamma_l) | \gamma = \gamma_i\} =$$

$$\int_G \ldots \int_G \Theta_{[g]}(\gamma_i) p_\gamma(x_0, \ldots, x_{s_1+s_2} | \gamma = \gamma_i) dx_0 \ldots dx_{s_1+s_2} =$$

$$\int_G \ldots \int_G g(x_0) \prod_{j=0}^{s_1} K(x_j) \prod_{j=1}^{s_2} f(x_{s_1+j}) \frac{dx_0 \ldots dx_{s_1+s_2}}{Pr\{\gamma = \gamma_i\}} =$$

$$= \frac{(g, u_l^i)}{Pr\{\gamma = \gamma_i\}}, \tag{22}$$

where

$$u_l^i = u_l^i(x_0) = \int_G \ldots \int_G \prod_{j=0}^{s_1} K(x_j) \prod_{j=1}^{s_2} f(x_{s_1+j}) dx_1 \ldots dx_{s_1+s_2}.$$

One can choose n subtrees of the full tree Γ_l. (Note that some of the subtrees may be chosen many times, since n is a large number.) n_{γ_i} is the number of the trees which correspond to γ_i. The following expression holds:

$$\frac{n_{\gamma_i}}{n} \approx Pr\{\gamma = \gamma_i\}.$$

On the other hand

$$J(u_l^i) = (g, u_l^i) \approx \frac{Pr\{\gamma = \gamma_i\}}{n_{\gamma_i}} \sum_{j=1}^{n_{\gamma_i}} (\Theta_{[g]}(\gamma_i))_j \approx \frac{1}{n} \sum_{j=1}^{n_{\gamma_i}} (\Theta_{[g]}(\gamma_i))_j. \tag{23}$$

Clearly

$$E\Theta_{[g]}(\gamma/\Gamma_l) = \sum_{i=0}^{N} E\{\Theta_{[g]}(\gamma/\Gamma_l)|\gamma = \gamma_i\} Pr\{\gamma = \gamma_i\},$$

where N is the number of all subtrees of the Γ_l.

Using (22) one can get

$$E\Theta_{[g]}(\gamma/\Gamma_l) = \sum_{i=0}^{N} (g, u_l^i) = (g, u_l) = J(u_l).$$

This completes the proof of the theorem.

Note, that if $l \to \infty$, then the mathematical expectation of the random variable is:

$$\lim_{l\to\infty} E\Theta_{[g]}(\gamma/\Gamma_l) = E\Theta_{[g]}(\gamma) = J(u) = \int_G g(x)u(x)dx, \tag{24}$$

where $u(x)$ is the solution of (12).

From (23) we obtain the following Monte Carlo method:

$$J(u_l) = \sum_{i=0}^{N} (g, u_l^i) \approx \frac{1}{n} \sum_{i=0}^{N} \sum_{j=1}^{n_{\gamma_i}} (\Theta_{[g]}(\gamma_i))_j = \frac{1}{n} \sum_{j=1}^{n} (\Theta_{[g]}(\gamma/\Gamma_l))_j. \tag{25}$$

Suppose the initial density function and the transition density function are *tolerant* to $g(x)$ and $k(x_{\gamma[q]}, x_{\gamma[q]0}, \ldots, x_{\gamma[q]m-1})$, respectively (the definition of the tolerant density functions is given in [11]).

In this case the probable error is

$$r_n \approx 0.6745\sigma(\Theta_{[g]}(\gamma/\Gamma_l))n^{-1/2},$$

where $\sigma(\Theta_{[g]}(\gamma/\Gamma_l))$ is the standard deviation of the random variable $\Theta_{[g]}(\gamma/\Gamma_l)$.

The problem of optimization of Monte Carlo algorithms consists in the minimization of the standard deviation, i.e. minimization of the second moment $E\Theta^2_{[g]}(\gamma/\Gamma_l)$ of the random variable $\Theta_{[g]}(\gamma/\Gamma_l)$. This is done by a suitable choice of the density function p_γ.

There are various techniques for variance reduction of the standard deviation. We consider one of them.

Let us chose the transition density function $p(x, y_1, y_2, \ldots, y_m)$ and the initial density function $p_0(x)$ in the following way:

$$p(x, y_1, y_2, \ldots, y_m) = C_x \mid k(x, y_1, y_2, \ldots, y_m) \mid \text{ and } p_0(x) = C_1 \mid g(x) \mid,$$

where

$$C_x^{-1} = \int_G \dots \int_G \mid k(x, y_1, y_2, \dots, y_m) \mid \prod_{i=1}^{m} dy_i \text{ and } C_1^{-1} = \int_G \mid g(x) \mid dx.$$

In this case the density function p_γ is called *almost optimal* [7] and one can write:

$$p_\gamma \equiv \overline{p}_\gamma =$$

$$(26) = C_1 \mid g(x_0) \mid \prod_{x_{\gamma[q]} \in A} p_m(x_{\gamma[q]}) C_{x_{\gamma[q]}} \mid k(x_{\gamma[q]}, x_{\gamma[q]0}, \dots, x_{\gamma[q]m-1}) \mid \prod_{x_{\gamma[r]} \in B} h(x_{\gamma[r]}).$$

Let us describe the Monte Carlo algorithm using the almost optimal density function. Calculate one value of the random variable $\Theta_{[g]}(\gamma/\Gamma_l)$ in the next steps:

Algorithm 3.1:

1. **Compute** *the point $\xi \in G$ which is a realization of the random variable τ with the density $p(x) = C_1 \mid g(x) \mid$. Here*

$$\Theta_{[g]}(\gamma/\Gamma_l) = C_1 \frac{g(\xi)}{\mid g(\xi) \mid}.$$

2. **Construct** *an independent realization, α, of the uniformly distributed random variable in the interval $[0, 1]$.*

3. **If** $(\alpha \leq p_m(\xi))$ **Then** *execute steps 5, 6 and 7.*
Else *execute step 4.*

4. **Multiply** *the value of the random variable $\Theta_{[g]}(\gamma/\Gamma_l)$ by*

$$\frac{f(\xi)}{h(\xi)}.$$

In this case we say that the point ξ dies out.

5. **Compute** *the points $\xi_1, \xi_2, \dots, \xi_m$ which are components of the m-th dimensional random variable $Y(y_1, y_2, \dots, y_m)$ with a transition density function*

$$p(x, y_1, y_2, \dots, y_m) = C_x \mid k(x, y_1, y_2, \dots, y_m) \mid .$$

6. **Multiply** *the value of the random variable $\Theta_{[g]}(\gamma/\Gamma_l)$ by*

$$\frac{k(\xi, \xi_1, \xi_2, \dots, \xi_m)}{p_m(\xi) C_\xi \mid k(\xi, \xi_1, \xi_2, \dots, \xi_m) \mid}.$$

7. **Repeat** *steps 2 and 3 for the generated points $\xi_1, \xi_2, \dots, \xi_m$.*

8. **Stop** *the algorithm when all points die out.*

Thus, we calculate one realization of the random variable $\Theta_{[g]}(\gamma/\Gamma_l)$. The above algorithm has to be executed n-times to obtain n realizations of the random variable $\Theta_{[g]}(\gamma/\Gamma_l)$.

This algorithm is very convenient for a parallel realization since every value of the random variable can be calculated independently and simultaneously.

Remark: Note that the above described algorithm can also be used for estimating the functional (1) when u is the solution of the equation:

$$u(x) = \sum_{m=1}^{s} \int_G \dots \int_G k(x, y_1, y_2, \dots, y_m) \prod_{i=1}^{m} u(y_i) \prod_{i=1}^{m} dy_i + f(x), \quad s \geq 2.$$

In this case the step 5 have to be modified such that m is an realization of a discrete random variable with s states.

4 Parallel Implementation and Numerical Results

It is well known that Monte Carlo algorithms are well suited for parallel architectures. In fact, if we consider the calculation of a trajectory as a single computational process, it is straightforward to regard the Monte Carlo algorithm as a collection of asynchronous processes evolving in parallel. Clearly, MIMD (multiple instruction, multiple data) - machines are the "natural" hardware platform for implementing such algorithms; it seems to be interesting to investigate on the feasibility of a parallel implementation on such type of machines. There are two main reasons:

1) since Monte Carlo methods are many times used from, within or in conjunction with more complex and large existing codes (usually written in FORTRAN, C), the easiness in programming makes the use of these machines very attractive;

2) the peak performance of every processor of these machines is usually not very good, but when a large number of processors is efficiently used a high general computational performance can be realized.

The MIMD computer used for our tests is a IBM SP1 with 32 processors. The environment for parallel programming is ATHAPASCAN which is developed by the research group in LMC/IMAG, Grenoble. ATHAPASCAN environment is developed using C-language and a special library for message passing which is similar to the well - known MPI-Message Passing Interface and PVM-Parallel Virtual Machine. ATHAPASCAN allows to distribute the computational problem on different type of processors or/and computers. This environment provides use of dynamic distribution of common resources and realizes a high level of parallel efficiency if the numerical algorithm is well parallelized. (For more details see ([9]).

Note that, in the case of an implementation on a sequential computer, all steps of the algorithm and all trajectories are executed iteratively, whereas on a parallel computer each trajectory can be carried concurrently.

Consider the almost optimal Monte Carlo algorithm to estimate the functional (1). The function $u(x)$ is a solution of the following integral equation with polynomial nonlinearity ($m = 2$):

$$u(x) = \int_G \int_G k(x,y,z)u(y)u(z)dydz + f(x), \tag{27}$$

where $x \in \mathbb{R}^1$ and $G = [0,1]$.

In our tests

$$k(x,y,z) = \frac{x(a_2y - z)^2}{a_1} \text{ and } f(x) = c - \frac{x}{a_3},$$

where $a_1 > 0$, $a_3 \neq 0$, a_2 and c are constants. Thus,

$$K_2 = \max_{x\in[0,1]} \int_0^1 \int_0^1 |k(x,y,z)|dydz = \frac{2a_2^2 - 3a_2 + 2}{6a_1}.$$

The equation (27) has an unique solution $u(x) = c$ when the following condition

$$c = \pm \left(\frac{6a_1}{a_3(2a_2^2 - 3a_2 + 2)} \right)^{1/2}$$

holds.

The transition density function which we use in our tests has the following form:

$$p(y,z) = \frac{6}{2a_2^2 - 3a_2 + 2}(a_2y - z)^2.$$

We consider the results for calculating the linear functional (1) for

$$a_1 = 11,\ a_2 = 4,\ a_3 = 12,\ c = 0.5,$$

and

$$g(x) = \delta(x - x_0).$$

Some results are plotted on Figures 3 - 5. Figure 3 shows the dependence of the approximated solution from the number of the random trees. Three different transition density functions are considered (0.25, 0.45, $x/4$). One can see that there are oscillations of the solution for a small numbers of random trees. The best solution is obtained for the third case ($p(x) = x/4$).

On Figure 4 the parallel efficiency of the algorithm is shown. The efficiency oscillate around $0.2 - 0.3$ when a small number $(200 - 300)$ of trees is realized on every of 16-th and 32-nd processors used in the system. The parallel efficiency grows up when the number of the random trees increase.

The Figure 5 shows how the parallel efficiency of the presented algorithm depends on the number of processors. In these numerical tests all random trees are distributed on all processors. That means that the system of 4 processors performs the same number of realizations as the system of 32 processors. As a result the system of 4 processors is more efficient (in general) than the system of 32 processors.

Figure 3: The exact solution and Monte Carlo solutions using the almost optimal density function. Three cases for the probability $p_2(x)$ which generate new points.

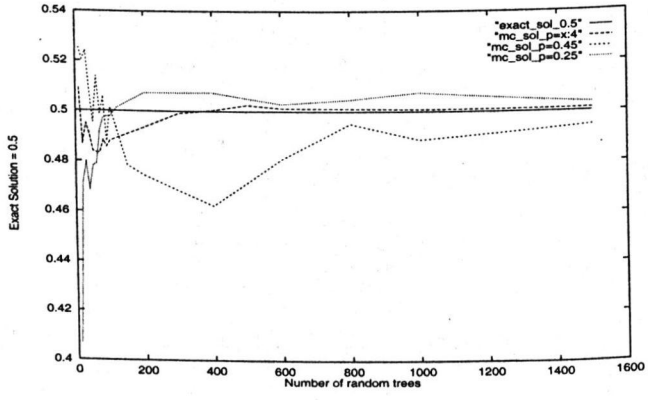


Figure 4: The parallel Monte Carlo efficiency when $p_2(x) = 0.25$. The number of processors is 16 and 32.

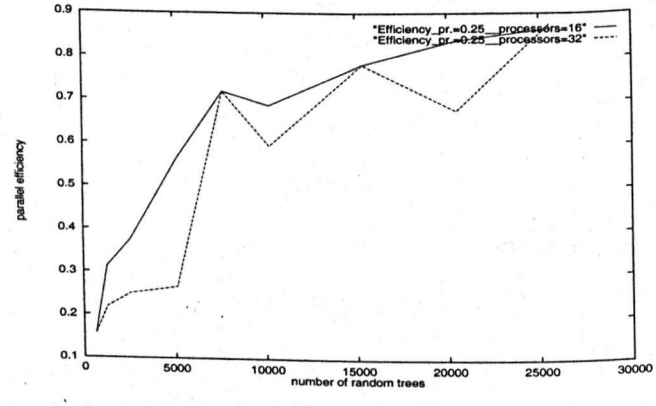
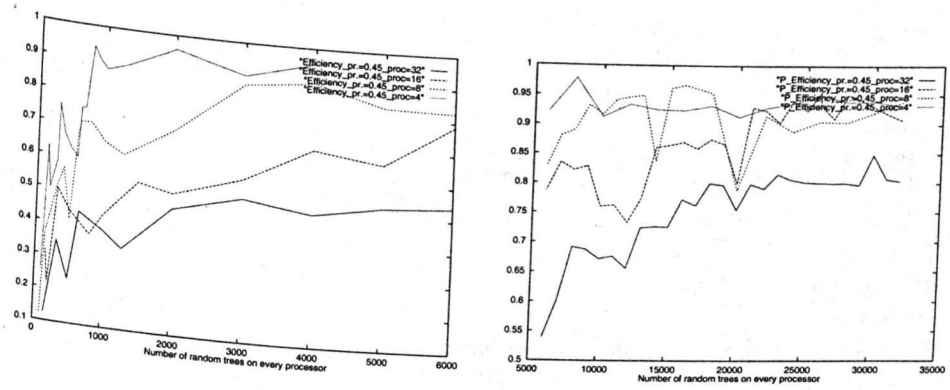


Figure 5: Parallel Monte Carlo efficiency for $p_2(x) = 0.45$. The number of processors is 4, 8, 16 and 32.



5 Conclusion

An almost optimal Monte Carlo algorithm is proposed and studied. The algorithm uses a simulation of branching stochastic processes.

It is proved that the mathematical expectation of the introduced random variable is equal to a linear functional of the solution.

The almost optimal Monte Carlo algorithms are very rapid. They give good results when one calculates the realizations of the random variable with an almost optimal density function directly.

Our tests show that Monte Carlo algorithms can be efficiency implemented on MIMD-machines.

REFERENCES

[1] I. T. Dimov. Minimization of the probable error for Monte Carlo methods. In: Application of Mathematics in Technology. Differential equations and applications, Varna 1986, Sofia 1987, 161-164.

[2] I. Dimov, O. Tonev. Performance Analysis of Monte Carlo Algorithms for Some Models of Computer Architectures. In: International Youth Workshop on Monte Carlo Methods and Parallel Algorithms - Primorsko (eds. Bl. Sendov, I. Dimov), World Scientific, Singapore, 1990, 91-95.

[3] I. Dimov, O. Tonev. Monte Carlo algorithms: performance analysis for some computer architectures. *J. of Comp. and Appl. Math.* **48** (1993), 253-277.

[4] S. M. Ermakov. On summation of series connected with integral equation. *Vestnik Leningrad Univ. Math.* **16** (1984), 57-63.

[5] S. M. Ermakov, G. A. Mikhailov. Statistical simulation. Moscow, Nauka, 1982.

[6] S. M. Ermakov, V. V. Nekrutkin, A. S. Sipin. Random processes for solving classical equations of the mathematical physics. Moscow, Nauka, 1984.

[7] T. Gurov. Monte Carlo Methods for Nonlinear Equations. Advances in Num. *Methods and Appl.*, World Scientific, 1994, 127-135.

[8] J. M. Hammersley, D. C. Handscomb. Monte Carlo methods. John Wiley & Sons inc., New York, London, Sydney, Methuen, 1964.

[9] B. Plateau. APACHE: Algorithmique Parallele et pArtagede CHargE, Raport APACHE, Institut IMAG, Grenoble, ♯1, pp. 28, 1994 .

[10] Yu. A. Shreider. Method of Statistical Testing. Monte Carlo method. Elsevier Publishing Co., 335 Jan Van Galenstraat, P.O. Box 211, Amsterdam (Netherlands), 1964.

[11] I. M. Sobol. Monte Carlo numerical methods. Moscow, Nauka, 1973.

Central Laboratory for Parallel Processing
Department of High Performance Computing and Parallel Algorithms
Bulgarian Academy of Sciences
Acad. G. Bonchev St.,bl. 25 A, 1113 Sofia, Bulgaria
e-mail: dimov@amigo.acad.bg gurov@iscbg.acad.bg
Web site: http://www.acad.bg/BulRTD/math/dimov2.html