Journal of Statistical Physics, Vol. 70, Nos. 5/6, 1993

Random Walk on Distant Mesh Points Monte Carlo Methods

I. Dimov[1] **and O. Tonev**[1]

Received September 21, 1991; final May 5, 1992

A new technique for obtaining Monte Carlo algorithms based on the Markov chains with a finite number of states is suggested. Instead of the classical "random walk on neighboring mesh points," a general way of constructing Monte Carlo algorithms that could be called "random walk on distant mesh points" is considered. It is applied to solve boundary value problems. The numerical examples indicate that the new methods are less laborious and therefore more efficient.

KEY WORDS: Monte Carlo method, random walk, Markov chain.

1. INTRODUCTION

Monte Carlo numerical methods are very efficient for solving multidimensional problems of mathematical physics and engineering where a linear functional of solution is sought. For problems to be efficiently solved by Monte Carlo methods, large-scale computations are needed, and so the construction of less laborious Monte Carlo algorithms is very important.

The methods constructed here are based on Markov chains with a finite number of states.

Let $G \subset \mathbb{R}^n$ be a bounded domain with a boundary ∂G.

We use the following notations: $\mathbf{x} = (x_1, x_2, ..., x_n)$ is a point in $\mathbb{R}^n$; $D^\alpha = D_1^{\alpha_1} D_2^{\alpha_2} \cdots D_n^{\alpha_n}$ is an $|\alpha| = \alpha_1 + \alpha_2 + \cdots + \alpha_n$ derivative, where $D_i = \partial/\partial x_i$, $i = 1, 2, ..., n$; and $C^k(\bar{G})$ is a space of functions $u(\mathbf{x})$ continuous on $\bar{G}$ such that $D^\alpha u$ exists in G and admits a continuous extension on $\bar{G}$ for every α: $|\alpha| \leqslant k$.

[1] Center for Informatics and Computer Technology, Bulgarian Academy of Sciences.

We consider the boundary value problem

$$Lu \equiv \sum_{|\alpha| \leqslant 2m} a_\alpha(\mathbf{x})\, D^\alpha u(\mathbf{x}) = -f(\mathbf{x}), \qquad \mathbf{x} \in G \tag{1}$$

$$u(\mathbf{x}) = g(\mathbf{x}), \qquad \mathbf{x} \in \partial G \tag{2}$$

where L is an arbitrary elliptic operator in $\mathbb{R}^n$ of order $2m$ and $a_\alpha(\mathbf{x}) \in C^\infty(\mathbb{R}^n)$.

A widely used definition of ellipticity is as follows.

The equation

$$\sum_{|\alpha| \leqslant 2m} a_\alpha(\mathbf{x})\, D^\alpha u(\mathbf{x}) = -f(\mathbf{x})$$

is called elliptic in a domain G if

$$\sum_{|\alpha| \leqslant 2m} a_\alpha(\mathbf{x}) \xi_{\alpha_1} \xi_{\alpha_2} \cdots \xi_{\alpha_n} \neq 0 \qquad \text{when} \quad |\xi| \neq 0$$

holds for every point $\mathbf{x} \in G$. The corresponding operator $\sum_{|\alpha| \leqslant 2m} a_\alpha(\mathbf{x}) D_\alpha$ is called elliptic in G.

This problem arises from models in field theory, electron optics, heat conductivity, and other areas of computational mathematics.

Assume that $f(\mathbf{x})$, $g(\mathbf{x})$, and the boundary ∂G satisfy all conditions ensuring that the solution of the problem (1), (2) exists and is unique.[1]

Constructing a regular mesh (lattice) with step h in $\mathbb{R}^n$ (see Fig. 1), we introduce the following notations: G_h is the set of all inner mesh points ($\gamma \in G_h \Leftrightarrow \gamma \in G$); ∂G_h is the set of all "boundary" mesh points ($\gamma \in \partial G_h$ if there exists a neighbor mesh point $\gamma' \in \mathbb{R}^n \backslash \bar{G}$); and v_h is a mesh function (function defined on a set of mesh points).

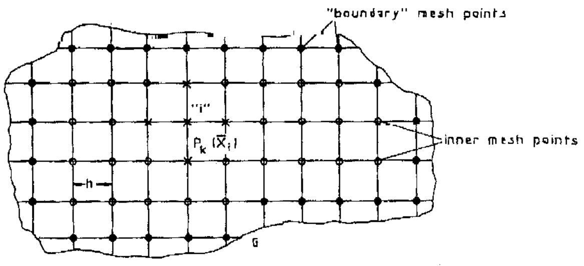

Fig. 1. A grid of mesh points. A mesh function v_h is defined on a set of mesh points $v_h = \{v_i : v_i = v(x_i)\}_{i=1}^n$, where n is the number of points belonging to the set $G_h \cup \partial G_h$. A simple pattern (lattice) $P_k(\bar{x}_h)$ in R^2 is shown.

Then the differential operator L at the mesh point $\mathbf{x}_i \in G_h$ is approximated by a difference operator L_h as follows:

$$(L_h v_h)_i = \sum_{\mathbf{x}_j \in P_k(\mathbf{x}_i)} A_h(\mathbf{x}_i, \mathbf{x}_j)\, v_h(\mathbf{x}_j) \tag{3}$$

where $A_h(\mathbf{x}_i, \mathbf{x}_j)$ are coefficients; and $P_k(\mathbf{x}_i)$ is a set of mesh points with center in $\mathbf{x}_i \in G_h$ called a *pattern.*

Since $\bar{G}_h = G_h \cup \partial G_h$ is a finite set, there exists a finite number of patterns $P_k(\mathbf{x}_i)$ ($k = 1, 2,..., m$ and m depends on $\mathbf{x}_i$) on which the differential operator could be approximated.

The "random walk on neighboring mesh points" Monte Carlo method is well known in the case when L is Laplace's operator[2]:

$$Lu \equiv D_1^2 u + D_2^2 u \tag{4}$$

and $P_k(\mathbf{x}_i)$ is the five-point pattern (see Fig. 1).

A Markov chain with a finite number of states is constructed. The transition probabilities on it are given as follows:

$$P_{\gamma\gamma'} = \begin{cases} 1/4, & \gamma \in G_h \\ \delta_{\gamma\gamma'}, & \gamma \in \partial G_h \end{cases} \tag{5}$$

where $\delta_{\gamma\gamma'}$ is the Kronecker notation ($\delta_{\gamma\gamma'} = 1$ when $\gamma \equiv \gamma'$ and it is 0 when $\gamma \neq \gamma'$) and γ' is a mesh point neighboring to γ. This means that beginning from a point $\gamma \in G_h$, the random process goes to the one of its neighboring points with a probability equal to 1/4. The random walk dies on the boundary ∂G_h. The "random walk on neighboring mesh point" is based on the following theorem:

Theorem 1. Let $\mathbf{x}_0$ be the originating point for the Markov chain constructed above. Then the solution of the difference problem corresponding to the problem (1), (2) at the point $\mathbf{x}_0$ is given by the mathematical expectation of the random variable constructed in the following manner:

$$\xi = \frac{1}{4} h^2 \sum_{i=0}^{i^*-1} f_h(\mathbf{x}_i) + g_h(\mathbf{x}_{i^*}) \tag{6}$$

This method is widely used, but it is relatively more laborious [the number of simple operations is $R_\varepsilon \approx c/\varepsilon^3$, where ε is the relative error needed for the solution of the problem (1), (2)].

Due to the transition from a point to one of its neighboring points this method has a slow rate of convergence.

The Monte Carlo method called the *spherical process*[3, 4] and its generalization[5, 6] are less laborious ($R_\mathrm{e} \approx |\ln \varepsilon|/\varepsilon^2$).

This approach is based on an integral representation of the solution by the Green's function. But for many boundary value problems an integral representation does not exist. When such a representation exists it is necessary to have a contractive integral operator with the Green's function in order to use a classical Monte Carlo method. This means that the norm of the integral operator K in the corresponding functional space (usually in the space L_1 or L_2) must be less than 1, i.e.,

$$\|K\| < 1$$

When this condition is not satisfied it is necessary to use a biased estimation stopping the random process in the ε-strip of the domain G.[4] But it is difficult to implement this when $f(\mathbf{x})$ is a mesh function. So it is better to construct an efficient Monte Carlo method for solving the discrete problem arising from the differential problem (1), (2) because usually the function $f(\mathbf{x})$ is obtained as a solution of another finite difference problem.

2. GENERAL DEFINITION OF THE "RANDOM WALK ON DISTANT MESH POINTS" METHODS

Due to the above considerations, the following questions arises: Are there any Markov processes for solving the problem (1), (2) realizing transitions not to neighboring but to more distant mesh points?

The definition of such processes will provide a construction of the "random walk on distant mesh points" algorithms.

We next consider the definition of the above Markov processes and the construction of the algorithms.

Assume that the values of $u_h(\mathbf{x})$ have been computed at the mesh points $\gamma \in \partial G_h$ using the values of $g(\mathbf{x})$ in such a way that the accuracy is the same as for the inner mesh points.

Then, by means of (3) the problem (1), (2) can be modified as follows:

$$\sum_{x_j \in P_{k_0}(x_i) \subset G_h} A_h(\mathbf{x}_i, \mathbf{x}_j)\, u_h(\mathbf{x}_j) = -f_h(\mathbf{x}_i) \tag{7}$$

or, which is the same,

$$u_h(\mathbf{x}_i) = \sum_{x_j \in P^*_{k_0}(x_i)} a_h(\mathbf{x}_i, \mathbf{x}_j)\, u_h(\mathbf{x}_j) + b_h(\mathbf{x}_i)\, f_h(\mathbf{x}_i) \tag{8}$$

where

$$P^*_{k_0}(\mathbf{x}_i) \equiv P_{k_0}(\mathbf{x}_i) \backslash \{\mathbf{x}_i\}$$

$$a_h(\mathbf{x}_i, \mathbf{x}_j) = -\frac{A_h(\mathbf{x}_i, \mathbf{x}_j)}{A_h(\mathbf{x}_i, \mathbf{x}_i)}$$

$$b_h(\mathbf{x}_i) = -\frac{1}{A_h(\mathbf{x}_i, \mathbf{x}_i)}$$

Instead of $u_h(\mathbf{x}_j)$ we can use the linear combination on the pattern $P^*_{k_1}(\mathbf{x}_j)$ where $\mathbf{x}_j \in P^*_{k_0}(\mathbf{x}_i)$ and P_{k_1} in general differs from P_{k_0}:

$$\begin{aligned} u_h(\mathbf{x}_i) = & \sum_{\mathbf{x}_j \in P^*_{k_0}(\mathbf{x}_i)} a_h(\mathbf{x}_i, \mathbf{x}_j) \left[\sum_{\mathbf{x}_k \in P^*_{k_1}(\mathbf{x}_j) \subset G_h} a_h(\mathbf{x}_j, \mathbf{x}_k)\, u_h(\mathbf{x}_k) + b_h(\mathbf{x}_j)\, f_h(\mathbf{x}_j) \right] \\ & + b_h(\mathbf{x}_i)\, f_h(\mathbf{x}_i) \\ = & \sum_{\mathbf{x}_k \in P^*_{k'_1}(\mathbf{x}_i) \subset G_h} a^1_h(\mathbf{x}_i, \mathbf{x}_k)\, u_h(\mathbf{x}_k) \\ & + \sum_{\mathbf{x}_j \in P^*_{k_0}(\mathbf{x}_i)} b^1_h(\mathbf{x}_j)\, f_h(\mathbf{x}_j) + b_h(\mathbf{x}_i)\, f_h(\mathbf{x}_i) \end{aligned} \tag{9}$$

where the $P^*_{k'_1}(\mathbf{x}_i)$ are a composition from the $P^*_{k_0}(\mathbf{x}_i)$ and $P^*_{k_1}(\mathbf{x}_j)$ $[\mathbf{x}_j \in P^*_{k_0}(\mathbf{x}_i)]$ while a_h and b_h are coefficients.

These replacements can be applied until a pattern $P^*_{k_{N+1}}(\mathbf{x}_i)$ is reached. Thus the following formula is obtained:

$$\begin{aligned} u_h(\mathbf{x}_i) = & \sum_{\mathbf{x}_m \in P^*_{k'_N}(\mathbf{x}_i) \subset \bar{G}_h} a^N_h(\mathbf{x}_i, \mathbf{x}_m)\, u_h(\mathbf{x}_m) + \sum_{\mathbf{x}_m \in P^*_{k_{N-1}}(\mathbf{x}_i)} b^N_h(\mathbf{x}_m)\, f_h(\mathbf{x}_m) \\ & + b_h(\mathbf{x}_i)\, f_h(\mathbf{x}_i) \end{aligned} \tag{10}$$

Assume that the operator L and the patterns P_k are such that

$$\sum_{\mathbf{x}_m \in P^*_{k'_N}(\mathbf{x}_i) \subset \bar{G}_h} a^N_h(\mathbf{x}_i, \mathbf{x}_m) = 1 \qquad \text{and} \qquad a^N_h(\mathbf{x}_i, \mathbf{x}_m) \geqslant 0 \tag{11}$$

Then the transition probabilities on the corresponding Markov chain can be defined in the following manner:

$$P(\mathbf{x}_i, \mathbf{x}_m) = a^N_h(\mathbf{x}_i, \mathbf{x}_m), \qquad \mathbf{x}_m \in P_{k'_N}(\mathbf{x}_i) \tag{12}$$

and the required random variable is

$$\theta = h^2 \sum_{i=1}^{i^*-1} \left[\sum_{\mathbf{x}_m \in P^*_{k_{N-1}}(\mathbf{x}_i)} b^N_h(\mathbf{x}_m)\, f_h(\mathbf{x}_m) + b_h(\mathbf{x}_i)\, f_h(\mathbf{x}_i) \right] + g_h(\mathbf{x}_{i^*}) \tag{13}$$

3. THE "RANDOM WALK ON MESH OCTAHEDRONS" METHOD

Consider the Laplace equation in R^3 approximated on seven-point patterns $[P_{k'_N}(\mathbf{x}_i)$ is the same for every $k = 1, 2, \ldots, N]$.

The superpattern $P_k(\mathbf{x}_i)$ in this case will be a family of concentric mesh octahedrons with centers in x_i the largest of which has an Nh

semidiagonal and the others, respectively, nh, where $n = N-2, N-4, \ldots, r$ and

$$r = \begin{cases} 1 & \text{when } N \text{ is odd} \\ 2 & \text{when } N \text{ is even} \end{cases}$$

So, if $\mathbf{x}_0 = (i_1^0 h, i_2^0 h, i_3^0 h)$ is the center of the octahedron set, the coordinates of an arbitrary mesh point on them will be

$$\begin{aligned} i_1 &= i_1^0 + n - l \\ i_2 &= i_2^0 + l \\ i_3 &= i_3^0 + m \end{aligned} \tag{14}$$

Figure 2 shows the intersection of the superpattern $P_{m,n,l}^N$ with the plane $x_3 = 0$.

Let $P_{m,n,l}^N$ be the transition probability from the point $\mathbf{x}_0$ to a point $\mathbf{x}_l$ which lies at the intersection of the mth plane and the nth octahedron from the family of octahedrons with a maximum semidiagonal Nh.

Theorem 2. The solution of the difference problem corresponding to the Dirichlet problem for the Poisson equation at the point $\mathbf{x}_0 \in G_h$ is equal to the mathematical expectation of the random variable

$$\theta = h^2(F_{\mathbf{x}_0^0} + F_{\mathbf{x}_1^0} + \cdots + F_{\mathbf{x}_{i^*-1}^0}) + g_h(\mathbf{x}_{i^*}^0)$$

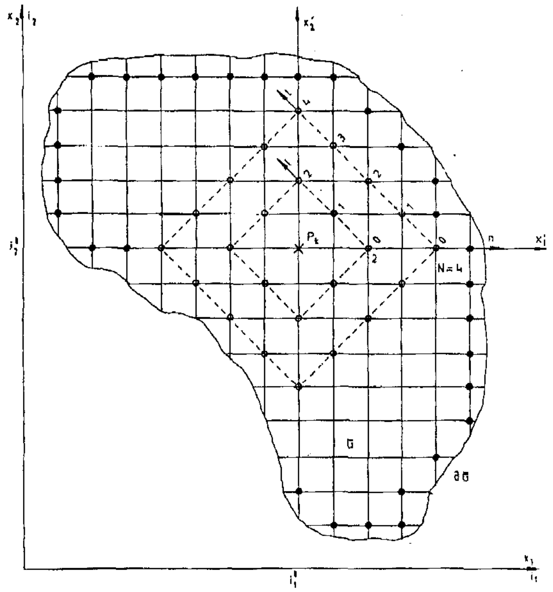

Fig. 2

The random variable θ is obtained when the transition probabilities $P^N_{m,n,l}$ on the Markov chain are $P^N_{m,n,l} = R^N_{m,n,l}/Q^N$, where

$$R^N_{m,n,l} = \begin{cases} \dbinom{N}{m}\dbinom{N-m}{l}, & l, m = 0,\dots, n, \quad \text{if} \quad n = N \\ R^{N-1}_{m,n+1,l} + R^{N-1}_{m,n+1,l+1} + R^{N-1}_{m,n-1,l-1} + R^{N-1}_{m,n-1,l} + R^{N-1}_{m+1,n,l} & \\ \qquad + R^{N-1}_{m-1,n,l}, & l = 1,\dots, n-1; \quad m = 0,\dots, N-1 \\ 2R^{N-1}_{m,n+1,l} + R^{N-1}_{m,n+1,l+1} + R^{N-1}_{m,n-1,l-1} + R^{N-1}_{m+1,n,l} + R^{N-1}_{m-1,n,l}, & \\ \qquad l = 1,\dots, n; \quad m = 0,\dots, N-1, & \text{if} \quad l = n \\ 0 & \text{in the other cases} \end{cases}$$

and

$$Q^N = 6^N - \sum_{i=1}^{N-1} 6^{N-1} R^i_{0,1,0}$$

and for the ith transition

$$F_{\mathbf{x}^0_i} = \sum_{m=1-N}^{N-1} \sum_{n=1}^{N-1} \sum_{l=1}^{n} S^N_{m,n,l}(f_{i^0_1+n-l,i^0_2+l,i^0_3+m} + f_{i^0_1+n-l,i^0_2-l,i^0_3+m}$$
$$+ f_{i^0_1-n+l,i^0_2+l,i^0_3+m} + f_{i^0_1-n+l,i^0_2-l,i^0_3+m})\, S^N_{0,0,0} f_{i^0_1,i^0_2,i^0_3}$$
$$T^N_{0,0,0} = 6^{N-1}$$
$$S^N_{m,n,l} = \frac{T^N_{m,n,l}}{Q^N} = \frac{1}{Q^N}\{R^{N-1}_{m,n,l} + 6[R^{N-2}_{m,n,l} + 6(R^{N-3}_{m,n,l} + \cdots + 6R^{m+n}_{m,n,l})\cdots]\}$$

These results are used for constructing the algorithm called "random walk on mesh octahedrons," which coincides with the "random walk on neighboring mesh points" when $N = 1$.

If Laplace's equation is approximated on a nine-point pattern (the set of corner points of the mesh cube and its center), we obtain a method that could be called "random walk on mesh cubes."

Next P^N_k will denote the transition probability from the mesh point (i^0_1, i^0_2, i^0_3) to the mesh point $(i^0_1 - N + 2k_1,\ i^0_2 - N + 2k_2,\ i^0_3 - N + 2k_3)$, which belongs to the mesh octahedron family the largest of which has a dimension $2Nh$.

To simplify the formulas, we introduce the following notation:

$$\gamma^i_j = \begin{cases} 1 & \text{if} \quad |j_r| = |i_r - i^0_r|, \quad r = 1, 2, 3 \\ 0 & \text{in the other cases} \end{cases} \tag{15}$$

Theorem 3. The solution of the difference Poisson equation at the point $\mathbf{x}_0 \in G_h$ is given by the mathematical expectation of the random variable

$$\theta = 4h^2(F_{\mathbf{x}_0^0} + F_{\mathbf{x}_1^0} + \cdots + F_{\mathbf{x}_{i^*-1}^0}) + g_h(\mathbf{x}_{i^*})$$

which corresponds to the Markov chain with transition probabilities $P_k^N = R_k^N / Q^N$, where

$$A_j^N = \begin{cases} \binom{N-2}{N/2-1-j_q}\binom{N-2}{N/2-1-j_r}, & j_s = \frac{N}{2} - 1 \\ \quad \text{for} \quad q = 1, 2, 3, \quad q \neq r \neq s \neq q & \\ \frac{1}{8}\binom{N}{N/2}^3, & j_1 = j_2 = j_3 = 0 \\ A_{j_1+1/2, j_2+1/2, j_3+1/2}^{N-1} + A_{j_1+1/2, j_2+1/2, j_3-1/2}^{N-1} + A_{j_1+1/2, j_2-1/2, j_3+1/2}^{N-1} & \\ \quad + A_{j_1+1/2, j_2-1/2, j_3-1/2}^{N-1} + A_{j_1-1/2, j_2+1/2, j_3+1/2}^{N-1} + A_{j_1-1/2, j_2+1/2, j_3-1/2}^{N-1} & \\ \quad + A_{j_1-1/2, j_2-1/2, j_3+1/2}^{N-1} + A_{j_1-1/2, j_2-1/2, j_3-1/2}^{N-1} & \text{in the other cases} \end{cases}$$

$$Q^N = 8^N - \sum_{i=1}^{N-1} 8^{N-i} R_{N/2, N/2, N/2}^i$$

and in the qth transition

$$F_{\mathbf{x}_q^0} = \sum_{l_3=1}^{2N-1} \sum_{l_2=1}^{2N-1} \sum_{l_1=1}^{2N-1} S_{l_1 l_2 l_3}^N f_{i_1^0 - N + l_1, i_2^0 - N + l_2, i_3^0 - N + l_3}$$

$$S_{l_1 l_2 l_3}^N = \frac{T_{l_1 l_2 l_3}^N}{Q^N} = \begin{cases} 8^{N-1}/Q^N, & l_1 = l_2 = l_3 = N \\ Q^{-N}(R_{(l_1-1)/2, (l_2-1)/2, (l_3-1)/2}^{N-1} + 8(R_{(l_1-2)/2, (l_2-2)/2, (l_3-2)/2}^{N-1} & \\ \quad + \cdots + R_{(l_1-m)/2, (l_2-m)/2, (l_3-m)/2}^{N-1}) \cdots) & \end{cases}$$

This result is used in constructing the "random walk on mesh squares" algorithm.

It is possible to construct an algorithm which is less laborious than those above.

Using a special sequence of patterns, we reach a superpattern of the same kind as the starting five-point one but in which one of the points "neighboring" $\mathbf{x}_0$ belongs to ∂G_h.

This construction is next applied to solve the Dirichlet problem for the Laplace equation in $\mathbb{R}^2$.

We shall use the following five-point patterns with centers at the mesh point $x_{i,j} = (ih, jh)$:

$$P_0(x_{i,j}) = \begin{pmatrix} & x_{i,j+1} & \\ x_{i-1,j} & x_{i,j} & x_{i+1,j} \\ & x_{i,j-1} & \end{pmatrix} \tag{16}$$

and

$$P_0(x_{i,j}) = \begin{pmatrix} x_{i-1,j+1} & & x_{i+1,j+1} \\ & x_{i,j} & \\ x_{i-1,j-1} & & x_{i+1,j-1} \end{pmatrix} \tag{17}$$

where the second-order derivatives are approximated with an accuracy $O(h^2)$.[1]

The first step is the approximation of the Laplace equation at the mesh point $x_{i,j}$ by using the pattern $P_{k_0}(x_0)$. Next, following the general definition of the method, we reach the approximation on a superpattern

$$P_N(x_{i,j}) = \begin{pmatrix} & x_{i,j+N} & \\ x_{i-N,j} & x_{i,j} & x_{i+N,j} \\ & x_{i,j-N} & \end{pmatrix}$$

we obtain the formula

$$u_{i,j} = \tfrac{1}{4}(u_{i-N,j} + u_{i+N,j} + u_{i,j-N} + u_{i,j+N}) \tag{18}$$

where $u_{i,j} = u_h(x_{i,j})$, $x_{i,j} \in G_h$.

This method could be called "random walk on most distant mesh points."

4. NUMERICAL EXAMPLES

To compare the labor cost of Monte Carlo algorithms, we will demonstrate the solution of a simple numerical example which has an exact solution.

The Laplace equation

$$(D_1^2 + D_2^2)u = 0 \tag{19}$$

with boundary condition

$$u = 400 + 100(x_1^2 - x_2^2) \tag{20}$$

where x belongs to the rectangle $\pi = \{0 \leqslant x_1 \leqslant 1,\ 0 \leqslant x_2 \leqslant 2\}$ was solved by the "random walk on neighboring mesh points" (RWNMP), "random walk on mesh squares" (RWMS), and "random walk on most distant mesh points" (RWMDMP) algorithms.

Table I. Results for Different Algorithms[a]

Coordinates of the point	Exact solution	K	Approximate solution RWNMP	Approximate solution RWMS	Approximate solution RWMDMP	$T = t \cdot \delta$ RWNMP	$T = t \cdot \delta$ RWMS	$T = t \cdot \delta$ RWMDMP
(1/2, 1)	325.00	200	323.2	119.4	302.7	0.1643	0.258	0.1374
(1/2, 1)	325.00	1000	316.5	315.8	316.3	4.366	2.858	0.268
(2/5, 2/5)	400.00	200	390.2	396.1	393.8	0.5386	0.1421	0.0217
(1/2, 2/5)	400.00	1000	393.6	394.3	396.3	1.869	1.008	0.1163

[a] RWNMP, random walk on neighboring mesh points; RWMS, random walk on mesh squares; RWMDMP, random walk on most distant mesh points.

The exact solution of the problem (19), (20) coincides with (20).

We seek the solution of (19), (20) at the mesh points with coordinates (0.5, 1) and (0.4, 0.4).

The transition probabilities for the RWMS algorithm are

$$P^N_{m,n,l} = P_{0,2,l} = \begin{cases} 1/12 & \text{when} \quad l = 0, 2 \\ 1/4 & \text{when} \quad l = 1 \end{cases} \tag{21}$$

The computing effort is measured by the parameter $t\delta$, where t is the time of obtaining the solution at the fixed point and δ is the relative error. In order to estimate the mathematical expectation, the arithmetic average of K realizations of the random processes is used (random variables are ξ and θ). The calculations are carried out on IBM PC compatible computers. Table I gives the calculations. The correlations between computing efforts for the other computers are approximately the same.

In conclusion, we mention that all Monte Carlo algorithms are parallel and could be easily realized on parallel computers.

REFERENCES

1. A. A. Samarskij, *The Theory of Difference Schemes* (Nauka, Moscow, 1983) [in Russian].
2. I. M. Sobol', *Monte Carlo Numerical Methods* (Nauka, Moscow, 1973) [in Russian].
3. M. E. Muller, Some continuous Monte Carlo methods for the Dirichlet problem, *Ann. Math. Stat.* **27**:569–589 (1956).
4. S. M. Ermakov and G. A. Mikhailov, *Statistical Modelling* (Nauka, Moscow, 1982) [in Russian].
5. B. S. Elepov and G. A. Mikhailov, Spherical process for solving the equation $\Delta u - cu = -g$, *Proc. Akad. Sci. USSR* **212**(1):15–18 (1973).
6. A. S. Sipin, Solving first boundary value problem for elliptic equations by Monte Carlo method, in *Monte Carlo Methods in Computational Mathematics and Mathematical Physics* (Novosibirsk, 1979), Vol. 2, pp. 113–119.