Liet. matem. rink., **44**, *spec. nr.*, 2004, 393–397

Parallel Monte Carlo computations in SCore environment

Svajonė VOŠTERIENĖ (KA)
e-mail: svajone.vosteriene@lka.lt

Abstract. Parallel Monte Carlo (PMC) methods are successful because particles are typically independent and easily distributed to multiple processors. We use the PC Cluster TAURAS [1] with the SCore cluster system software on top of Linux OS to present the strategy for parallelizing Monte Carlo calculations. The cluster help us to generate parallel random numbers for simulating our task and to computerize parallel programs. In this particle we present the PMC, who let us simulate the mortar firing. We propose computations that are very effective because the digital experiment can substitute costly firings at the shooting range, the pollution of the environment is reduced, etc.

Keywords: Monte Carlo methods, pseudo-random number generator, parallel computing..

1. Introduction

Numerical methods that are known as Monte Carlo methods can be described as statistical simulation methods, where statistical simulation is defined in quite general terms to be any method that uses sequences of random numbers to perform the simulation. Fig. 1 illustrates the idea of Monte Carlo.

The Linux OS have some functions for generating pseudo-random numbers using the linear congruential algorithm and 48-bit integer arithmetic. The *drand48()* function return non-negative double-precision floating-point values uniformly distributed between [0.0, 1.0). The *srand48()* function is initialization function, that should be called before using *drand48()*, *lrand48()* or *mrand48()*. All the functions work by gen-

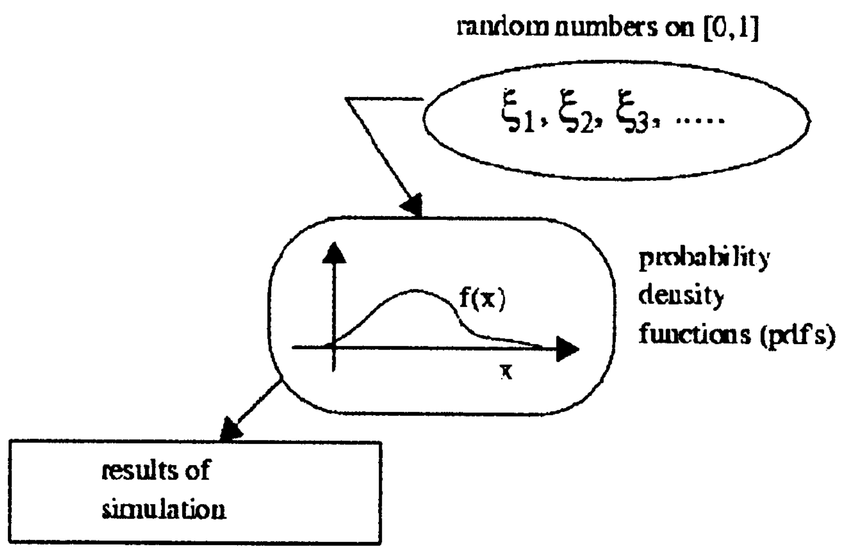

Fig. 1. Monte Carlo simulation of physical system.

erating a sequence of 48-bit integers, X_n, according to the linear congruential formula:

$$X_{n+1} = (aX_n + c) \bmod m, \tag{1}$$

where $n \geqslant 0$ and in isolated case $a = 5^{17}$ and $c = 0$. The parameter $m = 2^{48}$, hence 48-bit integer arithmetic is performed. Formula (1) means, that using numbers X_n are calculating pseudo-random numbers

$$\xi_n = 2^{-48} X_n. \tag{2}$$

For cluster with SCore environment on Linux OS, was homemade the program, that performs for each processes the sequences of pseudo-random numbers with initialization by server's clock time.

```
---------------------------------------------------------------------
*/ /*   random number initialization by server's clock time */ /*
---------------------------------------------------------------------
   srand( (unsigned int)time(NULL) );
    for (i=0; i<num_of_nodes; i++)
        rand_mas[i] = rand();
    }
    MPI_Bcast(rand_mas, num_of_nodes, MPI_INTEGER, server_id,
    MPI_COMM_WORLD);
    init_random(rand_mas[myid]);
    free(rand_mas);
```

This program establish a bound on the length of the subsequence, required for each realization. The first subsequence is chosen to consist of the first consecutive numbers in the sequence, the second subsequence to consist of the next consecutive numbers in the sequence, and so on. Each realization is assigned to the next processor that becomes free, then it becomes feasible to undertake the identical Monte-Carlo calculation on an arbitrary number of parallel processors. This holds regardless of the relative computational capacity or availability of the different processors.

2. PMC computations on cluster

Our calculations for PMC computations was performed in the PC Cluster TAURAS [1] with the SCore cluster system software on top of Linux OS. We used homemade programs, that generating parallel pseudo-random numbers and help to realize the simulation of trench-mortar firing. The task was for cluster environment to write the programs, that will help to find the necessary number of mines for hitting the target with the chosen confidence. All task was performed by scheme presented in Fig. 2. We chose generalized results from table [4], comparing date and state the dependence of resistance coefficient upon the distance to the target. We used the analytical solution, indicate the initial speed of mine v_0, the coefficient of resistance k and the influence of wind in the following manner

$$\begin{aligned} &v_0 + random[normal]\Delta v_0, \qquad && k + random[normal]\Delta k, \\ &F_1 + random[normal]\Delta F_1, && F_2 + random[normal]\Delta F_2, \end{aligned} \tag{3}$$

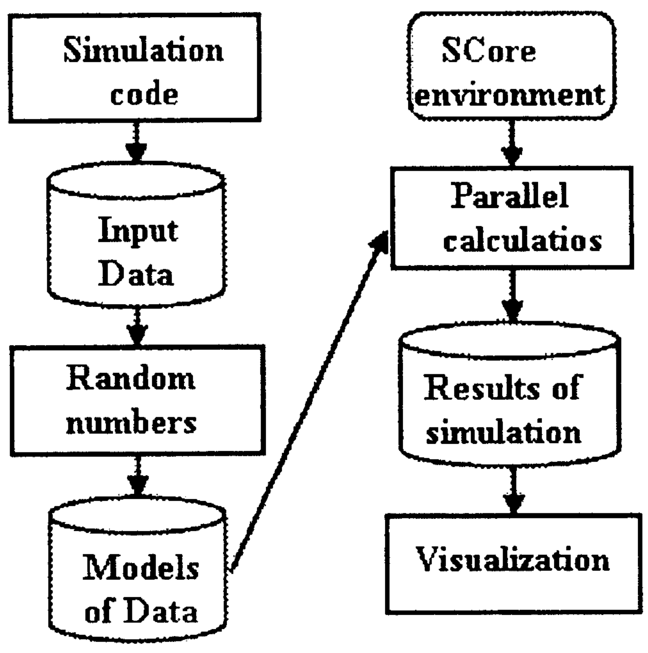

Fig. 2. The scheme of PMC simulation on cluster.

here *random* [*normal*] – generating standard normal distribution $N(0; 1)$ and Δv_0, Δk, ΔF_1, ΔF_2 - the maximal random errors of indicated variables discussed in specific situation. We repeat the numerical experience n times and every time evaluate the hurt part of target. For optimization task we calculate the optimal number of volleys, that must be shot to a group target situated at a definite distance for hitting more 95% of the target. It is known, that using the numerical experience n times the precision of results increase in proportional $1/\sqrt{n}$, where n means the number of experiments. When we need to get two dependability symbols, we doing the million numerical experiences and repeating million time calculations and then do the statistical processing of data. The time of calculations increase very fast. For example if the distance to the target is $L = 3050m$ and we shooting $N = 10000$ times, we can get the results presented in Fig. 3. One personal computer with 2,4 GHz CPU calculates this task about 3500 seconds. SCore cluster software and homemade PMC in C code with MPI standard [5] for cluster [1] let us do this computations better with portion evenly the job for all CPU's :

```
 cpu_speed_mas[server_id] = get_cpu_speed();
for(i=server_id+1; i<num_of_nodes; i++)
MPI_Recv(&cpu_speed_mas[i], 1, MPI_INT, i, 1, MPI_COMM_WORLD,
 &status);
for(i=0; i<num_of_nodes; i++)
    total_MHZ+=cpu_speed_mas[i];
MHZ_per_Iter = ceil((double)(total_MHZ)bandymu_sk);
for(i=0; i<num_of_nodes; i++){
    node_iters[i] = cpu_speed_mas[i]MHZ_per_Iter;
    tmp_total_iters+=node_iters[i];
}
delta_iters = bandymu_sk - tmp_total_iters;
i=0;
while(delta_iters>0){
    if (i>=num_of_nodes) i=0;
    node_iters[i] += 1;
    delta_iters--;
    i++;
}
```

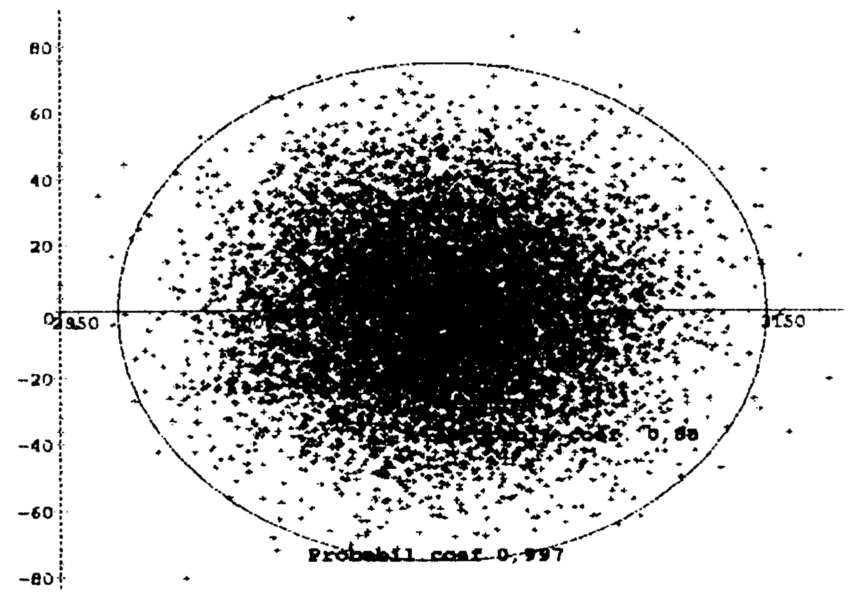

Fig. 3. The ellipse of dispersion of 10000 points hitting.

Table 1. The time of numerical experiences on separate number of CPU

Number of CPU's	6	8	10	12	14	16
Realization time in sec.	27	23	17	16	13	12

The time of calculations on separate number of CPU after 15 volleys of 10000 hits was perfect. It shows, that the PMC is better for realization our task. The time of numerical experiences are presented in Fig. 4 and Table 1.

After each volleys of hits we determines the degree of target destruction with given accuracy and calculate the number of mines that have missed the target. If the calculations are repeated a given number of times, we find mean value and evaluate the percentage N % injured part of target and average number of missing of the target. If we know the results of destruction, we can to sum up the situation. The computations shows that both the quality of the parallel pseudo-random number generator and the statistical independence of the results calculated on each processor are important.

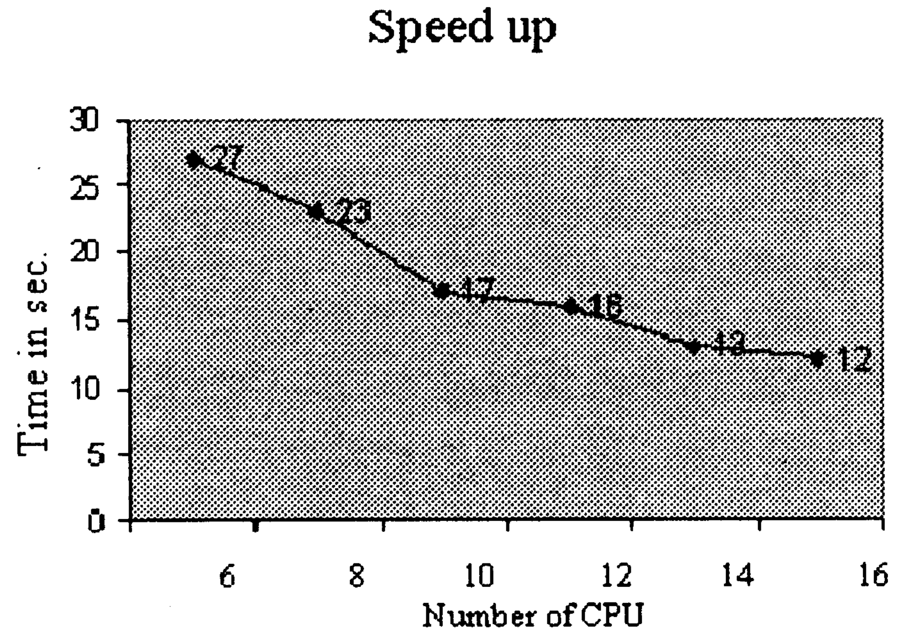

Fig. 4. Time dependence.

3. Conclusions

We present the parallel method for numerical simulation of trench-mortar tasks. These programs, help to calculate parameters of firing for destroy the target. Data that can be calculate: the standard quadratic variance, number of mines for realize successful firing and destroy separate or group targets with choose confidence probability and so on. All calculations continue a shot time and use the homemade pseudo-random number generator. These programs can be used for the teaching of the future officers as well.

References

1. `http://clusters.top500.org/db`.
2. I. Janchauskas, A. Venskus, *Ballistics*, Lithuanian Military Academy, Vilnius (1999) (in Lithuanian).
3. A. Pincevichus, R.J. Rakauskas, G. Misevichus, The numerical simulation in ballistics, *Nonlinear Analysis: Modeling and Control*, **6**(1), 89–104 (2001).
4. *Table of Rezults of Shooting by Trench-Mortar ChM-120*, Commander-in-Chief of Lithuanian Republic, CAS Grafema, Vilnius (1997) (in Lithuanian).
5. W. Gropp, E. Lusk, A.Skjellum, *Using MPI*, The MIT Press (1999).

REZIUMĖ

Svajonė Vošterienė. Lygiagretieji skaičiavimai Monte Carlo metodu SCore aplinkoje

Darbe parodytos asmeninių kompiuterių dirbančių lokaliame tinkle panaudojimo galimybės, sprendžiant netiesinių lygčių sistemą, apibūdinančią minosvaidžio iššautos minos judėjimą. Lygiagrečiai skaičiuojanti, naudojanti MPI standartą programa, padeda nustatyti optimalų minų skaičių taikiniui sunaikinti. Atsižvelgta ir į klasterio TAURAS architektūrą. Programa, paskirsto užduotis visiems klasterio procesoriams ir suteikia galimybę minimizuoti duomenų persiuntimo kaštus. Neteršiant aplinkos, šią programą galima naudoti kaip „treniruoklį“ apmokant būsimuosius karininkus.