Автоматика и телемеханика, № 5, 2007

PACS 02.60.Cb, 02.70.-c, 02.70.Tt, 02.70.Uu

© 2007 г. **М.А. МАРЧЕНКО, канд. физ.-мат. наук,**
Г.А. МИХАЙЛОВ, д-р физ.-мат. наук, чл.-корр. РАН
(Институт вычислительной математики и математической геофизики СО РАН, Новосибирск)

РАСПРЕДЕЛЕННЫЕ ВЫЧИСЛЕНИЯ ПО МЕТОДУ МОНТЕ-КАРЛО[1]

Обсуждаются вопросы эффективной параллельной реализации некоторых алгоритмов метода Монте-Карло. Описывается параллельная модификация генератора базовых псевдослучайных чисел, равномерно распределенных в единичном интервале. Описывается технология распределенных вычислений в сети персональных компьютеров с использованием разработанного авторами комплекса программ MONC.

1. Введение

При решении задач математической физики по методу Монте-Карло для искомого функционала φ подбирается вероятностное представление в виде математического ожидания от некоторой случайной величины

$$\varphi \approx \mathrm{E}\,\zeta = \mathrm{E}\,\zeta(\omega),$$

где ω – траектория моделируемого случайного процесса. Здесь имеется в виду, что вероятностное представление может быть приближенным, т.е. соответствующая случайная оценка имеет ненулевую детерминированную погрешность. Величина $\mathrm{E}\,\zeta$ приближенно оценивается выборочным средним для достаточного числа независимых реализаций ζ_n случайной величины ζ:

$$\varphi \approx \overline{\zeta} = \frac{1}{N}\sum_{n=1}^{N} \zeta_n.$$

Однако случайные оценки часто имеют большую трудоемкость. Здесь под трудоемкостью $C(\zeta)$ понимается средний объем вычислительной работы, необходимый для достижения заданной вероятностной погрешности, который пропорционален величине

$$C(\zeta) = t(\zeta)\,\mathrm{D}\,\zeta,$$

где $t(\zeta)$ – среднее время ЭВМ, необходимое для моделирования одного выборочного значения, $\mathrm{D}\,\zeta$ – дисперсия случайной оценки [1].

[1] Работа поддержана грантами Российского фонда фундаментальных исследований № 06-01-00586 и № 06-01-00046, грантом № НШ-4774.2006.1 программы "Ведущие научные школы", грантом Президента РФ № МК-1777.2005.1, грантом INTAS № 05-109-5267, грантом Лаврентьевского конкурса молодежных проектов СО РАН.

С целью понижения трудоемкости моделирование независимых выборочных значений можно распределить по процессорам параллельной вычислительной системы. Очевидно, что главным критерием осуществимости такой параллельной реализации является возможность "поместить" вычислительную программу моделирования выборочного значения в память каждого процессора.

При распределении вычислений по процессорам следует допускать возможность реализации различных объемов выборки на различных процессорах с использованием статистически оптимального способа осреднения результатов по формулам вида

$$\overline{\zeta} = \sum_{m=1}^{M} n_m \overline{\zeta}_m \Big/ \sum_{m=1}^{M} n_m,$$

где M – число процессоров, n_m – объем выборки для m-го процессора, $\overline{\zeta}_m$ – соответствующее среднее значение. Таким образом, можно добиться обратно пропорциональной зависимости величины трудоемкости случайной оценки от числа процессоров (при условии, что используемые процессоры имеют одинаковую производительность) [2].

Описанную методику параллельного моделирования можно легко модифицировать с целью эффективной оценки функционалов, зависящих от параметра $x \in X$:

$$\varphi(x) \approx \mathrm{E}\,\zeta(x;\omega).$$

При этом моделируемое распределение случайной величины ω может зависеть, а может и не зависеть от параметра x. Аналогичную методику можно также использовать для оценки функционалов, возникающих в результате осреднения базовых функционалов по случайному параметру σ задачи, например по случайной плотности среды. Вероятностное представление при этом имеет вид двойного математического ожидания

$$\varphi \approx \mathrm{E}\,\mathrm{E}\,\zeta(\omega,\sigma).$$

Примеры подобных задач приведены в разделах 2 и 3.

Как правило, при параллельной реализации необходимый объем выборки базовых случайных чисел очень велик, поэтому целесообразно использовать "длиннопериодные" псевдослучайные последовательности. Удовлетворяющий этому требованию "параллельный" генератор описан в разделе 4. Этот генератор был проверен с помощью содержательных статистических тестов.

Описанная методология параллельной реализации соответствует концепции использования инфраструктуры GRID, которая быстро развивается в последнее время [3]. Отметим, что существует достаточное число разработок GRID-систем, предназначенных для распределенных вычислений по методу Монте-Карло (см., например, [4–6]). В настоящей работе предлагается система MONC, обладающая бóльшей простотой и удобством в использовании, чем имеющиеся аналоги (см. раздел 5).

2. Глобальная оценка решения

Для глобальной оценки функции

$$\varphi(x) = \int_Y g(x,y)P(dy)$$

в ограниченной области D можно оценить ее значения в узлах сетки с шагом h методом Монте-Карло и затем осуществить какое-либо восполнение, например линейное

непрерывное. Такую оценку обозначим через $\widetilde{\varphi}(x)$. Вообще говоря, функция $\widetilde{\varphi}(x)$ – это случайное поле, распределение которого связано с объемом выборки (т.е., с числом реализаций в методе Монте-Карло), с типом базовой оценки ξ и с шагом h.

Задача сходимости оценки $\widetilde{\varphi}$ к φ в некоторой метрике может быть решена на основе рассмотрения величины

$$B(\varphi, \widetilde{\varphi}) = \mathrm{E}\,\|\varphi(x) - \widetilde{\varphi}(x)\|_{L(D)},$$

где $L(D)$ – соответствующее Банахово пространство, так как $P(|\xi| \geqslant \varepsilon) \leqslant \mathrm{E}\,|\xi|/\varepsilon\ \forall\varepsilon$. Эта задача связана с трудоемкостью алгоритма, т.е. со средним числом операций, которые обеспечивают выполнение неравенства $B(\varphi, \widetilde{\varphi}) < \delta$. Оценить $B(\varphi, \widetilde{\varphi})$ наиболее просто в пространстве $L_2(D)$:

$$B^2(\varphi, \widetilde{\varphi}) = \left(\mathrm{E} \left\{ \int\limits_D [\varphi(x) - \widetilde{\varphi}(x)]^2 dx \right\}^{1/2} \right)^2 \leqslant$$

$$\leqslant \int\limits_D \mathrm{E}\,[\varphi(x) - \widetilde{\varphi}(x)]^2\, dx = \int\limits_D \mathrm{D}\ \widetilde{\varphi}(x) dx + \int\limits_D [\varphi(x) - \mathrm{E}\,\widetilde{\varphi}(x)]^2\, dx.$$

Предположим, что производные 2-го порядка от $\varphi(x)$ равномерно ограничены в D. Тогда можно построить такую аппроксимацию сеточной функции, что выполнено соотношение

$$\int\limits_D [\varphi(x) - \mathrm{E}\,\widetilde{\varphi}(x)]^2 dx \leqslant C_0 h^4.$$

При этом

$$B^2(\varphi, \widetilde{\varphi}) \leqslant d/n + C_0 h^4. \tag{1}$$

Цель настоящего раздела – уменьшение трудоемкости функционального алгоритма метода Монте-Карло для заданного способа аппроксимации сеточной функции. Здесь используется лишь информация о порядке соответствующей аппроксимационной погрешности в зависимости от шага сетки. Если этот порядок можно повысить, то следующие далее оценки изменяются очевидным образом.

В рассмотренных условиях задача приближенной минимизации трудоемкости может быть сформулирована в виде:

$$S_0 = n t_0 h^{-k} \to \min_{n,h}, \quad d/n + C_0 h^4 = \delta^2, \tag{2}$$

где k – размерность пространства, n – обьем выборки, h – шаг сетки, смысл d виден из (1) и $t_0 = t\ \mathrm{mes}\,(D)$, где t – затраты на одну реализацию ξ, $\mathrm{mes}\,(D)$ – мера Лебега множества D. Оптимальный порядок величин h, n и S_0 таков:

$$h_0^* \asymp \delta^{1/2}, \quad n_0^* \asymp \delta^{-2}, \quad S_0^* \asymp \delta^{-(2+k/2)}.$$

При этом $B(\varphi, \widetilde{\varphi}) < \delta$ и, следовательно, $\widetilde{\varphi}$ сходится к φ в метрике L_2.

Аналогичные результаты получаются для метрики $C(D)$ путем использования теоремы вложения пространства $W_2^l(D)$ в пространство $C(D)$ при $2l > k$. Если используются производные первого порядка, то по необходимости $k = 1$, т.е. рассматривается решение одномерного уравнения (или многомерного уравнения на данной

линии). Здесь

$$\|\varphi - \widetilde{\varphi}\|_C^2 < K \left[\int\limits_D [\varphi(x) - \widetilde{\varphi}(x)]^2 dx + \int\limits_D [\varphi'(x) - \widetilde{\varphi}'(x)]^2 dx \right].$$

Поскольку дисперсия суммы независимых случайных величин равна сумме их дисперсий, то при независимой оценке значений решения в узлах решетки необходимо использовать неравенство

$$\mathrm{D}\,\widetilde{\varphi}'(x) < \frac{d(x)}{nh^2}.$$

Кроме того если $\widetilde{\varphi}'$ кусочно-постоянна, то

$$\int\limits_D [\varphi'(x) - \mathrm{E}\,\widetilde{\varphi}'(x)]^2 dx < C_1 h^2.$$

Таким образом, асимптотически при $h \to 0$

$$\|\varphi - \widetilde{\varphi}\|_C^2 < \frac{d_1}{nh^2} + C_1 h^2.$$

Эта оценка приводит к следующей задаче минимизации трудоемкости:

$$S_1 = nt_0 h^{-k} \to \min_{n,h}, \quad \frac{d_1}{nh^2} + C_1 h^2 = \delta^2. \tag{3}$$

Оптимальные порядки здесь таковы:

$$h_1^* \asymp \delta, \quad n_1^* \asymp \delta^{-4}, \quad S_1^* \asymp \delta^{-5}.$$

Итак, в одномерном случае трудоемкость глобальной оценки решения в $C(D)$ квадратична сравнительно с оценкой в $L_2(D)$. Трудоемкость оценки в $C(D)$ может быть существенно уменьшена путем зависимой оценки величины φ, обеспечивающей соотношение

$$|\widetilde{\varphi}(x) - \widetilde{\varphi}(x+h)| < Ch, \quad h \to 0,$$

с вероятностью 1. При этом вместо (3) возникает задача

$$S_2 = nt_0 h^{-1} \to \min_{n,h}, \quad d_2/n + C_2 h^2 = \delta^2,$$

решение которой дает

$$h_2^* \asymp \delta, \quad n_2^* \asymp \delta^{-2}, \quad S_2^* \asymp \delta^{-3}.$$

Имеются различные способы коррелирования оценок по методу Монте-Карло [1]. Например, целесообразно использовать одни и те же псевдослучайные числа для различных точек, дополнительно распределяя эти числа специальным детерминированным способом между отдельными траекториями (см. раздел 4). При этом серии траекторий, исходящих из различных точек фазового пространства, можно моделировать на различных процессорах.

3. Метод "двойной рандомизации"

1. Далее рассматриваются рандомизированные оценки вероятностных моментов решений задач со случайными параметрами. Пусть уравнение $L\phi = f$ решается методом Монте-Карло на основе моделирования некоторого случайного процесса с траекториями ω. Это означает, что случайные величины $\xi_k(\omega)$ строятся так, что

$$\mathrm{E}\,\xi_k(\omega) = J_k, \quad k = 1, 2, \ldots, m,$$

где J_k – оцениваемые функционалы от ϕ.

Пусть оператор L и функция f зависят от случайного поля σ (например, случайная среда в теории переноса, случайная сила в теории упругости и т.д.). Тогда

$$\xi_k = \xi_k(\omega, \sigma), \quad J_k = J_k(\sigma)$$

и

$$\mathrm{E}\,[\xi_k(\omega, \sigma)|\sigma] = J_k(\sigma),$$

где траектории ω и σ, вообще говоря, зависимы.

Рассмотрим задачу оценки величин

$$J_k = \mathrm{E}J_k(\sigma), \quad R_{kj} = \mathrm{E}\,[J_k(\sigma)J_j(\sigma)], \quad k, j = 1, \ldots, m,$$

где E обозначает математическое ожидание по распределению σ.

Для решения этой задачи целесообразно использовать метод "двойной рандомизации", который строится на основе следующих представлений:

$$\begin{aligned} &\mathrm{E}\,J_k(\sigma) = \mathrm{E}_\sigma \mathrm{E}_\omega \xi_k(\omega, \sigma) = \mathrm{E}_{(\omega,\sigma)}\xi_k(\omega, \sigma) \\ &\mathrm{E}\,[J_k(\sigma)J_j(\sigma)] = \mathrm{E}_{(\omega_1,\omega_2,\sigma)}[\xi_k(\omega_1, \sigma)\xi_j(\omega_2, \sigma)], \end{aligned} \tag{4}$$

где ω_1 и ω_2 – условно-независимые траектории, моделируемые для одной реализации σ, а индекс математического ожидания означает соответствующее распределение. Ясно, что надо предположить существование полного математического ожидания, представленного в (4), т.е.

$$\mathrm{E}_{(\omega,\sigma)}|\xi_k(\omega, \sigma)| < +\infty \;\; \mathrm{E}_{(\omega_1,\omega_2,\sigma)}\big[|\xi_k(\omega_1, \sigma)\xi_j(\omega_2, \sigma)|\big] < +\infty.$$

Соотношения (4) показывают, что для оценки J_k достаточно строить только одну траекторию для фиксированного σ, а оценка величины R_{kj} требует построения двух условно-независимых траекторий. Для оптимизации такого алгоритма естественно использовать метод расщепления (см., например, [1]).

2. В качестве частного случая использования формулы (4), можно рассмотреть рандомизацию оценки по столкновениям (см. [1])

$$\xi = \sum_{n=0}^{N} Q_n h(x_n),$$

где

$$Q_0 = \frac{f(x_0)}{\pi(x_0)}, \quad Q_n = Q_{n-1}\frac{k(x_{n-1}, x_n)}{p(x_{n-1}, x_n)}.$$

Пусть $\widetilde{k}(x_{n-1},x_n)$, $\widetilde{f}_0$, $\widetilde{h}_n$ – независимые несмещенные оценки соответствующих величин $k(x_{n-1},x_n), f(x_0), h(x_n)$ (например, случайные оценки интегралов, выражающих эти величины), $\widetilde{Q}_n$ – несмещенные оценки весов Q_n и K_1 – интегральный оператор с ядром $\mathrm{E}\,|\widetilde{k}(x',x)|$. Если $\rho(K_1)<1$, $\mathrm{E}\,|\widetilde{h}|\in L_\infty$, $\mathrm{E}\,|\widetilde{f}|\in L_1$, то [1]

$$\mathrm{E}\,\widetilde{\xi}=\mathrm{E}\,\xi,\quad \text{где}\quad \widetilde{\xi}=\sum_{n=0}^{N}\widetilde{Q}_n\widetilde{h}_n.$$

Кроме того,

$$\mathrm{E}\,\widetilde{\xi}^2=(\chi,h[2\varphi^*-h])+(\chi,\mathrm{D}\,\widetilde{h}),$$

где χ – ряд Неймана для уравнения

$$\chi'(x)=\int_X\frac{\mathrm{E}\,\widetilde{k}^2(x',x)}{p(x',x)}\chi'(x')dx'+\frac{\mathrm{E}\widetilde{f}^2(x)}{\pi(x)},\quad \text{или}\quad \chi'=K_p'\chi'+\mathrm{E}\,(\widetilde{f}^2/\pi),$$

если $\rho(K_p')<1$, $\mathrm{E}\,\widetilde{f}^2/\pi\in L_1$.

Если рандомизируется лишь $h(x)$, то

$$\mathrm{D}\,\widetilde{\xi}=\mathrm{D}\,\xi+(\chi,\mathrm{D}\,\widetilde{h}).$$

Можно показать, что из соотношения $\rho(K_p')<1$ следует соотношение $\rho(K_1')<1$.

4. Параллельный генератор псевдослучайных чисел

Параллельный генератор псевдослучайных чисел представлен в [7]. Кроме того, вычислительная программа, реализующая генератор и инструкции по ее использованию доступны в интернете [8].

1. Для моделирования базовых случайных величин, равномерно распределенных в интервале $(0,1)$, предлагается использовать следующую формулу [9]:

$$u_0=1,\ u_n\equiv u_{n-1}A\ (\mathrm{mod}\ 2^{128}),\qquad \alpha_n=u_n2^{-128},\qquad n=1,2,\ldots,$$

где

$$A\equiv 5^{100109}\ (\mathrm{mod}\ 2^{128}).$$

Здесь обозначение $a\equiv b\ (\mathrm{mod}\ m)$ означает операцию сравнения чисел a и b по модулю m. Формула такого вида называется 128-*битным конгруэнтным генератором псевдослучайных чисел* или *методом вычетов* [1], число A называется *множителем генератора*. Последовательность чисел $\{\alpha_n\}$ является периодической, длина ее периода равна $L=2^{126}\approx 10^{38}$ [1].

Для распределения псевдослучайных чисел между разными процессорами предлагается следующий порядок их использования. Последовательность $\{u_n\}$ предварительно разбивается на подпоследовательности длины μ, начинающиеся с чисел $u_{m\mu}$, $m=0,1,2,\ldots$, (будем их называть *начальными значениями подпоследовательностей*). Разные подпоследовательности используются на разных процессорах. Значение "прыжка" генератора μ должно выбираться из соображений, чтобы μ псевдослучайных чисел хватало для моделирования на каждом процессоре. Ясно, что

для метода вычетов начальные значения $u_{m\mu}$ указанных подпоследовательностей получаются по формуле

$$u_{(m+1)\mu} \equiv u_{m\mu} A_\mu (\mathrm{mod}\ 2^{128}),$$

где

$$A_\mu \equiv A^\mu (\mathrm{mod}\ 2^{128}).$$

Итак, для моделирования на m-м процессоре используется подпоследовательность метода вычетов, начинающаяся с

$$\alpha_{m\mu} = u_{m\mu} \cdot 2^{-128}.$$

Получаемый таким образом "крупномасштабный" генератор называется *bf-генератором* (сокращение от big-frog-генератор).

Для *bf*-генератора рекомендуется следующее значение длины "прыжка":

$$\mu = 10^{26} \approx 2^{86}.$$

Таких количеств псевдослучайных чисел для каждого процессора с избытком хватает для вычислительных потребностей. Предлагаемый *bf*-генератор позволяет, вообще говоря, распределять исходную последовательность равными долями длины 10^{26} примерно на $10^{12} \approx 2^{40}$ процессоров.

Для коррелирования результатов решения различных задач целесообразен следующий порядок использования псевдослучайных чисел, называемый *lf-генератором* (сокращение от little-frog-генератор). На m-м процессоре соответствующая ему подпоследовательность дополнительно разбивается на подпоследовательности длины d, начинающиеся с чисел $u_{(m-1)\mu+ld}$, $l = 0, 1, 2, \ldots$, и каждая подпоследовательность используется для построения соответствующих "выборочных траекторий" моделируемого процесса (т.е. отдельных случайных испытаний). Значение d должно выбираться из тех соображений, что d псевдослучайных чисел практически достаточно для построения одной траектории. Ясно, что начальные значения подпоследовательностей для *lf*-генератора для m-го процессора можно вычислять по формуле

$$u_{(m-1)\mu+ld} \equiv u_{(m-1)\mu+(l-1)d} A^d\ (\mathrm{mod}\ 2^{128}), \quad l = 0, 1, 2, \ldots$$

2. С целью проверки построенного *bf*-генератора были реализованы тесты n-мерной равномерности для выборки объема 10^{10}, которая была получена объединением начальных 10^9 чисел из первых 10 подпоследовательностей. Такая проверка ориентирована на одновременное использование 10 вычислительных процессоров, на каждом из которых предполагается использовать не более 10^9 псевдослучайных чисел с последующим осреднением полученных статистических оценок, как это указано во введении. Многомерные распределения проверялись на равномерность для $n = 1, 2, \ldots, 7$ по критерию χ^2 с разбиением по каждой оси на 100 частей для $n = 2, 3$ и на 10 частей для $n = 4, \ldots, 7$. Обозначим через $k(n)$ число степеней свободы распределения χ^2, соответствующее величине n. Таким образом, число классов, т.е. элементарных "кубиков", равное $k(n) + 1$, определялось величиной 10^{2n} при $n = 2, 3$ и величиной 10^n при $n = 4, \ldots, 7$.

Для $n = 1$ было использовано оптимальное в смысле максимума мощности критерия χ^2 число классов, определяемое формулой из [10]

$$k(1) + 1 \sim 4\sqrt[5]{2}(R/d_\alpha)^{2/5}, \tag{5}$$

где R – объем выборки, а постоянную $d_\alpha = O(1)$ можно практически полагать равной двум. В данном случае $R = 10^{10}$ и $k(1) + 1 \approx 34800$ согласно (5).

Обозначим через $\widetilde{\chi}^2_{k(n)}$ выборочное значение критерия:

$$\widetilde{\chi}^2_{k(n)} = \frac{1}{r_n} \sum_{i=1}^{k(n)+1} \left(r_i^{(n)} - r_n \right)^2,$$

где $r_n = R(n)/(k(n)+1)$ – теоретическая частота попадания в класс, $R(n)$ – объем выборки, соответствующий значению n, $\{r_i^{(n)}\}$ – полученные выборочные частоты. Для анализа результатов статистической проверки построенного bf-генератора использовалось то обстоятельство, что для "настоящих" случайных чисел величина $\widetilde{\eta}_n = (\widetilde{\chi}^2_{k(n)} - k(n))/\sqrt{2k(n)}$ при используемых значениях $k(n)$ с большой степенью точности является стандартно нормальной [10] и для нее выполнены следующие соотношения:

$$\text{(6)} \qquad \mathrm{P}\left(|\widetilde{\eta}_n| > 1\right) \approx 0{,}32, \quad \mathrm{P}\left(|\widetilde{\eta}_n| > 2\right) \approx 0{,}05, \quad \mathrm{P}\left(|\widetilde{\eta}_n| > 3\right) \approx 0{,}003.$$

Такое применение критерия χ^2 оправдано тем, что он является состоятельным для любой альтернативной гипотезы, меняющей теоретические частоты r_n [10].

Полученные численные результаты статистического теста сведены в таблице.

Результаты статистической проверки многомерной равномерности bf-генератора

Параметры	$n=1$	$n=2$	$n=3$	$n=4$	$n=5$	$n=6$	$n=7$
$R(n)$	10^{10}	$5 \cdot 10^9$	$\approx 3{,}3 \cdot 10^9$	$2{,}5 \cdot 10^9$	$2 \cdot 10^9$	$\approx 1{,}67 \cdot 10^9$	$\approx 1{,}43 \cdot 10^9$
$k(n)$	34799	9999	999999	9999	99999	999999	9999999
$\widetilde{\eta}_n$	0,215	0,622	1,472	−0,76	0,913	−0,448	1,104

Отметим, что значения случайных величин $\widetilde{\eta}_n$ попадают за пределы интервала $(-1, 1)$ в двух случаях из семи, что соответствует соотношениям (6). Учитывая большой объем выборок и состоятельность критерия, можно заключить, что полученные значения $\widetilde{\chi}^2_{k(n)}$ не являются значимыми, а тестирование можно назвать успешным.

5. Система распределенных вычислений MONC

Система распределенных вычислений MONC представлена в [2]. Кроме того, дистрибутивы служебных программ для системы MONC и инструкции по их использованию доступны в интернете [11].

1. Цель разработки системы MONC (сокращение от Monte Carlo) – создание универсального высокопроизводительного инструмента для распределенных вычислений по методу Монте-Карло на базе сети персональных компьютеров. Система MONC ориентирована на пользователя, которым выступает математик-вычислитель.

Система MONC имеет архитектуру клиент-сервер с несколькими серверами и одним клиентом. Система реализована для выполнения на персональных компьютерах, работающих под управлением операционных систем семейства Microsoft Windows и объединенных локальной сетью. Организация взаимодействия компьютеров осуществляется по протоколу TCP/IP.

Система MONC распределяет статистически независимые копии задачи по персональным компьютерам в сети, отдает команды на их запуск, следит за ходом выполнения заданий и по их завершению производит копирование и осредняет результаты

расчетов. В рамках системы MONC легко организовать продолжение расчетов после их вынужденной остановки.

Можно выделить следующие достоинства системы MONC:

• Высокая производительность при решении задач по методу Монте-Карло: все компьютеры в составе системы используются оптимально, с максимальной загрузкой, независимо от их быстродействия.

• Экономия интеллектуальных затрат математика-вычислителя, имеющего необходимость производить параллельные вычисления: он может целиком сконцентрироваться на математической стороне своей задачи и создавать программу на любом доступном ему языке программирования. Таким образом, для работы в системе MONC пользователь фактически должен предоставить исполняемый файл, созданный им только для одного компьютера.

• Экономическая доступность внедрения и простота в эксплуатации. В состав системы можно включать компьютеры с любым быстродействием.

2. Для решения прикладной задачи по методу Монте-Карло с помощью системы MONC математик-вычислитель должен подготовить *Проект пользователя*, который является совокупностью следующих файлов и информации:

1. Исполняемого файла (с расширением 'exe') для *вычислительной программы*, созданной на любом языке программирования с соблюдением описанных далее простых и необременительных требований, а также входных файлов, необходимых для ее выполнения.

2. Файла с *начальными значениями подпоследовательностей* для *bf*-генератора (далее – *файл для bf-генератора*), имеющего описанный ниже формат.

3. Информации о файле с результатами расчетов *вычислительной программы* (далее – *файл с результатами*), который должен иметь определенный ниже формат.

Здесь под результатами расчетов понимаются выборочные средние, оценивающие искомые функционалы.

Вычислительная программа должна удовлетворять следующим требованиям:

• Использовать описанный в разделе 4 конгруентный генератор псевдослучайных чисел.

• Быть консольным приложением для Windows и не требовать интерактивного ввода информации пользователем.

• В числе входных файлов должна иметь файл с начальными значениями подпоследовательности для генератора псевдослучайных чисел (см. раздел 4). Имя этого файла должно совпадать с именем *файла для bf-генератора* и сам он должен иметь такой-же формат.

• Сохранять результаты расчетов в *файле с результатами*.

• Обеспечивать периодическое сохранение промежуточных результатов расчетов в *файле с результатами* для обеспечения возможности их осреднения во время счета и для их "аварийного" осреднения Клиентом.

Файл для -генератора должен удовлетворять следующим требованиям:

• Он должен состоять из строк с одинаковым количеством числовых значений в каждой из них. Каждая строка должна содержать начальные значения подпоследовательностей для генератора псевдослучайных чисел, используемого в *вычислительной программе* (см. раздел 4).

• Количество строк в нем должно соответствовать количеству Серверов, входящих в систему MONC.

Далее приведен пример такого файла, рассчитанного на 10 компьютеров. Пояснения по поводу содержащихся в нем числовых значений можно получить в [7, 8].

Пример файла для bf-генератора.

```
1   0      0      0      0      0      0      0      0      0
1   0   7916   6769   8113   7234   4142   5015   3567   1526
1   0   7640   5347   2291   4799    945   6715   5714    914
1   0   7364   3925   7109    884   2888   4164   3748   1899
1   0   7088   2503   6183   3683   6066   4620   7519   1298
1   0   6812   1081   7705   5003   6576   7149   6654   1302
1   0   6536   7851   3482   4845    514   2625   3325   1280
1   0   6260   6429   1708   3208    360   6496   4053   1876
1   0   5984   5007   2382     92   2210   1444   3331    915
1   0   5708   3585   5504   3689   2159   2919      0   1593
```

Формат *файла с результатами* должен удовлетворять следующим требованиям:

• Первая его строка должна содержать количество фактически смоделированных *вычислительной программой* независимых реализаций (испытаний).

• Остальная часть файла должна иметь вид матрицы с фиксированным количеством строк и столбцов. Эта матрица должна содержать выборочные средние значения, оценивающие искомые функционалы. Эти выборочные средние значения должны соответствовать указанному в первой строке количеству смоделированных реализаций.

Числовые значения разделяются пробелами или символами табуляции. В конце любой строки обязателен символ перевода каретки.

Количество строк и столбцов в матрице не ограничено, но должно находиться в разумных пределах.

Далее приведен пример такого файла (// означает комментарий).

Пример файла с результатами.

```
100000                  // Количество реализаций
1.0e-10       1.239e-5  // Выборочные средние значения
-23.558e-1    41.025e1  // Выборочные средние значения
11.532e-0     -3.131e2  // Выборочные средние значения
```

2. Опишем далее конфигурацию компьютеров в сети и программное обеспечение системы MONC.

Среди компьютеров в сети выделяется один компьютер – Клиент, остальные становятся Серверами. Отметим, что под Клиента может выделяться любой из компьютеров сети, что позволяет пользователю максимально удобно работать с системой MONC. На Клиент и на Серверах устанавливается соответствующее служебное программное обеспечение (ПО), отвечающее их функциям в процессе выполнения *Проекта пользователя*. Схема взаимодействия компьютеров изображена на рисунке.

Для хранения файлов *Проекта пользователя* на жестком диске Клиента выделяется каталог – *рабочий каталог системы MONC*. Для хранения копий файлов *Проекта пользователя* на каждом Сервере выделяется каталог – *каталог копии Проекта пользователя*.

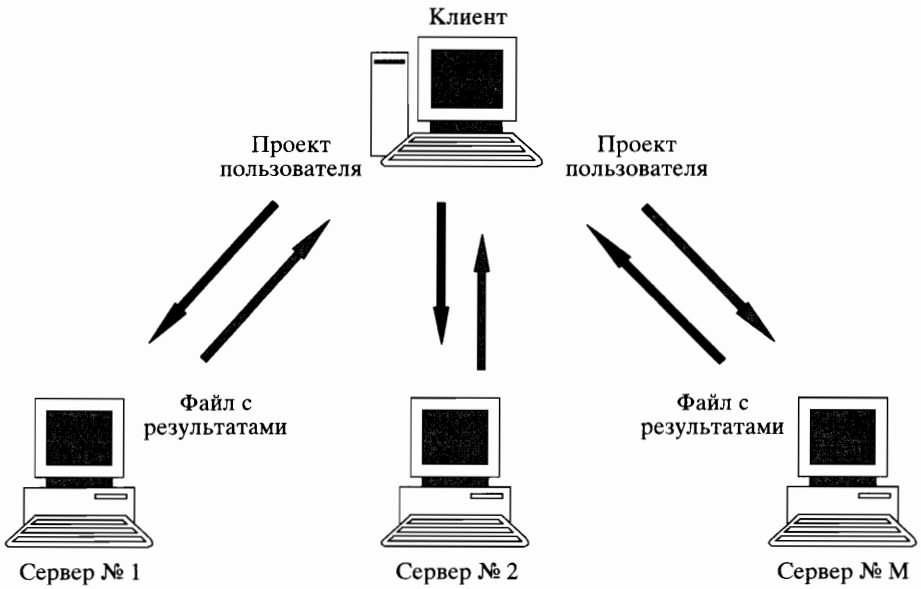
Служебное ПО состоит из следующих программ:

• программа, запускаемая на Клиенте, а также служебный файл с информацией о Серверах, входящих в систему MONC (находится на Клиенте).

• программа, запускаемая на Сервере.

Служебное ПО, установленное на Клиенте, обеспечивает выполнение этим компьютером следующих задач:

• Предоставление возможности перед началом расчетов задать необходимые параметры для *Проекта пользователя*: количество Серверов, имя *вычислительной*



Взаимодействие компьютеров в системе MONC.

программы, имя *файла для bf-генератора*, имя *файла с результатами*, а также количество строк и столбцов в нем.

• Подготовка для каждого из Серверов копии файлов *Проекта пользователя*. В частности, подготовка файлов с начальными значениями подпоследовательностей для генераторов, которые будут использоваться в соответствующих копиях *вычислительной программы* на Серверах, а именно "вырезание" нужных строк из *файла для bf-генератора*.

• Копирование подготовленных копий файлов *Проекта пользователя* на Серверы в *каталоги копии Проекта пользователя*.

• Посылка команд на запуск копий *вычислительной программы* на Серверах.

• Во время выполнения копий *вычислительной программы* на Серверах – возможность осреднения промежуточных результатов расчетов из соответствующих *файлов с результатами* на каждом из Серверов.

• Возможность остановки копий *вычислительной программы* на Серверах. При такой остановке, а также при успешном окончании выполнения копий *вычислительной программы* на Серверах – копирование *файлов с результатами* с Серверов на Клиент и их осреднение. Сохранение результатов осреднения в специальном файле с информацией о количестве смоделированных траекторий на каждом из Серверов, а также с информацией о времени выполнения копий *вычислительной программы* на каждом из Серверов.

• При непредвиденном аварийном завершении выполнения копий *вычислительной программы* на Серверах – осреднение сохраненных *файлов с результатами*.

• Возможность организации продолжения расчетов. При этом система автоматически использует предыдущий *файл с результатами*.

Служебное ПО, установленное на каждом из Серверов, обеспечивает выполнение этими компьютерами следующих задач:

• Получение команды с Клиента на запуск копии *вычислительной программы*, уже находящейся в *каталоге копии Проекта пользователя*, и ее запуск.

• Отслеживание ошибок, возникающих во время исполнения копии *вычислительной программы*, и при возникновении ошибки – отправка информации об ошибке Клиенту.

• Получение команды на остановку копии *вычислительной программы* и ее выполнение.

• Отправка сигнала об успешном завершении выполнения копии *вычислительной программы* Клиенту.

4. Таким образом, для запуска своей задачи в системе MONC пользователь должен произвести следующий действия:

• скопировать в *рабочий каталог системы* MONC на Клиенте *вычислительную программу* и необходимые для ее работы "входные" файлы, а также *файл для bf-генератора*;

• определить параметры работы системы MONC в служебной программе на Клиенте: количество Серверов, имя *вычислительной программы*, имя *файла для bf-генератора*, имя *файла с результатами*, а также количество строк и столбцов в нем.

Поскольку заранее трудно сделать точную оценку количества моделируемых реализаций на каждом процессоре, то целесообразно это количество не ограничивать. При необходимости следует воспользоваться возможностью системы прерывать расчеты одновременно на всех процессорах в любой момент времени. Такая методика позволит все компьютеры в составе системы использовать оптимально, с максимальной загрузкой, независимо от различий в их быстродействии.

Для продолжения расчетов следует только скопировать новые начальные значения подпоследовательностей для *bf*-генератора в *файл для bf-генератора* и запустить *Проект пользователя* с этим файлом. Получаемые "новые" результаты расчетов нужно далее осреднить вместе со "старыми" результатами (см. Введение). Файл с большим числом начальных значений подпоследовательностей для *bf*-генератора можно взять из [8].

5. Следующий пример использования системы MONC для решения трудоемких задач, связанных с моделированием диффузии, подробно описан в [2, 12].

Рассмотрим задачу оценки математических ожиданий аддитивных функционалов от траекторий диффузионных процессов, представляющих собой интегралы по некоторым малым областям расширенного фазового пространства от концентрации траекторий. При этом, по сути, оцениваются малые вероятности, поэтому необходимое из соображений точности число выборочных реализаций очень велико.

Пусть 3-мерный диффузионный процесс $y(\cdot)$ задан автономной системой стохастических дифференциальных уравнений (СДУ) в смысле Ито:

$$dy(t) = a(y(t))dt + b(y(t))dw(t), \quad 0 \leqslant t \leqslant T, \quad y(0) = y_0,$$

где $a(y)$ – 3-мерная векторная функция, $b(y)$ – матричная функция размерности 3×3, $b(y) = \{b_{ij}(y)\}$, $w(t)$ – 3-мерный винеровский процесс, y_0 – координаты источника.

Для статистического моделирования траекторий системы СДУ используется метод Эйлера:

$$y_{i+1} = y_i + \triangle t a(y_i) + \sqrt{\triangle t} b(y_i)\xi_i, \quad i = 0, 1, \ldots, n.$$

Здесь y_i – численное решение СДУ в узле t_i равномерной сетки на временно́м интервале с постоянным шагом интегрирования $\triangle t = T/n$, а $\{\xi_i\}_{i=0,1,\ldots,n}$ – последовательность независимых между собой нормальных случайных векторов с независимыми в совокупности компонентами, имеющими стандартное нормальное распределение.

Рассмотрим задачу вычисления полной концентрации $\Phi(T, y_c)$ диффузионных траекторий во временно́м интервале $[0, T]$ в некоторой заданной и удаленной от источника точке y_c. При переходе к приближенной оценке точку y_c заменим шаром Ω_c

малого радиуса r_c с центром в точке y_c; точные диффузионные траектории заменим приближенными траекториям метода Эйлера. Таким образом получим следующую статистическую оценку (будем называть ее оценкой прямого моделирования или аналоговой оценкой):

$$\Phi(T, y_c) \approx \mathrm{E}\,\zeta_D = \mathrm{E} \sum_{i=0}^{n} \omega(y_i)\triangle t,$$

где $\omega(y) = \chi_{\Omega_c}(y)/V_c$, V_c – объем шара Ω_c. Таким образом, нужно моделировать независимые реализации случайной величины ζ_D.

В качестве примера рассмотрим систему СДУ на временном интервале $[0, 10]$ с постоянным вектором сноса, единичной матрицей диффузии и нулевым начальным условием: $a \equiv (0,\ 0{,}5,\ 0{,}5)^{\mathrm{T}}$, $b \equiv I$, $y_0 \equiv 0$. Рассмотрим шар Ω_c радиуса $r_c = 0{,}1$ с центром в точке $y_c = (8, 0, 0)$. Для метода Эйлера положим $\triangle t = 10^{-2}$.

Для указанных параметров задачи значение величины Φ может быть выражено в аналитическом виде [12]. Кроме того, при реализации аналоговой оценки необходимо использовать большой объем псевдослучайных чисел. Поэтому, наряду с численным исследованием порядка детерминистской погрешности аналоговой оценки, на этом примере можно осуществить дополнительную статистическую проверку описанного в разделе 4 параллельного генератора псевдослучайных чисел.

Согласно предварительным тестовым расчетам для данной задачи необходимый уровень относительной погрешности в 10% мог быть достигнут при количестве реализаций случайной величины ζ_D порядка 10^7, что потребовало бы порядка 100 ч машинного времени при использовании одного компьютера с процессором типа Pentium III 733 МГц.

В составе системы MONC использовались 9 персональных компьютеров разной производительности: 1 компьютер с процессором Pentium III 733 МГц, 3 компьютера с процессорами Celeron 466 МГц, 3 компьютера с процессорами типа Duron 800 МГц, 1 компьютер с процессором типа Athlon 1133 и 1 компьютер с процессором типа Athlon XP 1500+.

Проект пользователя состоит из следующих файлов:

• *вычислительной программы* direct 3d.exe и нескольких файлов со входной информацией для нее;

• *файла для bf-генератора* frog.txt;

• имени *файла с результатами* direct.res, который имеет 2 строки и 1 столбец.

Программа direct 3d написана на языке Fortran 90 с учетом изложенных выше требований, а именно:

• в программе используется описанный в разделе 4 генератор псевдослучайных чисел;

• все входные параметры задаются в соответствующих файлах;

• начальные значения для подпоследовательности *bf*-генератора задаются в файле frog.txt;

• промежуточные результаты расчетов периодически сохраняются в файле direct.res через каждые 100 смоделированных реализаций;

• количество моделируемых реализаций не ограничивается.

Структура *файла с результатами* direct.res следующая:

• строка 1 – количество смоделированных траекторий;

• строка 2 – выборочное среднее значение ($\approx \mathrm{E}\,\zeta_D$);

• строка 3 – выборочный второй момент ($\approx \mathrm{E}\,\zeta_D^2$).

Ниже приведен пример распечатки файла direct.res, сформированного *вычислительной программой* на одном из Серверов.

Пример распечатки файла direct.res.

```
1056012
3.024352793635296E-005
1.234814266571703E-004
```

Расчеты проводились примерно 13 ч, после чего были остановлены. Общее количество смоделированных реализаций составило $N = 15\ 691\ 700$. В результате было получено следующее значение искомого функционала:

$$\Phi(T, y_c) \approx 2{,}94 \cdot 10^{-5} \pm 3{,}51 \cdot 10^{-7},$$

что соответствует относительной погрешности в 9,62%.

Результаты расчетов показали, что "среднее" быстродействие одного "узла" системы MONC с указанным составом персональных компьютеров соответствует быстродействию компьютера с процессором типа Pentium III 733 МГц. Таким образом, применение системы MONC при решении задач показало существенный выигрыш в быстродействии по сравнению с "непараллельными" вычислениями.

СПИСОК ЛИТЕРАТУРЫ

1. *Михайлов Г.А., Войтишек А.В.* Численное статистическое моделирование. Методы Монте-Карло. М.: Образ. изд. центр "Академия", 2006.
2. *Марченко М.А.* Комплекс программ MONC для распределенных вычислений методом Монте-Карло // Сиб. ЖВМ. 2004. Т. 7. № 1. С. 43–55.
3. *Foster I., Kesselman K. (Editors)* The Grid 2 Blueprint for a New Computing Infrastructure. Second Edition. Elsevier, 2003.
4. *Mendes B., Pereira A.* Parallel Monte Carlo Driver (PMCD) – a software package for Monte Carlo simulations in parallel // Comput. Phys. Comm. 2003. V. 151. P. 89–95.
5. *Thomasona M.G., Longtona R.F., Gregora J. et al.* Simulation of emission tomography using grid middleware for distributed computing // Comput. Method. Program. Biomed. 2004. V. 75. P. 251–258.
6. *Yuasaa F., Tobimatsub K., Kawabata S.* Recent developments in parallelization of the multi-dimensional integration package DICE // Nuclear Instrument. Method. Phys. Res. A. 2006. V. 559. P. 306–309.
7. *Mikhailov G.A., Marchenko M.A.* Parallel realization of statistical simulation and random number generators // Russ. J. Numer. Anal. Math. Modelling. 2002. V. 17. № 1. P. 113–124.
8. *Марченко М.А.* Параллельный 128-битный конгруэнтный генератор псевдослучайных чисел: рукопись доступна на сайте http://osmf.sscc.ru/~mam/generator_rus.htm.
9. *Dyadkin I.G., Hamilton K.G.* A study of 128-bit multipliers for congruential pseudorandom number generators // Comput. Phys. Comm. 2000. V. 125. P. 239–258.
10. *Кендалл М., Стьюарт А.* Статистические выводы и связи. М.: Наука, 1973.
11. *Марченко М.А.* Система распределенных вычислений MONC: рукопись доступна на сайте http://osmf.sscc.ru/~mam/monc_rus.htm.
12. *Марченко М.А., Михайлов Г.А.* Весовые алгоритмы статистического моделирования диффузионных процессов // Журн. вычисл. мат. и мат. физики. 2003. Т. 43. № 4. С. 571–584.

Статья представлена к публикации членом редколлегии А.И. Кибзуном.

Поступила в редакцию 18.12.2006