РЕАЛИЗАЦИЯ МАСШТАБИРУЕМЫХ АЛГОРИТМОВ РАСПРЕДЕЛЕННОГО СТАТИСТИЧЕСКОГО МОДЕЛИРОВАНИЯ НА СУПЕРКОМПЬЮТЕРЕ С ПОМОЩЬЮ ПРОГРАММНОЙ БИБЛИОТЕКИ PARMONC

М. А. Марченко

Институт вычислительной математики и математической геофизики СО РАН, 630090, Новосибирск, Россия
Новосибирский государственный университет, 630090, Новосибирск, Россия

УДК 681.327; 658.588.2

Представлена библиотека PARMONC (Parallel Monte Carlo), предназначенная для эффективного распараллеливания различных приложений метода Монте-Карло, требующих больших вычислительных затрат. При распараллеливании используется "естественная" крупноблочная фрагментированность алгоритмов метода Монте-Карло. "Ядром" библиотеки является тщательно протестированный, быстрый и надежный длиннопериодный параллельный генератор псевдослучайных чисел. Библиотека представляет собой простой в использовании программный инструмент для организации распределенных вычислений, не требующий от пользователя знания языка MPI; распараллеливание сложных последовательных программ статистического моделирования не вызывает затруднений. Библиотека PARMONC позволяет масштабировать вычисления на практически неограниченное число ядер, которое зависит только от используемой вычислительной системы, причем вычислительная нагрузка равномерно распределяется по всем ядрам.

Ключевые слова: статистическое моделирование, метод Монте-Карло, генераторы псевдослучайных чисел, распределенные вычисления, библиотеки программ.

In this paper, the software library PARMONC (an acronym of the PARallel MONte Carlo) that was developed for the massively parallel simulation by the Monte Carlo method on supercomputers is presented. Native coarse-grain parallelism of the Monte Carlo method is used in the library. A "core" of the library is a well-tested, fast and reliable long-period parallel pseudorandom numbers generator. The PARMONC is an easy-to-use program instrument to perform distributed stochastic simulation. Routines from the PARMONC can be called in the user-supplied programs without explicit usage of MPI instructions. Parallelization of complicated sequential Monte Carlo programs can be made in an easy way. The PARMONC enables one to scale Monte Carlo simulation to practically infinite number of processors, a computational load being automatically distributed among processors in an optimal way.

Key words: stochastic simulation, Monte Carlo method, pseudorandom numbers generators, distributed computing, program libraries.

Введение. Можно предположить, что в ближайшем будущем при проведении компьютерного моделирования будут широко использоваться вероятностные имитационные модели

Работа выполнена при финансовой поддержке Российского фонда фундаментальных исследований (коды проектов 12-01-00034, 12-01-00727, 13-01-00746, 13-07-00589) и в рамках Междисциплинарных интеграционных проектов СО РАН № 39, 47, 126, 130.

и методы Монте-Карло (методы численного статистического моделирования). С одной стороны, это предположение можно обосновать тем, что вероятностные имитационные модели дают адекватное описание физических, химических, биологических и других явлений при их рассмотрении "из первых принципов"; с другой — алгоритмы метода Монте-Карло, реализующие вероятностные модели, допускают возможность эффективного распараллеливания. Использование методов Монте-Карло весьма перспективно, поскольку в ближайшем будущем вероятно появление суперЭВМ экзафлопсного уровня [1].

Отметим важную область применения методов статистического моделирования. При повышении температуры газовых смесей выше критической (например, в процессе горения газовых топлив в камерах сгорания, соплах и горелках различной геометрической формы) нередко возникают течения химически реагирующих газов. Численное моделирование таких течений методом Монте-Карло является весьма трудоемким, поскольку требует учета различных физических и химических особенностей процессов, происходящих при горении. В последние десятилетия понимание химии горения (по крайней мере, с участием небольших молекул) и возможности моделирования процессов горения на компьютерах большой мощности достигли такого уровня, который обеспечивает необходимую надежность результатов [2]. Численное моделирование процессов горения требует значительных вычислительных затрат и относится к числу задач, для решения которых необходимо использование суперЭВМ. Такие вычислительные потребности обусловлены большим средним временем расчета каждой реализации ансамбля тестовых частиц, большим числом независимых реализаций ансамбля, которое определяется исходя из необходимой точности расчетов, и большим объемом используемой машинной памяти. Применение описываемой в настоящей работе библиотеки PARMONC при исследовании таких задач позволит существенно уменьшить трудоемкость расчетов.

1. Распределенное статистическое моделирование. Под численным статистическим моделированием обычно понимается реализация с помощью компьютера вероятностной модели некоторого объекта с целью оценивания изучаемых интегральных характеристик на основе закона больших чисел [3]

$$\varphi \approx \mathrm{E}\zeta(\omega) \approx \bar{\zeta} = \frac{1}{L}\sum_{i=1}^{L} \zeta^{(i)}.$$

При этом чем больше объем выборки L, составленной из смоделированных независимых реализаций $\{\zeta^{(i)}\}$, тем выше точность оценивания, причем статистическая погрешность убывает обратно пропорционально квадратному корню от объема выборки. Трудоемкость статистического моделирования (т. е. затраты машинного времени, необходимые для достижения заданного уровня статистической погрешности ε) определяется величиной $\tau_L L$, которая в свою очередь пропорциональна величине

$$C = \tau_L \mathrm{D}\zeta \varepsilon^{-2}.$$

Здесь τ_L — среднее время моделирования одной реализации случайной величины ζ; $\mathrm{D}\zeta$ — дисперсия случайной величины. "Большие" задачи статистического моделирования характеризуются либо большими величинами τ_L или $\mathrm{D}\zeta$, либо малой величиной ε, либо сочетанием этих факторов. Кроме того, "большие" задачи статистического моделирования часто требуют больших объемов оперативной памяти.

С целью уменьшения трудоемкости можно применять распараллеливание статистического моделирования [4–6]. В зависимости от условий задачи и параметров алгоритма статистического моделирования применяются различные методики параллельной реализации.

Следует отметить класс приложений, например методы прямого статистического моделирования пространственно неоднородных задач газовой динамики и теории дисперсных систем, для которых ресурсов одного вычислительного ядра недостаточно для моделирования отдельных реализаций (в данном случае реализацией является ансамбль тестовых частиц) и требуется применять методы пространственной декомпозиции ансамбля. При этом можно использовать способы динамической балансировки вычислительной нагрузки, основанные на специальных формулах прогноза времени вычислений для каждого вычислительного ядра [5]. Суть метода распределенного статистического моделирования состоит в распределении моделирования независимых реализаций по вычислительным ядрам с периодическим осреднением полученных выборочных значений по статистически эффективной формуле

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

где M — общее число ядер; n_m — объем выборки для m-го ядра; $\bar{\zeta}_m$ — соответствующее среднее значение.

Очевидно, что главным критерием осуществимости такой параллельной реализации является возможность "поместить" данные вычислительной программы для моделирования реализаций в оперативную память каждого ядра. Заметим, что при проведении распределенного статистического моделирования допустимо использовать вычислительные ядра с разной производительностью [4]. При этом обмен данными можно свести к минимуму, допуская только начальную "загрузку" вычислительных ядер и финальное получение выборочных средних. Действуя таким образом, можно добиться обратно пропорциональной зависимости величины трудоемкости "распределенной" случайной оценки (1) от числа ядер при условии, что используемые ядра имеют одну и ту же производительность. Возможные отказы компонент вычислительной системы и связанные с этим потери данных компенсируются моделированием реализаций на других вычислительных узлах.

Описанную методику параллельного моделирования нетрудно модифицировать для получения 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 или не зависеть от параметра x. Аналогичную методику можно использовать также для оценки функционалов, возникающих в результате осреднения базовых функционалов по случайному параметру σ задачи, например по случайной плотности среды. При этом вероятностное представление имеет вид двойного математического ожидания

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

Примеры подобных задач приведены в [4].

Как правило, при параллельной реализации необходимый объем выборки базовых случайных чисел очень велик, поэтому целесообразно использовать длиннопериодные псевдослучайные последовательности. В частности, для решения "больших" задач методом Монте-Карло предлагается использовать генератор вида [4–7]

$$u_0 = 1,\ u_n \equiv u_{n-1}A \pmod{2^{128}}, \qquad \alpha_n = u_n 2^{-128}, \qquad n = 1, 2, \ldots,$$

где

$$A \equiv 5^{100\,109} \pmod{2^{128}},$$

запись $a \equiv b \pmod{m}$ означает операцию сравнения чисел a и b по модулю m. Формула такого вида называется 128-битным конгруэнтным генератором псевдослучайных чисел, или методом вычетов, число A называется множителем генератора. Последовательность чисел $\{\alpha_n\}$ является периодической, длина ее периода равна $L = 2^{126} \approx 10^{38}$ [3].

Предлагается следующий порядок распределения псевдослучайных чисел между разными вычислительными ядрами. Последовательность $\{u_n\}$ предварительно разбивается на подпоследовательности длиной μ, начинающиеся с чисел $u_{m\mu}$, $m = 0, 1, 2, \ldots$ (будем называть их начальными значениями подпоследовательностей). На разных ядрах используются разные подпоследовательности. Необходимо выбирать такое значение "прыжка" генератора μ, чтобы μ псевдослучайных чисел было достаточно для моделирования на каждом ядре. При использовании метода вычетов начальные значения $u_{m\mu}$ указанных подпоследовательностей вычисляются по формуле

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

где

$$A_\mu \equiv A^\mu \pmod{2^{128}}.$$

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

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

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

Для *bf*-генератора рекомендуется значение длины "прыжка" $\mu = 10^{26} \approx 2^{86}$. Такого количества псевдослучайных чисел достаточно для удовлетворения вычислительных потребностей для каждого ядра. Предлагаемый *bf*-генератор позволяет распределять исходную последовательность равными долями длиной 10^{26} приблизительно на $10^{12} \approx 2^{40}$ ядер.

Для коррелирования результатов решения различных задач целесообразен порядок использования псевдослучайных чисел, называемый *lf*-генератором (little-frog-генератор). На m-м ядре соответствующая подпоследовательность дополнительно разбивается на подпоследовательности длиной d, начинающиеся с чисел $u_{(m-1)\mu+ld}$, $l = 0, 1, 2, \ldots$. Каждая подпоследовательность используется для построения соответствующих "выборочных траекторий" моделируемого процесса (т. е. отдельных случайных испытаний). При выборе значения d следует учитывать, что d псевдослучайных чисел практически достаточно для построения одной траектории. Начальные значения подпоследовательностей для *lf*-генератора для m-го ядра можно вычислить по формуле

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

Заметим, что длина периода генератора позволяет независимо распределять псевдослучайные числа по реализациям на практически неограниченное число вычислительных ядер. Параллельный генератор (см. (2)) успешно используется в ряде институтов СО РАН на протяжении последних 10 лет.

2. Реализация распределенного статистического моделирования с помощью библиотеки PARMONC. С целью унификации применения распределенного статистического моделирования при решении широкого круга задач методом Монте-Карло разработана и внедрена программная библиотека PARMONC (Parallel Monte Carlo). Область применения библиотеки — решение "больших" задач статистического моделирования для естественных и гуманитарных наук (физика, химия, биология, медицина, экономика, социология и др.) Библиотека PARMONC установлена на кластерах Сибирского суперкомпьютерного центра (ССКЦ КП СО РАН) и может использоваться в вычислительных системах с аналогичной архитектурой. При этом для использования библиотеки не требуются какие-либо конкретные компиляторы языков C, Fortran или MPI. Инструкции по использованию библиотеки с примерами приведены в [8, 9].

Возможность применения библиотеки PARMONC определяется "естественной" крупноблочной фрагментированностью программ статистического моделирования. В упрощенном виде структура такого рода программ следующая:

```
void main( void ) {
    long int i, L;
    TypeRL RL, SUBT;
    SUBT = 0.0;
    //  цикл по реализациям
    for( i = 0; i < L; i++ ) {
        // далее идут операторы, вычисляющие реализацию RL
        ...
        SUBT= SUBT + RL;
    }
    SUBT = SUBT / L;
}
```

Здесь L — общее число независимых реализаций случайного объекта, задаваемых композитным типом данных TypeRL (допускается поэлементное суммирование переменных такого типа); реализации RL моделируются внутри цикла по переменной i. Полученные таким образом реализации RL (статистически независимые в совокупности) добавляются к "счетчику" SUBT и на выходе из цикла осредняются, что дает статистическую оценку искомого математического ожидания случайного объекта. При распараллеливании последовательных программ с помощью PARMONC определяется процедура Realization (моделирующая подпрограмма), возвращающая одну реализацию RL (возвращение осуществляется через аргумент процедуры). При этом считается, что моделирующая подпрограмма использует потоки псевдослучайных чисел, генерируемых внешней по отношению к ней подпрограммой. Взаимосвязь программных объектов при статистическом моделировании показана на рис. 1. С учетом такой взаимосвязи цикл по независимым реализациям и финальное осреднение заменяются вызовом библиотечной процедуры следующего вида:

```
parmoncc( realization, L, SUBT, ... );
```

Здесь имя моделирующей подпрограммы и общее число независимых реализаций передаются в библиотечную процедуру parmoncc в качестве входных аргументов; выборочное среднее будет возвращаться в переменную SUBT; для простоты остальные аргументы процедуры

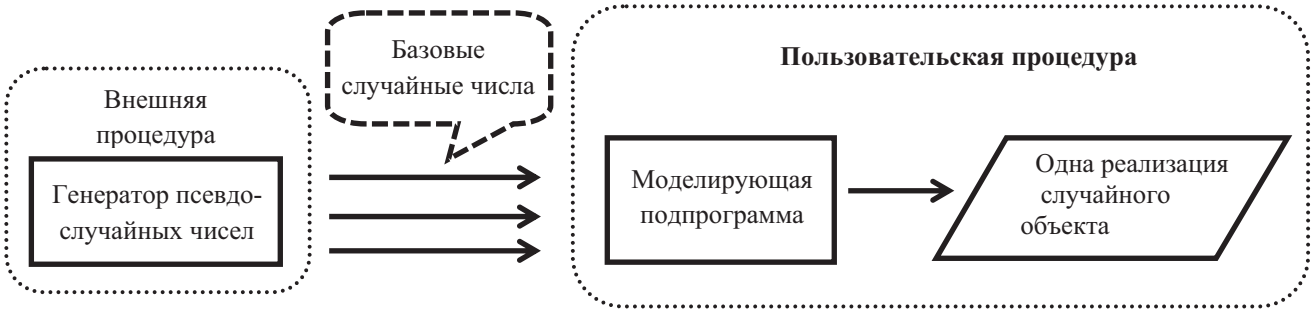

Рис. 1. Взаимосвязь программных объектов при статистическом моделировании

parmoncc опущены и заменены многоточием. Процедура parmoncc автоматически распределяет моделирование независимых реализаций по вычислительным ядрам. Все остальные операторы пользовательской программы остаются без изменений. Таким образом, в итоге имеем следующий код, пригодный для компиляции и сборки с помощью библиотеки PARMONC:

```
void main( void ) {
    TypeRL SUBT;
    parmoncc (realization, L, SUBT, ... );
}

void realization( TypeRL  RL ){
    //  далее идут операторы моделирующей подпрограммы
    ...
    //  вычисленная реализация возвращается в переменную RL
}
```

В процессе распределенных вычислений на каждом ядре используются потоки независимых псевдослучайных чисел, получаемые в результате работы подпрограммы, реализующей параллельный генератор (см. (2)). В процедуре realization библиотечный параллельный генератор вызывается следующим образом (см. п. **3**):

```
a = rnd128();
```

Здесь a — очередное псевдослучайное число, равномерно распределенное в интервале от нуля до единицы. Инициализация параллельного генератора выполняется автоматически при запуске программы, скомпилированной и собранной с помощью библиотеки PARMONC.

Таким образом, библиотечные подпрограммы применяются без явного использования команд MPI. Как отмечено в п. **1**, число используемых в целях моделирования в PARMONC вычислительных ядер практически не ограничено и зависит только от применяемой вычислительной системы.

Для достижения равномерной загрузки ядер распределенное статистическое моделирование организовано следующим образом. Каждое ядро (например, с номером $m = 0, 1, \ldots, M - 1$) независимо моделирует реализации $\zeta^{(1)}, \zeta^{(2)}, \ldots$ и периодически, допустим, через каждые LS реализаций, дает команду на отправку вычисленных выборочных средних на

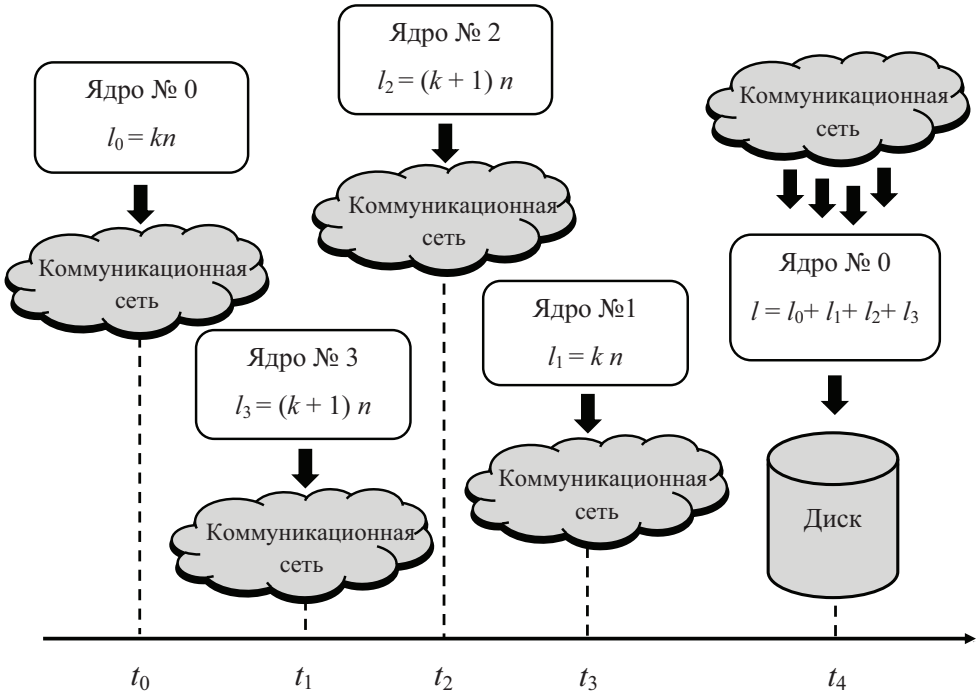

Рис. 2. Организация вычислений в библиотеке PARMONC для числа ядер $M = 4$

выделенное ядро (с номером 0 для определенности). В свою очередь выделенное ядро периодически, например через каждые LR реализаций, дает команду на получение отправленных с других ядер значений, осредняет их по формуле (1) и сохраняет на диске. Отметим, что отправка данных со всех ядер на выделенное ядро и получение этим ядром отправленных ему данных происходят в асинхронном режиме. Значения периодов отправки LS и приемки LR задаются в числе параметров процедуры parmoncc:

```
parmoncc( realization, L, SUBT, LR, LS ).
```

При такой организации вычислений параметр L целесообразно задавать достаточно большим, а параметры отправки LS и приемки LR определять таким образом, чтобы результаты осреднения на выделенном ядре обновлялись на диске через удобные для пользователя промежутки времени. Такая организация вычислений позволяет пользователю контролировать погрешность моделирования. На рис. 2 схематически представлена организация вычислений для числа ядер $M = 4$. По горизонтальной оси отложены затраты машинного времени. В прямоугольниках, обозначающих вычислительные ядра, указывается число реализаций, полученных на данном процессоре к моменту отправки (приема).

Как показали численные эксперименты с использованием PARMONC, при использовании большого числа ядер, при больших объемах пересылаемых данных и при большой частоте отправки (приема) асинхронная процедура передачи данных в процессе счета практически не влияет на эффективность распараллеливания, величина которой составляет почти 100 % [11].

В процессе счета происходит автоматическое вычисление выборочных средних и пределов погрешностей для статистических оценок, алгоритм моделирования которых задается в моделирующей подпрограмме; результаты вычислений периодически сохраняются на диске

в удобном для дальнейшей обработки виде. С помощью библиотеки PARMONC нетрудно продолжить ранее проводившиеся расчеты с автоматическим учетом их результатов. Также с помощью библиотеки можно получать коррелированные статистические оценки различных функционалов [8, 9].

3. Особенности реализации библиотеки PARMONC на кластере ССКЦ КП СО РАН. Оборудование кластера НКС-30Т ССКЦ КП СО РАН позволяет выполнять длительные по времени расчеты [10]. Для моделирования реализаций в соответствии с методологией распределенного статистического моделирования на вычислительных ядрах доступна оперативная память объемом от 2 до 8 Гб. Коммуникационная сеть кластера позволяет пересылать между вычислительными ядрами данные большого объема (данные соответствуют рассчитанным на ядрах выборочным значениям). С целью апробации библиотеки моделирование реализаций распределялось на все ядра кластера, общее число которых составляет около 3000. Таким образом, на кластере НКС-30Т можно эффективно проводить "большие" расчеты с использованием метода Монте-Карло.

На кластере НКС-30Т библиотека PARMONC установлена в директории /ifs/apps /parmonc/ и состоит из следующих основных подпрограмм и исполняемых файлов [8, 9]:

— rnd128 — функция для получения одного случайного числа, равномерно распределенного в интервале от 0 до 1, с помощью параллельного генератора случайных чисел (см. (2));

— parmoncf — процедура, осуществляющая распределенное статистическое моделирование (для программ на языке Fortran-e);

— parmoncc — процедура, осуществляющая распределенное статистическое моделирование (для программ на языке C);

— manaver — программа для осреднения выборочных средних, рассчитанных независимо на разных ядрах;

— genparam — программа для расчета параметров параллельного генератора случайных чисел.

Здесь rnd128, parmoncf и parmoncc — библиотечные подпрограммы для использования в пользовательских программах на языках Fortran или C; genparam и manaver — исполняемые файлы для запуска из командной строки. Объектные файлы PARMONC упакованы в статическую библиотеку libparmonc.a. Пользователь вставляет вызовы процедур rnd128 и parmoncf/parmoncc в свои программы. Главная пользовательская программа, в которой находится вызов parmoncf/parmoncc, рассматривается компилятором как MPI-программа, несмотря на то что в самой пользовательской программе отсутствуют явные вызовы директив MPI. Это означает, что такая программа должна компилироваться и собираться с использованием команд mpiicc или mpiifort. Результаты расчетов, выполненных с использованием процедур parmoncf/parmoncc, сохраняются в файлах, которые находятся в специальной поддиректории рабочей директории пользователя. Все эти файлы обновляются каждый раз, когда выделенное ядро (например, нулевое) получает данные с других ядер, осредняет их и сохраняет на диске.

В файл .bashrc, который находится в домашней директории пользователя, необходимо добавить следующую строку:

```
source /ifs/apps/parmonc/bin/parmoncvars.sh
```

Тем самым объявляются три переменные окружения

```
$PRMCBIN, $PRMCLIB, $PRMCINC,
```

которые используются при компиляции и сборке приложений с помощью PARMONC, а также при запуске команд. Для компиляции и сборки пользовательских программ с использованием библиотеки PARMONC следует использовать команду (приведен пример для программы на языке C)

```
mpiicc -o test -LPRMCLIB -IPRMCINC test.c -lparmonc.
```

Здесь test — имя исполняемого файла; test.c — пользовательская программа.

В заключение укажем некоторые направления дальнейшего развития библиотеки PARMONC. Библиотеку можно сделать базовым программным уровнем для масштабируемых параллельных приложений метода Монте-Карло, реализующих сложные вероятностные модели естествознания [3]. Кроме того, целесообразно адаптировать библиотеку к современным высокопроизводительным кластерам с гибридной архитектурой.

Список литературы

1. Глинский Б. М., Родионов А. С., Марченко М. А. и др. Агентно-ориентированный подход к имитационному моделированию суперЭВМ экзафлопсной производительности в приложении к распределенному статистическому моделированию // Вестн. ЮУрГУ. Сер. Мат. моделирование и программирование. 2012. № 18. Вып. 12.

2. Glassman I. Combustion. 4th ed. / I. Glassman, R. Yetter. Salt Lake City: Acad. Press, 2008.

3. Михайлов Г. А. Численное статистическое моделирование. Методы Монте-Карло / Г. А. Михайлов, А. В. Войтишек. М.: Академия, 2006.

4. Марченко М. А., Михайлов Г. А. Распределенные вычисления по методу Монте-Карло // Автоматика и телемеханика. 2007. Вып. 5. С. 157–170.

5. Marchenko M. A. Majorant frequency principle for an approximate solution of a nonlinear spatially inhomogeneous coagulation equation by the Monte Carlo method // Russ. J. Numer. Anal. Math. Modelling. 2008. V. 21, N 3. P. 199–218.

6. Marchenko M. A., Mikhailov G. A. Parallel realization of statistical simulation and random number generators // Russ. J. Numer. Anal. Math. Modelling. 2002. V. 17, N 1. P. 113–124.

7. Marchenko M. A. Parallel pseudorandom number generator for large-scale Monte Carlo simulations // Lecture Notes Comput. Sci. 2007. V. 4671. P. 276–282.

8. Страница библиотеки PARMONC на сайте ССКЦ КП СО РАН. [Электрон. ресурс]. Режим доступа: http://www2.sscc.ru/SORAN-INTEL/paper/2011/parmonc.htm.

9. Документация к библиотеке PARMONC на сайте ССКЦ КП СО РАН. [Электрон. ресурс]. Режим доступа: http://www2.sscc.ru/SORAN-INTEL/paper/2011/parmonc.pdf.

10. Marchenko M. PARMONC — a software library for massively parallel stochastic simulation // Lecture Notes Comput. Sci. 2011. V. 6873. P. 302–315.

11. Описание кластера НКС-30Т на сайте ССКЦ КП СО РАН. [Электрон. ресурс]. Режим доступа: http://www2.sscc.ru/НКС-30Т/НКС-30Т.htm.

Марченко Михаил Александрович — канд. физ.-мат. наук, учен. секретарь Института вычислительной математики и математической геофизики СО РАН; доц. Новосибирского государственного университета; e-mail: mam@osmf.sscc.ru

Дата поступления– 06.03.13