УДК 519.6

БИБЛИОТЕКА PARMONC ДЛЯ РЕШЕНИЯ «БОЛЬШИХ» ЗАДАЧ ПО МЕТОДУ МОНТЕ-КАРЛО*

М.А. Марченко

Институт вычислительной математики и математической геофизики СО РАН, Новосибирск;
Новосибирский госуниверситет

mam@osmf.sscc.ru

Поступила в редакцию 10.09.2012

Представлена библиотека PARMONC (сокращение от «PARallel MONte Carlo»), предназначенная для эффективного распараллеливания широкого круга приложений метода Монте-Карло, обладающих большой вычислительной трудоемкостью. 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 производится достаточно просто. Библиотека PARMONC позволяет масштабировать вычисления на практически ограниченное 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.

Введение

В настоящее время ведущими специалистами по вычислительной математике высказывается убеждение в том, что в ближайшем будущем в области компьютерного моделирования будут широко использоваться вероятностные имитационные модели и методы Монте-Карло (методы численного статистического моделирования). С одной стороны, это убеждение основано на том, что вероятностные имитационные модели дают адекватное описание физических, химических, биологических и др. явлений при их рассмотрении «из первых принципов». С другой стороны, алгоритмы метода Монте-Карло, реализующие вероятностные модели, допускают возможность эффективного распараллеливания. Использование методов Монте-Карло весьма перспективно в связи с вероятным появлением в ближайшем будущем суперЭВМ экзафлопсного уровня [1].

Упомянем одно из важных применений методов статистического моделирования. С течением химически реагирующих газов часто приходится сталкиваться при рассмотрении физических явлений, связанных с повышением температуры газовых смесей выше критической. К таким явлениям относятся процессы горения газовых топлив в камерах сгорания, соплах и горелках различных геометрических форм. Численное моделирование на компьютерах течений химически реагирующих газов по методу Монте-Карло является весьма трудоёмким, поскольку приходится учитывать многие физические и химические особенности протекающих при горении процессов. Как показывает анализ литературы, в последние десятилетия понимание химии горения (по крайней мере, с участием небольших молекул) и возможности моделирования процессов горения на больших компьютерах достигли такого уровня, который обеспечивает необходимую надежность результатов [2]. Численное моделирование процессов горения требует значительных вычислительных ресурсов и относится к числу задач, для которых необходимо использование суперЭВМ. Такие вычислительные потребности обусловлены большим средним временем расчета каждой реализации ансамбля тестовых частиц, большим числом независимых реализаций ансамбля, которое определяется из необходимой точности расчетов, и большим объемом используемой машинной памяти. Для таких задач применение описываемой в настоящей работе библиотеки

*Статья рекомендована к публикации программным комитетом Международной научной конференции «Параллельные вычислительные технологии 2012».

PARMONC позволит существенно сократить трудоемкость расчетов.

Распределенное статистическое моделирование

Под численным статистическим моделированием обычно понимают реализацию с помощью компьютера вероятностной модели ζ некоторого объекта с целью оценивания изучаемых интегральных характеристик φ на основе закона больших чисел [3]:

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

При этом чем больше объем выборки L, составленной из смоделированных независимых реализаций $\zeta^{(1)},\zeta^{(2)},...,\zeta^{(L)}$, тем выше точность оценивания, причем статистическая погрешность убывает обратно пропорционально квадратному корню от объема выборки L.

Трудоемкость статистического моделирования (т.е. затраты машинного времени, необходимые для достижения заданного уровня статистической погрешности ε) определяется величиной $\tau_\zeta L$, которая, в свою очередь, пропорциональна величине $\tau_\zeta D\zeta\varepsilon^{-2}$ [3]. Здесь τ_ζ – среднее время моделирования одной реализации случайной величины ζ, $D\zeta$ – дисперсия случайной величины ζ. «Большие» задачи статистического моделирования характеризуются либо большой величиной τ_ζ, либо малой величиной ε, либо большой величиной $D\zeta$, либо же сочетанием этих факторов. Кроме того, «большие» задачи статистического моделирования часто требуют больших объемов оперативной памяти.

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

Следует упомянуть 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 (реализацией здесь является ансамбль тестовых частиц) и требуется применять методы пространственной декомпозиции ансамбля. При этом можно применять способы динамической балансировки вычислительной нагрузки, основанные на специальных формулах прогноза времени вычислений для каждого вычислительного ядра [5].

Метод *распределенного статистического моделирования* состоит в распределении моделирования независимых реализаций по вычислительным ядрам с периодическим осреднением полученных выборочных значений по статистически эффективной формуле

$$\overline{\zeta} = \left[\sum_{m=1}^{M} l_m\right]^{-1}\sum_{m=1}^{M} l_m\overline{\zeta}_m, \qquad (1)$$

где M – общее число ядер, l_m – объем выборки, полученной на m-м ядре, $\overline{\zeta}_m$ – соответствующее m-му ядру выборочное среднее значение. Очевидно, что главным критерием осуществимости такой параллельной реализации является возможность «поместить» данные вычислительной программы для моделирования реализаций в оперативную память каждого ядра. Подчеркнем, что в целях 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 одинаковую производительность [4]. Отметим, что возможные отказы компонентов вычислительной системы и связанные с этим потери данных компенсируются моделированием реализаций на других вычислительных узлах [4].

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

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

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

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

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. А именно, для решения «больших» задач по методу Монте-Карло предлагается использовать генератор следующего вида [4, 6, 7]:

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

где

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

Последовательность псевдослучайных чисел $\{\alpha_n\}$ является периодической, длина ее периода равна $2^{126} \approx 10^{38}$.

Для распределения псевдослучайных чисел между разными вычислительными ядрами предлагается следующий порядок их использования. Последовательность $\{\alpha_n\}$ предварительно разбивается на подпоследовательности длины μ, начинающиеся с чисел $\{\alpha_{m\mu}\}$, m = 0, 1, 2, …, после чего разные подпоследовательности используются на разных ядрах. Значение «прыжка» генератора μ должно выбираться из соображений, чтобы μ псевдослучайных чисел хватало для моделирования на каждом ядре. Легко показать, что для метода вычетов начальные значения $\{\alpha_{m\mu}\}$ указанных подпоследовательностей получаются по формуле

$$u_{(m+1)\mu} = u_{m\mu} A_\mu (\mathrm{mod}\, 2^{128}),$$
$$\alpha_{m\mu} = u_{m\mu} 2^{-128}, \quad m = 0, 1, \ldots \qquad (3)$$
$$A_\mu \equiv A^\mu (\mathrm{mod}\, 2^{128}).$$

Отметим, что длина периода генератора позволяет независимым образом распределять псевдослучайные числа по реализациям на практически неограниченное число вычислительных ядер. Параллельный генератор (3) успешно используется в ряде институтов СО РАН на протяжении последних десяти лет.

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 в широкое использование программная библиотека *PARMONC* (сокращение от *PARallel MONte Carlo*). Область применения библиотеки: «большие» задачи статистического моделирования в естественных и гуманитарных науках (физика, химия, биология, медицина, экономика и финансы, социология и др.). Библиотека *PARMONC* установлена на кластерах Сибирского суперкомпьютерного центра (ССКЦ КП СО РАН) и может использоваться на вычислительных системах с аналогичной архитектурой. При этом использование библиотеки не привязано к каким-то определенным компиляторам языков C и FORTRAN или MPI. Инструкции по использованию библиотеки с примерами можно найти по ссылкам [8, 9].

Возможность применения библиотеки *PARMONC* определяется «естественной» крупноблочной фрагментированностью программ статистического моделирования. Общая структура такого рода программ, в упрощенном виде, следующая (в нотации языка C++):

```
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 по выходу 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 – 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 поясняется 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 об-

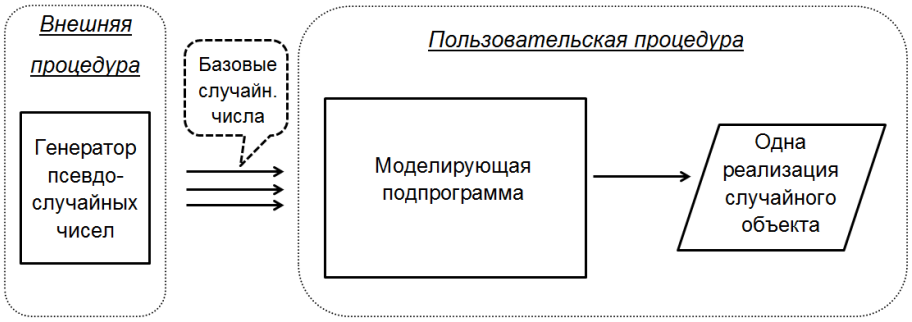

Рис. 1. Взаимосвязь программных объектов при статистическом моделировании

щее число независимых реализаций передаются в библиотечную процедуру *parmoncc* в качестве входных аргументов; выборочное среднее будет возвращаться в переменную *SUBT*; для простоты остальные аргументы процедуры *parmoncc* пока 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 (3). В процедуре *realization* библиотечный параллельный генератор вызывается следующим образом (см. следующий раздел):

$$a = \text{rnd128}();$$

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

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

Для достижения равномерной загрузки ядер распределенное статистическое моделирование организовано следующим образом. Каждое ядро (например, с номером m, где $m = 0, 1, 2, \ldots, M - 1$) независимо моделирует реализации $\zeta^{(1)}, \zeta^{(2)}, \ldots$ и периодически, скажем, через каждые LS реализаций, дает команду на отправку вычисленных выборочных средних $\overline{\zeta}_m$ на выделенное ядро (с номером 0, для определенности). В свою очередь, 0-е ядро периодически, скажем, через каждые LR реализаций, дает команду на получение отправленных с других ядер значений, осредняет их по формуле (1) и сохраняет на диск. Отметим, что отправка данных со всех ядер на 0-е ядро и получение 0-м ядром отправленных ему данных происходит 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 так, чтобы результаты осреднения на 0-м процессоре обновлялись на диске через удобные для пользователя промежутки времени. Такая организация вычислений позволяет пользователю контролировать погрешность моделирования. Организация вычислений схематически представлена на рис. 2 для числа ядер $M = 4$.

Как показали численные эксперименты с использованием *PARMONC*, при использовании большого числа ядер, при больших объемах пересылаемых данных и при большой частоте отправки/приемки 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 границ погрешностей для статистических оценок, алго-

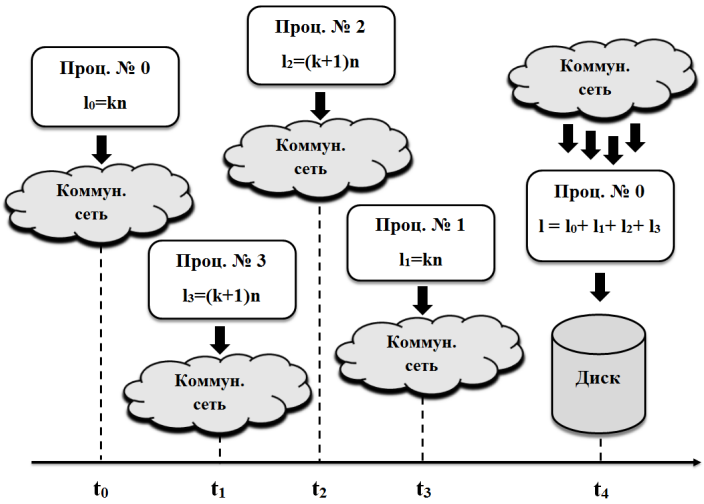

Рис. 2. Организация вычислений в библиотеке PARMONC для числа ядер $M = 4$. По горизонтальной оси отложено машинное время. В прямоугольниках, обозначающих вычислительные ядра, указывается число реализаций, полученных на данном процессоре к моменту отправки/приемки

ритм моделирования которых задается в моделирующей подпрограмме; результаты вычислений периодически сохраняются на диске в удобном для дальнейшей обработки виде. С помощью библиотеки *PARMONC* можно легко организовать продолжение ранее проведенных расчетов с автоматическим учетом их результатов. Также с помощью библиотеки можно получать коррелированные статистические оценки различных функционалов [8, 9].

Особенности реализации библиотеки PARMONC на кластере ССКЦ КП СО РАН

Оборудование кластера НКС-30Т ССКЦ КП СО РАН позволяет производить длительные по времени расчеты [10]. Для моделирования реализаций в соответствии с методологией распределенного статистического моделирования на каждом из вычислительных ядер доступно 2 Гб оперативной памяти. 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 2688. Таким образом, на кластере НКС-30Т можно эффективно производить «большие» расчеты по методу Монте-Карло.

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

- *rnd*128 – функция для получения одного случайного числа, равномерно распределенного в интервале от 0 до 1, с помощью параллельного генератора случайных чисел (3);
- *parmoncf* – процедура, осуществляющая распределенное статистическое моделирование (для программ на Фортране);
- *parmoncc* – процедура, осуществляющая распределенное статистическое моделирование (для программ на С);
- *manaver* – программа для осреднения выборочных средних, независимо рассчитанных на разных ядрах;
- *genparam* – программа для расчета параметров параллельного генератора случайных чисел.

Здесь *rnd*128, *parmoncf* и *parmoncc* являются библиотечными подпрограммами для использования в пользовательских программах на Фортране, С или С++, *genparam* и *manaver* являются исполняемыми файлами для запуска из командной строки. Объектные файлы *PARMONC* упакованы в статическую библиотеку *libparmonc.a*.

Пользователь вставляет вызовы процедур *rnd*128 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 нет явных вызовов 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 файлы, которые находятся в специальной поддиректории в рабочей директории пользователя. Все эти фай-

лы обновляются всякий раз, когда выделенное ядро (например, 0-е, для определенности) получает данные с других ядер, осредняет их и сохраняет на диск.

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. Эти переменные 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 следующую команду (приведен пример для программы на C): mpiicc -o test -LPRMCLIB−IPRMCINC test.c -lparmonc, где test – имя исполняемого файла, test.c – пользовательская программа.

Заключение

В заключение упомянем 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 [10].

Работа над статьей проводилась в рамках реализации федеральной целевой программы «Исследования и разработки по приоритетным направлениям развития научно-технологического комплекса России на 2007–2013 гг.», государственный контракт 07.514.11.4016, а также при финансовой поддержке грантов РФФИ №№ 10-07-00454, 12-01-00034, 12-01-00727, 12-07-00499; МИП №39 СО РАН, МИП № 47 СО РАН 47, МИП № 126 СО РАН, МИП № 130 СО РАН.

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 // Вестник ЮУрГУ. Серия Математическое моделирование и программмирование. 2012 (принято в печать в №18 (277). Вып. 12).

2. Glassman I., Yetter R. Combustion. 4 ed. Academic Press, 2008. 800 p.

3. Михайлов Г.А., Войтишек А.В. Численное статистическое моделирование. Методы Монте-Карло. М.: Академия, 2006. 368 с.

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. №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. №1. P. 113–124.

7. Marchenko M.A. Parallel Pseudorandom Number Generator for Large-scale Monte Carlo Simulations // LNCS. 2007. V. 4671. P. 276–282.

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

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

10. Описание кластера НКС-30Т на сайте ССКЦ КП СО РАН [Электронный ресурс]. – Режим доступа: http://www2.sscc.ru/HKC-30T/HKC-30T.htm.

11. Marchenko M. PARMONC – A Software Library for Massively Parallel Stochastic Simulation // LNCS. 2011. V. 6873. P. 302–315.

PROGRAM LIBRARY PARMONC FOR SOLVING «LARGE» PROBLEMS BY THE MONTE CARLO METHOD

M.A. Marchenko

The article presents the software library PARMONC (an acronym of the PARallel MONte Carlo) developed for effective parallelization of a wide range of Monte Carlo method applications with the large computational complexity. The library uses native coarse-grain parallelism of the Monte Carlo method. The «core» of the library is a well-tested, fast and reliable long-period parallel pseudorandom number generator. The PARMONC is an easy-to-use tool to perform distributed stochastic simulation. It does not require any knowledge of the MPI language since the parallelization of complicated sequential programs is carried out in an easy way. The PARMONC enables one to scale Monte Carlo simulations to a practically infinite number of processors, the computational load being evenly distributed across all cores.

Keywords: stochastic simulation, Monte Carlo method, pseudorandom number generators, distributed computing, program libraries.