Жидкие кристаллы и их практическое использование. 2014. Т. 14, № 4. С. 91–96

УДК 004.942:[51-72:530.145]

И. А. Обжерин[1], Ф. Н. Ясинский[1], В. В. Соцкий[2]

ПРИМЕНЕНИЕ МЕТОДОВ МОНТЕ-КАРЛО В РАСПАРАЛЛЕЛЕННЫХ ВЫЧИСЛЕНИЯХ МОЛЕКУЛЯРНЫХ СТРУКТУР МЕТОДОМ ХАРТРИ – ФОКА

[1] Ивановский государственный энергетический университет им. В. И. Ленина,
ул. Рабфаковская, 34, 153003 Иваново, Россия. E-mail: ivan.obzherin@gmail.com
[2] НИИ Наноматериалов, Ивановский государственный университет,
ул. Ермака, 39, 153025 Иваново, Россия. E-mail: sotsky2005@yandex.ru

В методах молекулярных орбиталей наиболее трудоемкой частью расчетов является численный расчет большого количества кратных интегралов. Повышения скорости вычислений можно добиться, используя приближенные методы расчетов, поддающиеся распараллеливанию. Описанный подход опирается на методы Хартри – Фока, Монте-Карло, а также средства распараллеливания вычислений nVIDIA CUDA. Описаны расчеты по методу Хартри – Фока, а также в рамках этого метода подходы к ускорению вычислений с использованием методов Монте-Карло в сочетании с приемами уменьшения дисперсии. Рассмотрены схемы распараллеливания описанных методов и приемов. Описанный подход позволяет создать приложение для оценочных расчетов молекулярных структур без применения сложных программных пакетов, а также получить дополнительное повышение производительности, используя возможности распараллеливания на общедоступном оборудовании с относительно невысокой стоимостью.

Ключевые слова: *методы молекулярных орбиталей, метод Хартри – Фока, методы Монте-Карло, параллельные вычисления, CUDA.*

I. A. Obzherin, F. N. Yasinsky, V. V. Sotsky

APPLYING MONTE-CARLO METHODS TO HARTREE – FOCK METHOD IN PARALLEL COMPUTING OF MOLECULAR STRUCTURES

[1] Ivanovo State Power University,
Rabfakovskaya str., 34, 153003 Ivanovo, Russia. E-mail: ivan.obzherin@gmail.com
[2] Nanomaterials Research Institute, Ivanovo State University,
Ermak str., 39, 153025 Ivanovo, Russia. E-mail: sotsky2005@yandex.ru

The most time consuming part of computations with molecular orbitals is calculation of numerous multiple integrals. It is possible to enhance computation speed using approximate problem solving methods that can be used in parallel computing. The described approach is based on Hartree – Fock method, Monte-Carlo method and NVIDIA CUDA parallel computing tools. The calculations with Hartree – Fock method are described, as well as approaches to enhancing computational speed using Monte-Carlo method combined with dispersion reduction techniques. The parallel computation of the aforementioned methods and techniques is outlined. The described approach allows to create software for assessing calculations of molecular structures without the help of complex software packages and to enhance the computation speed using parallel computing abilities of wide-spread and relatively cheap hardware.

Key words: *molecular orbitals, Hartree – Fock method, Monte-Carlo method, parallel computing, CUDA.*

Введение

Жидкие кристаллы (ЖК) находят широкое применение в различных областях деятельности человека, таких как электроника, медицина, хроматография. Поэтому получение ЖК-соединений, обладающих необходимым набором свойств, является актуальной задачей. Экспериментальный путь получения соединений с нужными свойствами чрезвычайно затратен как по времени, так и по стоимости. Способом решения данной проблемы является использование математического моделирования, которое позволяет предсказывать свойства молекул и существенно экономить время эксперимента. Одним из методов, позволяющих моделировать свойства ЖК-молекул, является метод Хартри – Фока [1]. В то же время молекулы веществ, проявляющих мезоморфные свойства, состоят из большого числа атомов (порядка 40–1000 атомов на молекулу), что делает моделирование ЖК-соединений затратным с точки зрения вычислений. В связи с этим целесообразно внедрение в вычислительный эксперимент параллельных (высокопроизводительных) вычислений, способных значительно сократить время, затрачиваемое на расчеты. Графические устройства, обладая большим количеством параллельных процессоров, в настоящее время являются наиболее мощными ускорителями вычислений. В то же время использование технологий параллельных вычислений (особенно GPU, ввиду особенностей их архитектуры) для ускорения вычислительных экспериментов ставит задачу разработки или адаптации существующих алгоритмов с целью получения максимальной производительности.

Постановка задачи

Метод Хартри – Фока относится к методам молекулярных орбиталей и предназначен для приближенного вычисления энергий электронов, принадлежащих молекулярным орбиталям. В рамках метода решается матричное уравнение вида:

$$FC = \varepsilon S C, \tag{1}$$

где известны F и S, ε ищутся как собственные числа, а C – матрица значений собственных векторов.

Уравнение решается итеративно, причем на каждой k-ой итерации:

$$F_{ij} = H_{ij} + \sum_{i_1, j_1} D_{i_1 j_1} (2I_{iji_1 j_1} - I_{ii_1 jj_1}), \tag{2}$$

$$D_{i_1 j_1}^{<k>} = \sum_{l=1}^{n/2} C_{i_1 l}^{<k-1>} C_{j_1 l}^{<k-1>}, \tag{3}$$

$$D_{ij}^{<0>} = \delta_{ij} \begin{cases} 1, i \le N/2, j \le N/2, \\ 0, i > N/2, j > N/2. \end{cases}$$

Здесь и далее S – интеграл перекрывания, характеризующий энергию связи через волновые функции электронов молекулы:

$$S_{ij} = \int \chi_i(\vec{r}) \chi_j(\vec{r}) d\vec{r}. \tag{4}$$

H – интеграл суммы кинетической и потенциальной энергий электронов в поле атома:

$$H_{ij} = \int \chi_i(\vec{r}) \cdot \left[-\frac{\Delta^2}{2} - \sum_a \frac{Z_a}{\left| \vec{r} - \vec{R}_a \right|} \right] \cdot \chi_j(\vec{r}) d\vec{r}. \tag{5}$$

I – интеграл обменного взаимодействия между электронами:

$$I_{iji_1 j_1} = \int \chi_i(\vec{r}_1) \chi_j(\vec{r}_1) \cdot \frac{1}{\left| \vec{r}_{12} \right|} \cdot \chi_{i_1}(\vec{r}_2) \chi_{j_1}(\vec{r}_2) d\vec{r}_1 d\vec{r}_2. \tag{6}$$

Базисные функции χ_i подбираются таким образом, чтобы наилучшим образом аппроксимировать поведение электронов атомных орбиталей. В частности, базисные функции Слейтера можно представить в виде

$$\chi_i = c_i e^{-\alpha_i r}. \tag{7}$$

Такое описание имеет ряд существенных недостатков. Интегралы на его основе не поддаются аналитическому решению, а численное решение интегралов является чрезвычайно трудоемким. Кроме того, они хорошо описывают хартри-фоковские атомные орбитали только в отдельных диапазонах расстояний от ядра (либо вблизи от него, либо на удалении). Для устранения этого недостатка можно использовать в одной базисной функции несколько орбитальных экспонент (дубль-зета- или DZ-функции), но это еще больше усложняет расчеты.

В качестве альтернативы были рассмотрены функции Гаусса вида:

$$\chi_i = c_i e^{-\alpha_i r^2}. \quad (8)$$

Они поддаются аналитическому интегрированию, но плохо описывают поведение электрона вблизи ядра и слишком сильно убывают вдали от него. В качестве альтернативы можно использовать комбинации функций, которые могут дать лучшее приближение:

$$\chi_i = \sum_{k=1}^{n} c_{ik} e^{-\alpha_{ik} r^2}. \quad (9)$$

Сравнение базисных функций гауссова и слейтерова типа показано на рис. 1.

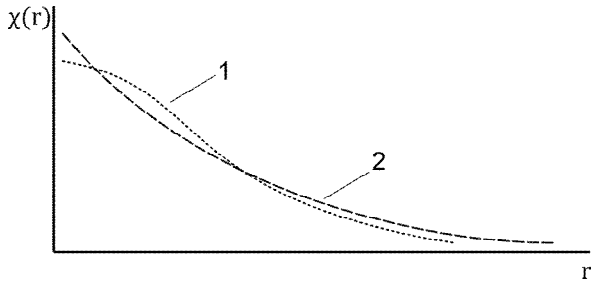

Рис. 1. Схематичное сравнение базисных функций: 1 – гауссова типа, 2 – слейтерова типа

Расчет интегралов численными методам является самой трудоемкой частью алгоритма Хартри - Фока. Во-первых, интегралы (4) и (5) являются трехкратным (очевидно, потому что вектор *r* в трехмерном пространстве определяется 3 координатами), а (6) – 6-кратным. Во-вторых, число интегралов стремительно растет с ростом числа орбиталей молекулы *N*, как зависимость порядка O([N/2]^4) – даже с учетом симметрии *I*, *S* и *H*. Исходя из этого, проблема решения интегралов в рамках метода Хартри – Фока является наиболее интересной с точки зрения оптимизации и распараллеливания вычислений.

Применение методов Монте-Карло к методу Хартри – Фока

Для упрощения расчетов интегралов можно применить методы Монте-Карло, основным приемом которых является сведение задачи вычисления некоторый величины *y* к расчету математических ожиданий *M* случайной величины: $y \approx M\xi$. Тогда для расчета интеграла вида

$$I = \int_V f(x)dx, \quad (10)$$

где *f(x)* – интегрируемая функция, $x \in V$, введем случайную величину *Q* с равномерным распределением по *V*. Тогда *f(Q)* = *Z*, а значение интеграла можно оценить как

$$\theta_N \approx (1/N)\sum_{i=1}^{N} Z_i = (1/N)\sum_{i=1}^{N} f(Q_i). \quad (11)$$

Если для выбранной случайной величины *Z* помимо математического ожидания *MZ* для выбранной случайной величины также определена и дисперсия *DZ*, ее можно использовать для оценки погрешности реализации метода Монте-Карло:

$$DZ = M(Z^2) - M(Z)^2$$

$$DZ \approx (1/N)\sum_{i=1}^{N} Z_i^2 - \left[(1/N)\sum_{i=1}^{N} Z_i\right]^2. \quad (12)$$

Основное преимущество метода Монте-Карло над простым способом численного интегрирования методом сеток в значительном сокращении числа узлов *N*, которые используются в вычислении [3, 4]. Для метода сеток это число жестко определено исходя из шага по каждой из осей координатной сетки как $N_G = N_x \cdot N_y \cdot N_z$. Метод Монте-Карло позволяет уменьшать число N_{MK}, увеличивая погрешность оценки значения интеграла. В некоторых случаях увеличение NMK уменьшает дисперсию слишком медленно, тогда $N_{MK} \to N_G$, сокращая преимущество метода Монте-Карло до минимума.

Дисперсию можно уменьшить с помощью ряда приемов. В частности, рассмотрим прием выделения главной части. Если часть задачи можно решить аналитически, то будем вычислять методом Монте-Карло не всю задачу, а лишь разницу между всей задачей и выделенной главной частью.

Пусть существует функция *h(x)*, достаточно близкая к *f(x)*, такая что

$$\int_V h(x)dx = C. \quad (13)$$

Тогда используется следующая оценка интеграла:

$$\theta'_N = C + (1/N)\sum_{i=1}^{N}\left[f(Q_i) - h(Q_i)\right]. \quad (14)$$

Дисперсию *DZ'*, где *Z'* = *C* + *f(Q)* – *h(Q)*, в этом случае можно оценить следующим образом:

$$DZ' \le \int_V \left[f(x) - h(x)\right]^2 dx. \quad (15)$$

В идеальном случае можно подобрать такую *h(x)*, интеграл которой можно рассчитать аналитически.

Применительно к задаче решения интегралов в методе Хартри – Фока в качестве целевой функции *f(x)* можно взять подынтегральные выражения на основе базисных функций слейтеровского типа. Тогда в качестве вспомогательной функции *h(x)* используем подынтегральные функции на основе гауссовых функций, для которых можно найти аналитическое решение.

Аналитическое решение интеграла S_{AB} при использовании гауссовых функций будет выглядеть следующим образом:

$$\chi_i(r) = c_i e^{-\alpha_i r^2},$$
$$S_{ij} = \int \chi_j(r)\chi_i(r)dr = \left(\frac{2\sqrt{\alpha_i\alpha_j}}{\alpha_i+\alpha_j}\right)^{\frac{3}{2}} e^{-\frac{\alpha_i\alpha_j}{\alpha_i+\alpha_j}R_{ij}^2}. \quad (16)$$

Величина R_{ij} – расстояние между центрами орбиталей. В качестве центра орбиталей можно взять координаты ядер атомов. Результат вычислений *S* используется в расчетах *H*:

$$H_{ij} = \int \chi_i(\vec{r})\left(-\frac{\Delta}{2}\right)\chi_j(\vec{r})d\vec{r} = \left[\frac{3\alpha_i\alpha_j}{\alpha_i+\alpha_j} - 2\left(\frac{\alpha_i\alpha_j}{\alpha_i+\alpha_j}\right)^2 R_{ij}\right]S_{ij}. \quad (17)$$

Аналитическое решение интеграла *I* также использует значения *S*:

$$I_{iji_1j_1} = \int \chi_i(\vec{r}_1)\chi_j(\vec{r}_1)\cdot\left(\frac{1}{r_{12}}\right)\cdot \chi_{i_1}(\vec{r}_2)\chi_{j_1}(\vec{r}_2)d\vec{r}_1 d\vec{r}_2 = \frac{2}{\sqrt{\pi}}\sqrt{\frac{(\alpha_i+\alpha_j)(\alpha_{i_1}+\alpha_{j_1})}{\alpha_i+\alpha_j+\alpha_{i_1}+\alpha_{j_1}}}\cdot S_{ij}S_{i_1j_1}\cdot G\left(\frac{(\alpha_i+\alpha_j)(\alpha_{i_1}+\alpha_{j_1})}{\alpha_i+\alpha_j+\alpha_{i_1}+\alpha_{j_1}}R_{pq}^2\right). \quad (18)$$

Здесь *G(z)* – функция, определенная следующим образом:

$$G(z) = \frac{\sqrt{\pi}}{2}\cdot\begin{cases}\sum_{i=0}^{5}\beta_i\left(\frac{3\left(1-\sqrt{z}\right)}{3+\sqrt{z}}\right)^i, & \sqrt{z}\le 3{,}093;\\ \frac{1}{\sqrt{z}}, & \sqrt{z}>3{,}093.\end{cases} \quad (19)$$

Расстояние R_{pq} вычисляется для виртуальных центров орбиталей *P* и *Q*. Их координаты рассчитываются с использованием реальных центров:

$$P = \left(\frac{\alpha_i R_{ix}+\alpha_j R_{jx}}{\alpha_i+\alpha_j}, \frac{\alpha_i R_{iy}+\alpha_j R_{jy}}{\alpha_i+\alpha_j}, \frac{\alpha_i R_{iz}+\alpha_j R_{jz}}{\alpha_i+\alpha_j}\right),$$
$$Q = \left(\frac{\alpha_{i_1} R_{i_1x}+\alpha_{j_1} R_{j_1x}}{\alpha_{i_1}+\alpha_{j_1}}, \frac{\alpha_{i_1} R_{i_1y}+\alpha_{j_1} R_{j_1y}}{\alpha_{i_1}+\alpha_{j_1}}, \frac{\alpha_{i_1} R_{i_1z}+\alpha_{j_1} R_{j_1z}}{\alpha_{i_1}+\alpha_{j_1}}\right). \quad (20)$$

Здесь R_{ix} – координата *x* центра *i*-ой орбитали. Коэффициенты β в (19) – набор предопределенных чисел, при их выборе можно руководствоваться значениями из таблицы 1.

Таблица 1. **Значения коэффициентов β**

i	**β_i**	***i***	**β_i**
0	0,84270001	6	–0,07848160
1	0,57015786	7	–0,02565184
2	–0,16778274	8	0,03788288
3	–0,25060352	9	0,00236288
4	0,10478112	10	–0,00824832
5	0,10125920		

Лучшее приближение можно получить, используя комбинации гауссовых функций [2]. Для получения аналитического решения интегралов раскроем, взаимно перемножим члены комбинаций и представим полученный интеграл суммы в виде суммы интегралов:

$$\chi_i(r) = \sum_{k=1}^{n} c_{ik} e^{-\alpha_{ki} r^2}$$

$$\begin{aligned} S_{ij} &= \int \chi_j(r)\chi_i(r)dr = \\ &= \int \sum_{k=1}^{n} c_{jk} e^{-\alpha_{jk}(\overrightarrow{r-R_j})^2} \sum_{l=1}^{n} c_{il} e^{-\alpha_{il}(\overrightarrow{r-R_i})^2} = \\ &= \sum_{k=1}^{n}\sum_{l=1}^{n} \left(\frac{2\sqrt{\alpha_{il}\alpha_{jk}}}{\alpha_{il}+\alpha_{jk}} \right)^{\frac{3}{2}} e^{-\frac{\alpha_{il}\alpha_{jk}}{\alpha_{il}+\alpha_{jk}} R_{ij}^2}. \end{aligned} \quad (21)$$

$$\begin{aligned} H_{ij} &= \int \chi_i(\vec{r}) \left(-\frac{\Delta}{2} \right) \chi_j(\vec{r}) d\vec{r} = \\ &= S_{ij} R_{ij} \sum_{k=1}^{n}\sum_{l=1}^{n} \left[\frac{3\alpha_{il}\alpha_{jk}}{\alpha_{il}+\alpha_{jk}} - 2\left(\frac{\alpha_{il}\alpha_{jk}}{\alpha_{il}+\alpha_{jk}} \right)^2 \right]. \end{aligned} \quad (22)$$

$$\begin{aligned} I_{iji_1j_1} &= \int \chi_i(\vec{r}_1)\chi_j(\vec{r}_1) \left(\frac{1}{r_{12}} \right) \cdot \\ &\cdot \chi_{i_1}(\vec{r}_2)\chi_{j_1}(\vec{r}_2) d\vec{r} = \\ &= \frac{2}{\sqrt{\pi}} S_{ij} S_{i_1j_1} \cdot \\ &\cdot \sum_{k=1}^{n}\sum_{l=1}^{n}\sum_{k_1=1}^{n}\sum_{l_1=1}^{n} \left[\sqrt{F_{klk_1l_1}} G\left(F_{klk_1l_1} R_{pq}^2 \right) \right], \\ F_{klk_1l_1} &= \frac{(\alpha_{ik}+\alpha_{jl})(\alpha_{i_1k_1}+\alpha_{j_1l_1})}{\alpha_{ik}+\alpha_{jl}+\alpha_{i_1k_1}+\alpha_{j_1l_1}}. \end{aligned} \quad (23)$$

Наконец, для каждого слагаемого (23) координаты P и Q вычисляются как

$$P = \begin{pmatrix} \dfrac{\alpha_{ik}R_{ix}+\alpha_{jl}R_{jx}}{\alpha_{ik}+\alpha_{jl}}, & \dfrac{\alpha_{ik}R_{iy}+\alpha_{jl}R_{jy}}{\alpha_{ik}+\alpha_{jl}}, \\ \dfrac{\alpha_{ik}R_{iz}+\alpha_{jl}R_{jz}}{\alpha_{ik}+\alpha_{jl}} \end{pmatrix}, \quad (24)$$

$$Q = \begin{pmatrix} \dfrac{\alpha_{i_1k_1}R_{i_1x}+\alpha_{j_1l_1}R_{j_1x}}{\alpha_{i_1k_1}+\alpha_{j_1l_1}}, & \dfrac{\alpha_{i_1k_1}R_{i_1y}+\alpha_{j_1l_1}R_{j_1y}}{\alpha_{i_1k_1}+\alpha_{j_1l_1}}, \\ \dfrac{\alpha_{i_1k_1}R_{i_1z}+\alpha_{j_1l_1}R_{j_1z}}{\alpha_{i_1k_1}+\alpha_{j_1l_1}} \end{pmatrix} \quad (24)$$

Преимущество метода Монте-Карло также в том, что он просто распараллеливается. Все итерации метода независимы друг от друга, поэтому вычисление Z_i для каждого узла можно передать отдельной нити без каких-либо затрат на синхронизацию их работы (рис. 2). Основной сложностью является получение неповторяющихся наборов случайных чисел для каждой нити. Самым простым и универсальным подходом было бы формировать заранее набор псевдослучайных чисел, используя соответствующие библиотеки используемого языка, а затем дробить его, распределяя между отдельными нитями.

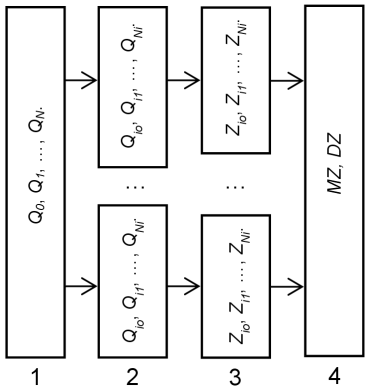

Рис. 2. Схема многопоточных вычислений методом Монте-Карло: 1 – создание набора псевдослучайных чисел; 2 – распределение псевдослучайных чисел между нитями; 3 – расчет $Z = f(Q)$ на соответствующих нитях; 4 – суммирование, вычисление оценок интеграла и дисперсии

Применение технологии CUDA к распараллеливанию вычислений

Некоторые технологии распараллеленных вычислений имеют встроенные инструменты для работы со случайными числами. Так, технология CUDA, использующая для параллельных вычислений устройства с GPU, включает библиотеки с генераторами псевдослучайных чисел с различным распределением [5, 6].

Нас интересует функция получения псевдослучайных чисел с равномерным распределением *curand uniform*. Тем не менее, генератор устроен так, что его состояние для каждой нити должно инициализироваться и храниться в отдельной переменной на всем протяжении работы с генератором.

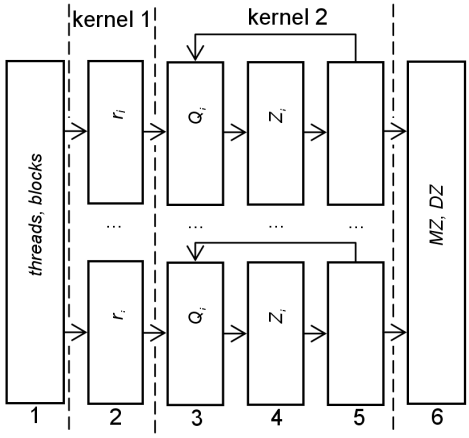

Рис. 3. Схема многопоточных вычислений методом Монте-Карло с использованием CUDA: 1 – определение параметров выполнения многопоточной процедуры CUDA (*kernel*): число требуемых нитей, блоков нитей, разделяемой памяти; 2 – инициализация N состояний r генератора случайных чисел для каждой нити $i = 0..N–1$; 3 – получение случайного числа Q_i, обновление состояния r_i; 4 – расчет $Z_i = f(Q_i)$; 5 – частичное суммирование результатов нитей с одинаковым индексом; 6 – окончательное суммирование, вычисление оценок интеграла и дисперсии.

Нити CUDA, выполняющиеся на GPU, (*thread*) организуются в блоки (*block*), блоки образуют сетку (*grid*). Инструкции для нити определяются внутри *kernel*-функции (рис. 3). Число нитей и блоков, а также при необходимости объем разделяемой нитями памяти указывается в параметрах при вызове *kernel*-функции из процесса выполняемого CPU. Внутри *kernel*-функции текущая нить идентифицируется при помощи специальных переменных *threadIdx*, *blockIdx*, *blockDim*. Соответственно, состояния генератора случайных чисел можно повторно использовать при вычислении на нитях с тем же индексом, но принадлежащих другом блоку. По этим же соображениям можно проводить частичное суммирование результатов вычислений на нитях разных блоков. Таким образом, можно уменьшить время вычислений не только сокращая число операций, выполняемых в однопоточном режиме, но и уменьшив объем передаваемых данных между устройствами CPU и GPU.

Работа поддержана программой Минобрнауки РФ в рамках государственного задания Ивановскому государственному университету для выполнения научно-исследовательских работ, № 4.106.2014К.

Список литературы / References

1. *Заводинский В. Г.* Компьютерное моделирование наночастиц и наносистем. М. : ФИЗМАТЛИТ, 2013. 176 с. [*Zavodinskiy V. G.* Komp'yuternoe modelirovanie nanochastits i nanosistem (Computer simulation of nanoparticles and nanosystems). Moscow : Fizmatlit, 2013. 176 p. (in Russian)].
2. *Жоголев Д. А.* Программы «Орбиталь-1», «Орбиталь-3» для неэмпирических расчетов молекул в базисе линейных комбинаций безузловых гауссовых функций. Киев : Наукова думка, 1974. [*Zhogolev D. A.* Programmy «Orbital-1», «Orbital-3» dlya neempiricheskih raschetov molekul v bazise lineynyh kombinatsiy bezuzlovyh gaussovyh funktsiy (Programs «Orbital-1», «Orbital-3» for non-empiric calculation of molecules in a basis of linear combinations of smooth Gauss functions). Kiev : Naukova Dumka, 1974 (in Russian)].
3. *Соболь И. М.* Численные методы Монте-Карло. М.: Наука, 1973. [*Sobol' I. M.* Chislennye metody Monte-Karlo. (Numerical Monte Carlo methods). M. : Nauka, 1973 (in Russian)].
4. *Соболь И. М.* Метод Монте-Карло. М.: Наука, 1968. [*Sobol' I. M.* Metod Monte-Karlo (Monte Carlo method). M. : Nauka, 1968.
5. NVIDIA CUDA C Programming Guide, Version 4.1, 2011.
6. CUDA C Best Practices Guide, DG-05603-001_v4.1, 2012.

Поступила в редакцию 5.11.2014 г.