

УДК 004.422.81

**РЕШЕНИЕ УРАВНЕНИЙ ЗАДАЧИ ТЕОРИИ ДИНАМИЧЕСКОГО СРЕДНЕГО ПОЛЯ
НА МНОГОПРОЦЕССОРНЫХ СИСТЕМАХ**

С. Н. Исаков¹, В. В. Мазуренко¹

Предложен метод организации распределенного доступа к ресурсам многопроцессорных вычислительных систем при решении задач, связанных с анализом электронной структуры материалов. Работа выполнена при финансовой поддержке гранта “Развитие научного потенциала высшей школы” № 2.1.1/779.

Ключевые слова: теория динамического среднего поля, метод точной диагонализации, одностронние коммуникации MPI.

1. Введение. Открытие тяжелых фермионных соединений и высокотемпературных сверхпроводников вызвало интерес исследователей к сильнокоррелированным системам, в которых кинетическая энергия и энергия межэлектронного взаимодействия имеют один порядок, что приводит к невозможности построить теорию возмущений по какому-либо параметру. Описание многочастичных эффектов в таких системах может быть осуществлено с использованием модели Хаббарда, однако ее применение возможно лишь при малом числе частиц. Для описания реальных соединений используется теория динамического среднего поля [1], в которой многочастичная проблема модели Хаббарда сводится к эффективной примесной задаче, решение которой можно осуществить с помощью одного из известных численных подходов, таких как квантовый метод Монте-Карло, метод точной диагонализации, метод ренорм-группы и т.д. [1]. По сравнению с остальными каждый из этих методов имеет свои преимущества и недостатки, которые в основном связаны с аппаратными ограничениями, и, как следствие, предоставляет возможность описывать только определенные области фазовой диаграммы.

В рамках настоящей статьи модернизирована существующая вычислительная схема метода точной диагонализации, что позволило исследовать сильнокоррелированные системы в области очень низких температур с эффективным использованием многопроцессорных систем и дало возможность проводить расчеты с меньшими вычислительными и временными затратами. Кроме того, частично преодолена проблема экспоненциального роста размерности задачи с увеличением числа частиц.

2. Теория. В рамках модели Хаббарда задача описания физических свойств сильнокоррелированных систем сводится к нахождению собственных значений и собственных векторов следующего гамильтониана, описывающего взаимодействия частицы со своими соседями:

$$H = - \sum_{(ij), \sigma} t_{ij} (c_{i\sigma}^+ c_{j\sigma} + c_{j\sigma}^+ c_{i\sigma}) + U \sum_i n_{i\uparrow} n_{i\downarrow}. \tag{1}$$

Здесь $c_{i\sigma}^+$ ($c_{i\sigma}$) — оператор рождения (уничтожения) на узле решетки i , t_{ij} — интеграл перехода между узлами i и j , U — локальное кулоновское взаимодействие и n_i — оператор числа частиц. Точное решение в модели (1) возможно лишь в случае малого количества частиц. В случае же реальных систем исследователи используют различные приближения. Например, в рамках теории динамического среднего поля (DMFT — Dynamical Mean-Field Theory) решеточная задача для модели Хаббарда сводится к эффективной примесной модели Андерсона:

$$H_{AM} = \varepsilon_0 (n_{0\uparrow} + n_{0\downarrow}) + U n_{0\uparrow} n_{0\downarrow} + \sum_{p>0, \sigma} [V_{0p} c_{0\sigma}^+ c_{p\sigma} + V_{0p}^* c_{p\sigma}^+ c_{0\sigma}] + \sum_{p>0, \sigma} \varepsilon_p c_{p\sigma}^+ c_{p\sigma}. \tag{2}$$

Здесь $\varepsilon_0 (n_{0\uparrow} + n_{0\downarrow})$ — энергия примесных уровней, $U n_{0\uparrow} n_{0\downarrow}$ — взаимодействие на примесных уровнях, $\sum_{p>0, \sigma} [V_{0p} c_{0\sigma}^+ c_{p\sigma} + V_{0p}^* c_{p\sigma}^+ c_{0\sigma}]$ — взаимодействие с резервуаром и $\sum_{p>0, \sigma} \varepsilon_p c_{p\sigma}^+ c_{p\sigma}$ — энергия резервуара. При

¹ Уральский государственный технический университет, физико-технический факультет, ул. Мира, д. 19, 620002, г. Екатеринбург; С. Н. Исаков, аспирант, e-mail: iskakoff@nqs.ru; В. В. Мазуренко, доцент, e-mail: mvv@dpt.ustu.ru

таким подходе вся динамика конкретной частицы рассматривается как ее взаимодействие с внешним резервуаром, созданным другими частицами. В рамках модели Андерсона для получения электронных свойств материала необходимо построить функцию Грина

$$G_{\sigma}^{cl}(i\omega_n) = \frac{1}{Z} \sum_{\nu\mu} \frac{\|\langle \mu | c_{\sigma}^{\dagger} | \nu \rangle\|^2}{E_{\nu} - E_{\mu} - i\omega_n} [e^{-\beta E_{\nu}} + e^{-\beta E_{\mu}}] = \frac{1}{Z} \sum_{\nu} e^{-\beta E_{\nu}} [G_{\sigma}^{\nu-}(i\omega_n) - G_{\sigma}^{\nu+}(i\omega_n)], \quad (3)$$

где E_{ν} — собственные значения матрицы гамильтониана модели Андерсона (2), $G_{\sigma}^{\nu+}$ и $G_{\sigma}^{\nu-}$ — функции Грина возбужденных состояний, $\omega_n = (2n + 1)k_B T$ — мацубаровские частоты и $\beta = \frac{1}{k_B T}$ — обратная температура. Исходя из экспоненциальной зависимости в (3), при больших значениях β достаточно знать лишь малое число собственных значений гамильтониана (2) для получения точности расчета 10^{-7} [2]. Например, для системы с числом орбиталей $n_s = 12$ при температуре около $T = 10$ К достаточно знать не более 20 наименьших собственных значений гамильтониана, при которых фактор Больцмана больше 10^{-7} . Поскольку для нахождения собственных значений гамильтониана необходимо провести его диагонализацию, то основная задача теории динамического среднего поля сводится к диагонализации матрицы гамильтониана (2) для последующего нахождения ее собственных значений. Однако размерность матрицы сильно зависит от числа степеней свободы, что накладывает значительные ограничения на возможность напрямую решить эту задачу [1, 3].

В настоящей статье проблема экспоненциального роста требуемой памяти для хранения матрицы гамильтониана была решена путем организации распределенного хранения массивов большой размерности, что позволило сократить объем памяти, требуемой на одном узле многопроцессорного комплекса, пропорционально числу используемых процессоров.

3. Анализ использования памяти для метода точной диагонализации. За основу нами взята существующая непараллельная реализация метода точной диагонализации [1].

При поиске состояний с наименьшей энергией в модернизируемом программном комплексе для нахождения собственных значений матрицы гамильтониана (2) используется метод Ланцоша. Этот метод выполняется быстрее полной диагонализации матрицы при поиске небольшого количества собственных значений, но для более точного описания возбужденных состояний требуется многократная реортогонализация пространства, что приводит к значительным вычислительным затратам.

В отличие от метода Ланцоша метод Арнольди позволяет избежать реортогонализации пространства, что, в свою очередь, существенно увеличивает быстродействие вычислений [3]. Основное время вычислений в методе Арнольди приходится на нахождение произведения матрицы гамильтониана на вектор. На предварительном этапе метод Ланцоша был заменен методом Арнольди, который позволяет решать более общий класс задач с меньшими затратами по времени [4]. В данном методе наибольший объем памяти расходуется на хранение матрицы гамильтониана (2) и ее собственного вектора. Однако при увеличении числа состояний размерность матрицы гамильтониана (2) быстро возрастает, что приводит к сложностям хранения матрицы гамильтониана в памяти.

Для сокращения занимаемой памяти в существующем программном комплексе используется схема хранения разреженной матрицы [5], при которой в памяти хранятся только ненулевые элементы матрицы и их индексы, однако применение этого программного комплекса для решения задач с числом состояний, большим $n_s = 12$, существенно затруднено из-за экспоненциального роста размерности задачи (рис. 1). Для решения данной проблемы была реализована схема распределенного хранения массивов при использовании многопроцессорных вычислительных систем с распределенной памятью. Суть подхода заключается в организации хранения и доступа к массивам, части которых физически хранятся на разных узлах вычислительной системы (рис. 2).

4. Реализация распределенного хранения данных. Для организации распределенного хранения больших массивов данных был разработан метод, обеспечивающий прозрачный доступ к распределенным данным. На первом этапе решения этой задачи был создан программный интерфейс приложения (API —

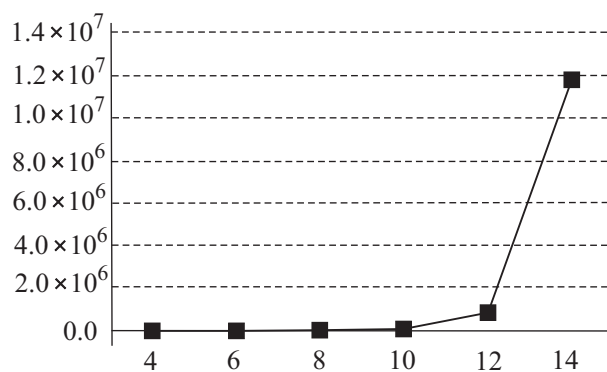


Рис. 1. Зависимость размерности задачи от числа состояний

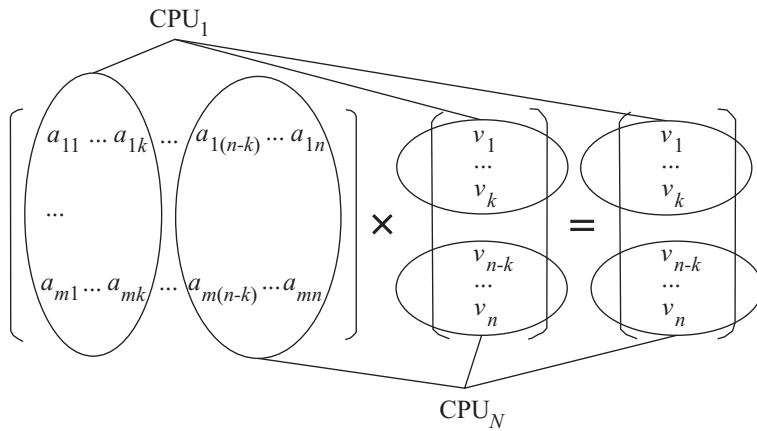


Рис. 2. Распределение массивов между вычислительными узлами

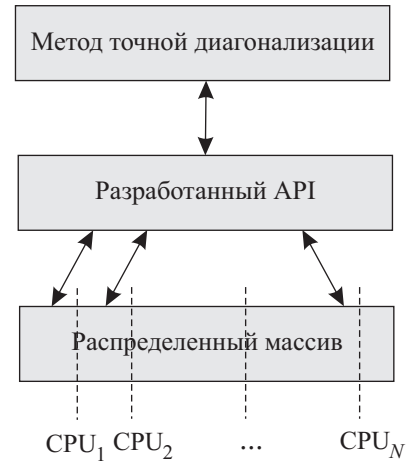


Рис. 3. Схема организации распределенного хранения данных

Application Programming Interface), позволяющий реализовать взаимодействие с распределенно хранимыми данными и сводящий к минимуму изменения, вносимые в существующий программный комплекс (рис. 3).

Разработанный API представляет собой процедуры и функции взаимодействия с распределенно хранимыми массивами, которые определяют, на каком узле находится требуемая ячейка матрицы, и организуют межпроцессное взаимодействие с требуемой частью массива. В предлагаемом API процедуры и функции можно разделить на два класса: процедуры и функции чтения данных из распределенного массива и процедуры и функции записи данных в распределенный массив.

Их принципиальное отличие заключается в необходимости организации межпроцессного взаимодействия. Операции, связанные с чтением данных, выполняются по следующей схеме:

- 1) определение узла вычислительного кластера, хранящего требуемую ячейку массива;
- 2) считывание данных из требуемой ячейки в локальную переменную на данном узле;
- 3) передача необходимых данных на вычислительные узлы с использованием выбранного типа межпроцессного обмена.

Операции, связанные с записью данных, реализованы без организации межпроцессных коммуникаций, поскольку все циклы доступа к распределенным массивам, ориентированные на запись, выполняются с использованием локальной адресации в рамках лишь той части распределенного массива, которая хранится на этом вычислительном узле.

Для реализации межпроцессного обмена в предлагаемом API было рассмотрено два альтернативных варианта, использующих коммуникационную библиотеку MPI [6]. Это широковещательные рассылки, которые позволяют передавать данные сразу всем процессам данного коммутатора, и односторонние коммуникации, которые позволяют процессу получать данные, хранимые в памяти другого процесса, без непосредственного участия второго процесса в доступе к данным. Во втором случае коммуникации организуются через заранее созданное другим процессом интеркоммуникационное окно, позволяющее текущему процессу самому указывать номер узла, из которого требуется получить данные.

Для выбора используемой схемы межпроцессного взаимодействия был проведен вычислительный эксперимент по оценке эффективности предложенных схем обмена данными.

5. Анализ эффективности методов доступа к данным. Важным критерием для определения эффективности параллельного алгоритма является его масштабируемость, которая характеризуется способностью системы увеличивать свою производительность при увеличении доступных ресурсов.

Для определения узких мест масштабирования разработанного программного комплекса был проведен вычислительный эксперимент на многопроцессорной кластерной вычислительной системе, установленной в Центре параллельных вычислений Уральского государственного технического университета. В процессе анализа результатов вычислительного эксперимента в разработанном программном комплексе были выделены два основных блока, занимающих большую часть вычислительного времени. Это блок заполнения матрицы гамильтониана — ориентировочно 15% общего времени, а также блок вычисления собственных значений матрицы гамильтониана — ориентировочно 75% общего времени.

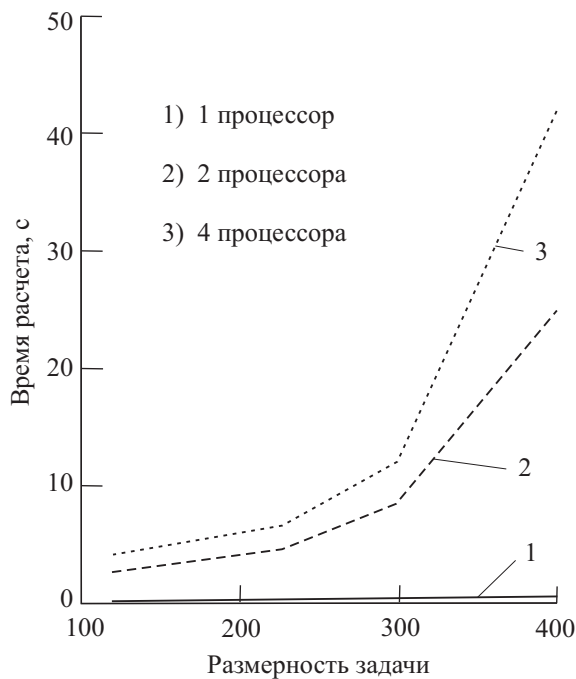


Рис. 4. Зависимость времени перемножения матрицы на вектор от размерности задачи при использовании широковещательных рассылок

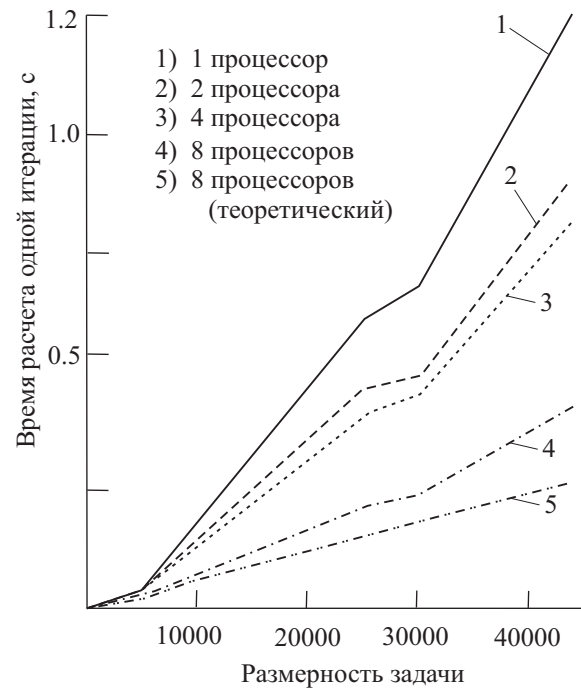


Рис. 5. Зависимость времени перемножения матрицы на вектор от размерности задачи при использовании односторонних коммуникаций

Основное внимание было уделено сравнению масштабируемости двух возможных схем организации межпроцессного взаимодействия и поиск путей повышения их эффективности. По результатам вычислительного эксперимента было выявлено, что метод доступа к данным, основанный на односторонних коммуникациях, позволяет существенно сократить время межпроцессного обмена при увеличении количества процессоров по сравнению с механизмом широковещательных рассылок, а также располагает большими возможностями для повышения эффективности вычислений. При использовании широковещательных рассылок даже при использовании лишь двух процессоров происходит значительный рост времени вычислений по сравнению со временем вычислений на однопроцессорной системе (рис. 4).

Причина такого увеличения времени связана с тем, что при использовании широковещательных рассылок в рамках решаемой задачи невозможно выполнять передачу всех требуемых данных в пределах одного цикла межпроцессного взаимодействия, что приводит к существенному увеличению суммарного времени передачи данных. В случае же односторонних коммуникаций все требуемые данные передаются в рамках единого цикла межпроцессного обмена, что существенно сокращает время, требуемое на передачу данных. Использование односторонних коммуникаций позволяет также сократить объем передаваемых данных пропорционально числу задействованных процессоров, что, в свою очередь, уменьшает время расчета при использовании большего количества доступных процессоров (рис. 5). Кроме того, при оптимизации работы программного комплекса многие многомерные массивы были преобразованы в одномерные, что позволило более эффективно задействовать процессорный кэш в вычислениях [7].

Основываясь на модели Хокни [8] для определения времени передачи данных между вычислительными узлами, эффективность разработанной схемы может быть оценена по формуле

$$S_p = \frac{T_1(N)}{T_p(N)} = \frac{T + Nt_s}{T/p + \alpha + N/(p\beta)}, \quad (4)$$

где S_p — ускорение от использования p процессоров, $T_1(N)$ — время вычисления в однопроцессорном режиме задачи размерности N , $T_p(N)$ — время вычисления задачи размерности N с использованием p процессоров, T — время перемножения матрицы на вектор, N — размерность задачи, t_s — время локального доступа к данным, α — латентность сети передачи данных в секундах и β — пропускная способность сети передачи данных. В случае использования высокоскоростных сетей ускорение от использования многопроцессорной системы будет стремиться к линейному. Полученный теоретический расчет времени вычисления на восьми процессорах (рис. 5) имеет некоторую погрешность, так как он

не учитывает некоторых особенностей сети передачи данных.

В ходе вычислительного эксперимента была также определена зависимость времени заполнения гамильтониана от числа процессоров (рис. 6). Причины роста времени связаны с большим объемом межпроцессорных коммуникаций, вызванные неоптимальной схемой заполнения матрицы гамильтониана в модернизируемом программном комплексе. В дальнейшем нами запланировано провести анализ возможных путей оптимизации соответствующего блока программного комплекса.

6. Заключение. В результате проведенной работы был разработан программный комплекс, позволяющий эффективно использовать многопроцессорные системы для решения задачи теории динамического среднего поля методом точной диагонализации. Использование параллельных алгоритмов на многопроцессорных кластерных системах позволяет решить проблему роста требуемой памяти на одном узле при помощи организации схемы распределенного хранения данных. Таким образом, исследователи получают доступ к решению сверхбольших задач (с большим количеством частиц) на многопроцессорных кластерных системах. Кроме того, было проведено сравнение двух альтернативных схем организации межпроцессорного взаимодействия, в результате которого межпроцессорный обмен был реализован при помощи односторонних коммуникаций. Данная схема межпроцессорного взаимодействия позволяет избежать дополнительных временных затрат на организацию прямого взаимодействия вычислительных узлов, как это происходит в случае двусторонних коммуникаций и, в частности, широковещательных рассылок.

В дальнейшем планируется оптимизация блока заполнения матрицы гамильтониана. В частности, планируется использовать более оптимальную схему компрессии разреженной матрицы [9], что позволит существенно сократить объем занимаемой памяти. Планируется также провести анализ возможных путей снижения числа межпроцессорных коммуникаций.

СПИСОК ЛИТЕРАТУРЫ

1. *Georges A., Kotliar G.* Dynamical mean-field theory of strongly correlated fermion systems and the limit of infinite dimensions // *Reviews of Modern Physics*. 1996. **68**, N 1. 13–125.
2. *Capone M.* Solving the dynamical mean-field theory at very low temperatures using the Lanczos exact diagonalization // *Physical Review B*. 2007. **76**, N 24. 245116.
3. *Perroni C.A., Ishida H., Liebsch A.* Exact diagonalization dynamical mean-field theory for multiband materials: effect of Coulomb correlations on the Fermi surface of $\text{Na}_{0.3}\text{CoO}_2$ // *Physical Review B*. 2007. **75**, N 4. 045125.
4. *Maschhoff K.J., Sorensen D.C.* A portable implementation of ARPACK for distributed memory parallel architectures (<http://www.caam.rice.edu/software/ARPACK>).
5. *Bunch J.R., Rose D.J.* Sparse matrix computations. New York–San Francisco–London: Academic Press, 1976.
6. MPI: A Message-Passing Interface standard. Message Passing Interface Forum. Knoxville, 2008.
7. SP Parallel Programming Workshop on Optimization Topic (<http://www.mhpcc.edu/training/workshop2/optimization>).
8. *Hockney R.* The communication challenge for MPP: Intel Paragon and Meiko CS-2 // *Parallel Computing*. 1994. **20**. 389–398.
9. *Chang C.C., Buchrer D.J., Kowng H.C.* An improvement to Ziegler’s sparse matrix compression algorithm // *J. of Systems and Software*. 1996. **35**, N 1. 67–71.

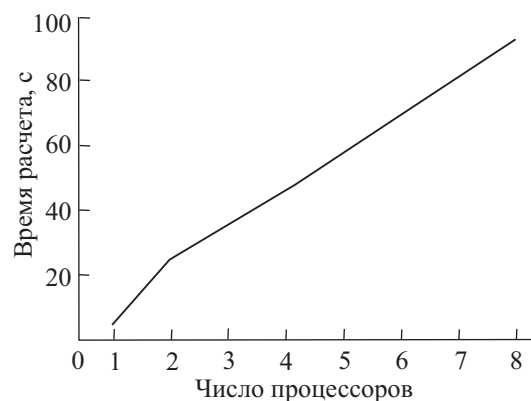


Рис. 6. Зависимость времени заполнения матрицы от числа процессоров

Поступила в редакцию
07.04.2009