ISSN 2079-3316 ПРОГРАММНЫЕ СИСТЕМЫ: ТЕОРИЯ И ПРИЛОЖЕНИЯ №3(30), 2016, с. 99–107

УДК 519.682.3

В. А. Роганов, В. И. Осипов, Г. А. Матвеев

Решение задачи Дирихле для уравнения Пуассона методом Гаусса-Зейделя на языке параллельного программирования T++

Аннотация. В статье описывается реализация решения задачи Дирихле для уравнения Пуассона методом Гаусса-Зейделя на языке параллельного программирования T++.

Ключевые слова и фразы: уравнения в частных производных, математическая физика, T++, OpenTS, MPI, параллельные вычисления, задача Пуассона, уравнение Пуассона, метод Гаусса-Зейделя.

Введение

Задача Дирихле для уравнения Пуассона является одной из классических задач математической физики. Для решения уравнений с частными производными как правило используются сеточные методы, Нередко с помощью компьютера в области определения строится сетка, и составляется разностное уравнение, в котором искомыми неизвестными являются значения функции в узлах сетки. Решение разностного уравнение также можно искать по-разному. На практике широко применяются итерационные методы. Вычислительная схема в этом случае описывает, как следующее состояние сетки зависит от предыдущего. В результате счета на компьютере получается приближенное решение уравнений с частными производными.

Бывают сложные модели, в которых строятся сетки с большим количеством узлов: десятки миллионов и даже больше. Актуальной задачей для таких моделей является распараллеливание вычислений с целью сокращения времени счета. При распараллеливании моделей с расчетными сетками обычно осуществляется декомпозиция

Работа выполнена в рамках проекта "Исследование и разработка методов решения ряда прикладных задач, использующих адаптивные алгоритмы" по программе фундаментальных исследований РАН I.5 на 2016 год "Проблемы создания высокопроизводительных распределенных и облачных систем и технологий. Интеллектуальные информационные технологии и системы".

расчетной области, то есть расчетная сетка делится на подобласти, которые распределяются между вычислительными узлами высокопроизводительных вычислительных комплексов. В процессе счета узлы комплекса обмениваются граничными значениями областей для сохранения корректности модели. Примеры с решением этой же задачи указанными сеточными методами описаны в [1–4].

OpenTS — одна из систем для параллельного программирования [5–9]. В ней используется язык программирования T++, который является синтаксическим расширением языка программирования C++. Язык T++ отличается от C++ наличием нескольких дополнительных ключевых слов, которые вставляются в описание некоторых функций или переменных. Для распараллеливания OpenTS использует библиотеку для параллельных вычислений MPI, но вызывать функции MPI внутри программы на T++ программисту при этом нет необходимости.

В настоящей статье описывается гибридная параллельная реализация задачи Дирихле для уравнения Пуассона на языке T++, в которой функциональная модель вычислений OpenTS сочетается с традиционным подходом с обменом сообщениями. Это дает возможность построения адаптивных алгоритмов путем сочетания динамического (T++) и статического распараллеливания уже существующих MPI-программ. В качестве основы для реализации используется алгоритм распараллеливания, описанный в [10].

1. Постановка задачи

Дано уравнение $\Delta u = f(x, y)$, где
Δ — оператор Лапласа, функция f определена на квадрате
$D = \{x, y \mid 0 \geqslant x \geqslant 1, 0 \geqslant y \geqslant 1\}$,
∂D — граница квадрата D,
$u(x, y) = g(x, y)$ при $x, y \in \partial D$,
функции f и g заданы.

Требуется найти функцию $u(x, y)$.

2. Решение. Последовательная версия

Разделим каждую сторону квадрата D на $N + 1$ равных частей и проведем прямые параллельно сторонам квадрата. Получим прямоугольную сетку с шагом $h = \frac{1}{N+1}$. Пусть u_{ij}^k — значение функции $u(x, y)$ на узле сетки с индексами i, j на итерации с номером k; i, j

изменяются от 0 до $N+1$. Согласно вычислительной схеме для метода Гаусса-Зейделя имеем:

$$u_{ij}^k = 0.25(u_{i-1,j}^k + u_{i,j-1}^k + u_{i,j+1}^k - u_{i+1,j}^k - h^2 f_{ij}).$$

Реализация метода на C++ при этом может выглядеть следующим образом [10] (приведен только основной цикл):

C++

```
do
 {
     max = 0;
     for (int i = 1; i < N + 1; i++)
        for (int j = 1; j < N + 1; j++)
        {
            double u0 = u[i][j];
            u[i][j] = 0.25 * (u[i -1][j] + u[i + 1][j]
                + u[i][j -1] + u[i][j + 1] -h * h * f[i -1][j -1]);
            double d = fabs(u[i][j] -u0);
            if (d > max)
                max = d;
        }
    } while (max > eps); // eps --- заданная точность.
```

3. Параллельные версии на MPI и T++

В параллельной реализации распараллеливание цикла производится с индексом i следующим образом. Пусть $rank$ – номер вычислительного узла, $0 \leqslant rank < size$, где $size$ – количество вычислительных узлов. Распределим вычисление между узлами так, что на узле с номером $rank$ переменная i изменялась от $i_{\min}$ до $i_{\max} - 1$, где

$$i_{\min} = \begin{cases} (N+2) * (rank)/size, & rank > 0, \\ 1, & rank = 0, \\ (N+2) * (rank+1)/size, & rank < size - 1, \\ N+1, & rank = size - 1. \end{cases}$$

Для сохранения целостности вычислений смежные вычислительные узлы обмениваются граничными значениями функции u в конце каждой итерации внешнего оператора цикла.

Обозначим через U^r_{ij} значения $u_{i,j}$ на вычислительном узле с номером r. Обмен граничными значениями функции u между узлами производится по формулам

$$U^r_{i_{\max} j} = U^{r+1}_{i_{\min} j}, \quad r = 0, ..., size - 2, \quad j = 0, ..., N + 1;$$
$$U^r_{i_{\min} - 1 j} = U^{r-1}_{i_{\max} j}, \quad r = 1, ..., size - 1, \quad j = 0, ..., N + 1.$$

В конце каждой итерации внешнего оператора цикла на каждом из узлов вычисляется невязка, потом находится ее максимальное значение. Если значение максимальной невязки меньше заданной точности вычисления eps, вычисления заканчиваются и значения функции u собираются на основном узле вычислительного комплекса.

Разница в реализациях задачи на T++ и MPI C++ состоит в следующем:

(1) программа на T++ использует лишь небольшое количество функций библиотеки MPI (MPI_Send, MPI_Receive и MPI_Allreduce);
(2) программа на T++ использует свой механизм распараллеливания задач, реализованный в ядре OpenTS;
(3) инициализация переменных в программе на T++ происходит так же, как и в последовательной версии, функция инициализации переменных вызывается на каждом вычислительном узле;
(4) после вычисления результаты собираются на основном вычислительном узле по-разному: программа на MPI C++ использует для сборки результатов функцию MPI_Gather, в то время как программа на T++ передает результаты с других узлов на основной узел через параметры T-функции;
(5) функции MPI вызываются в T++ немного по-другому: например, вызов в программе MPI
   ```
   MPI_Send(u, n, MPI_DOUBLE, ...)
   ```
   будет выглядеть в программе на T++ так:
   ```
   MPI_Send(u, n*sizeof(double), MPI_BYTE) , ...).
   ```
   Это связано с некоторыми ограничениями используемого программного адаптера DMPI.

4. Результаты испытаний

Выполнялись сравнительные испытания параллельных реализаций решения задачи Дирихле для уравнения Пуассона методом Гаусса-Зейделя на T++ и MPI C++. Испытания проводились на вычислительной системе, состоящей из 8-ми узлов с операционной системой Linux. Для испытаний использовалась расчетная сетка размером 1200×1200

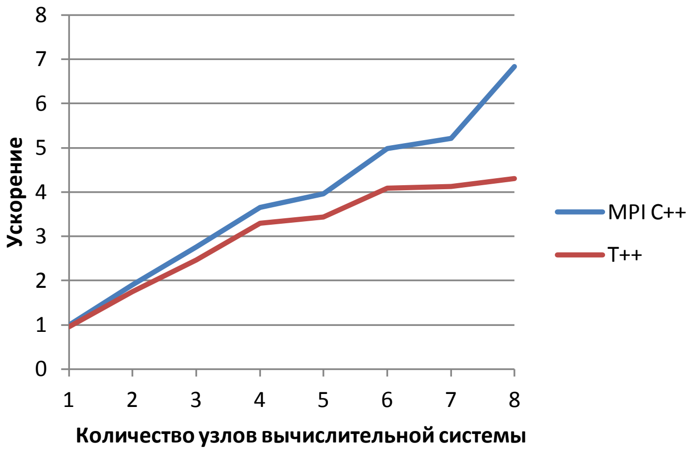

Рис. 1. Результаты испытаний параллельных реализаций решения задачи Дирихле

узлов. В процессе испытаний задача запускалась на разном количестве вычислительных узлов (от 1 до 8). Измерялось время работы программы. Результаты испытаний представлены на рис. 1. По горизонтальной оси отмечено количество вычислительных узлов, по вертикальной — ускорение. Верхний график — это ускорение MPI-программы, нижний график — ускорение программы на T++.

Время работы последовательной версии 43.8 сек. Время работы параллельных версий программы представлены в таблице 1. Проводилось также сравнение результатов вычисления параллельной и последовательной версии. Значения соответствующей невязки не превышают 0.002 для заданной точности eps = 0.0001.

Заключение

Из представленного графика видно, что программа на T++ работает несколько медленнее программы MPI. Это объясняется тем, что в программе на T++ результат счета, довольно объемная информация, в конце счета передается через параметры функции на основной узел, на что тратится больше времени по сравнению с программой на MPI, которая использует для этого функцию MPI_Gather.

Этот результат закономерен, так как MPI программа использует широкий набор оптимизированных операций, тогда как программа на T++ использует ограниченный набор функций MPI.

Таблица 1. Время работы параллельных версий программы

Количество узлов	1	2	3	4	5	6	7	8
Время работы программы на C++ MPI в сек.	43.8	23.1	15.9	12.0	11.1	8.8	8.4	6.4
Время работы программы на T++ в сек.	45.9	25	17.8	13.3	12.8	10.7	10.6	10.2

Ценность полученных результатов состоит в том, что опробованы программные приемы, позволившие эффективно совместить логику Т-системы и MPI. Первая хорошо распределяет и асинхронно работает с данными, второй — хорош для сложившихся конфигураций.

Существует много программ, написанных на MPI. Полученные результаты подтверждают, что можно использовать Т-систему для создания адаптивных вычислительных конфигураций на базе готовых MPI-кодов.

Список литературы

[1] R. J. Gonsalves. *Partial Differential Equations in Physics*, Computational Physics PHY 410-505, URL: http://www.physics.buffalo.edu/phy410-505/topic6/index.html ↑ 100

[2] V. P. Gergel. *Introduction to Parallel Programming*, CS338, section 12, University of Nizhni Novgorod Faculty of Computational Mathematics & Cybernetics, URL: http://www.hpcc.unn.ru/mskurs/ENG/DOC/pp12.pdf ↑ 100

[3] C. W. Oosterlee, H. Bijl, H. X. Lin, S. W. de Leeuw, J. B. Perot, C. Vuik, P. Wesseling. *Lecture Notes for course (wi4145TU) "Computational Science and Engineering"*, URL: http://ta.twi.tudelft.nl/mf/users/oosterle/oosterlee/boek.pdf ↑ 100

[4] Th. Guilleta, R. Teyssiera. *A Simple Multigrid Scheme for Solving the Poisson Equation with Arbitrary Domain Boundaries*, arXiv: 1104.1703v [physics.comp-ph]. ↑ 100

[5] С. М. Абрамов, В. А. Васенин, Е. Е. Мамчиц, В. А. Роганов, А. Ф. Слепухин. «Динамическое распараллеливание программ на базе параллельной редукции графов. Архитектура программного обеспечения новой версии Т-системы», *Научная сессия МИФИ-2001*, Сборник научных трудов. Т. 2 (Москва, 22–26 января 2001 г.), с. 234. ↑ 100

[6] С. М. Абрамов, А. А. Кузнецов, В. А. Роганов. «Кроссплатформенная версия Т-системы с открытой архитектурой», *Вычислительные методы и программирование*, **8**:1(2) (2007), с. 175–180, URL: http://num-meth.srcc.msu.ru/zhurnal/tom_2007/v8r203.html ↑ 100

[7] *Система параллельного программирования OpenTS*, URL: http://www.opents.net/index.php/ru/ ↑ 100

[8] С. М. Абрамов, И. М. Загоровский, М. Р. Коваленко, Г. А. Матвеев, В. А. Роганов. «Миграция от MPI к платформе OpenTS: эксперимент с приложениями PovRay и ALCMD», *Международная конференция "Программные системы: теория и приложения"*. Т. 1 (Переславль-Залесский, октябрь 2006), Наука. Физматлит, М., с. 265–275. ↑ 100

[9] A. Moskovsky, V. Roganov, S. Abramov, A. Kuznetsov. “Variable Reassignment in the T++ Parallel Programming Language”, *Parallel Computing Technologies*, Proceedings, 9th International Conference, PaCT 2007 (Pereslavl-Zalessky, Russia, September 2007), Lecture Notes in Computer Science, vol. **4671**, ed. V. Malyshkin, Springer, Berlin etc., 2007, pp. 579–588. ↑ 100

[10] М. Э. Абрамян. *Параллельные методы решения краевых задач, основанные на технологиях OpenMP и MPI*, 2015, URL: http://edu.mmcs.sfedu.ru/mod/resource/view.php?id=2165 ↑ 100,101

Рекомендовал к публикации *д.ф.-м.н. С. В. Знаменский*

Пример ссылки на эту публикацию:

В. А. Роганов, В. И. Осипов, Г. А. Матвеев. «Решение задачи Дирихле для уравнения Пуассона методом Гаусса-Зейделя на языке параллельного программирования T++», *Программные системы: теория и приложения*, 2016, **7**:3(30), с. 99–107.

URL: http://psta.psiras.ru/read/psta2016_3_99-107.pdf

Об авторах:

Владимир Александрович Роганов

Научный сотрудник ИПС им. А.К. Айламазяна РАН. Разработчик современных версий Т-системы, ведущий разработчик системы OpenTS. Принимал активное участие в суперкомпьютерных проектах Союзного государства России и Беларуси, в том числе в проектах «СКИФ» и «СКИФ-ГРИД»

e-mail: var@pereslavl.ru

Валерий Иванович Осипов

К.ф.-м.н., научный сотрудник ИПС им. А.К. Айламазяна РАН. Один из разработчиков системы OpenTS. Принимал участие в суперкомпьютерных проектах Союзного государства России и Беларуси

e-mail: val@pereslavl.ru

Герман Анатольевич Матвеев

Ведущий инженер-исследователь ИЦМС ИПС им. А.К. Айламазяна РАН. Один из разработчиков системы OpenTS. Принимал участие в суперкомпьютерных проектах Союзного государства России и Беларуси

e-mail: gera@prime.botik.ru

Vladimir Roganov, Valerii Osipov, German Matveev. *Solving the 2D Poisson PDE by Gauss-Seidel method with parallel programming system OpenTS.*

Abstract. The article describes Solving the 2D Poisson PDE by Gauss-Seidel method with parallel programming system OpenTS. (*In Russian*).

Key words and phrases: dynamic parallelization, T-system with an open architecture, OpenTS, T++ programming language, partial differential equations, pde, Poisson, Dirichlet, Gauss, Seidel.

References

[1] R. J. Gonsalves. *Partial Differential Equations in Physics*, Computational Physics PHY 410-505, URL: http://www.physics.buffalo.edu/phy410-505/topic6/index.html

[2] V. P. Gergel. *Introduction to Parallel Programming*, CS338, section 12, University of Nizhni Novgorod Faculty of Computational Mathematics & Cybernetics, URL: http://www.hpcc.unn.ru/mskurs/ENG/DOC/pp12.pdf

[3] C. W. Oosterlee, H. Bijl, H. X. Lin, S. W. de Leeuw, J. B. Perot, C. Vuik, P. Wesseling. *Lecture Notes for course (wi4145TU) "Computational Science and Engineering"*, URL: http://ta.twi.tudelft.nl/mf/users/oosterle/oosterlee/boek.pdf

[4] Th. Guilleta, R. Teyssiera. *A Simple Multigrid Scheme for Solving the Poisson Equation with Arbitrary Domain Boundaries*, arXiv: 1104.1703v [physics.comp-ph].

[5] S. M. Abramov, V. A. Vasenin, Ye. Ye. Mamchits, V. A. Roganov, A. F. Slepukhin. "Dinamicheskoye rasparallelivaniye programm na baze parallel'noy reduktsii grafov. Arkhitektura programmnogo obespecheniya novoy versii T-sistemy", *Nauchnaya sessiya MIFI-2001*, Sbornik nauchnykh trudov. V. 2 (Moskva, 22–26 yanvarya 2001 g.), pp. 234 (in Russian).

[6] S. M. Abramov, A. A. Kuznetsov, V. A. Roganov. "Krossplatformennaya versiya T-sistemy s otkrytoy arkhitekturoy", *Vychislitel'nyye metody i programmirovaniye*, **8**:1(2) (2007), pp. 175–180 (in Russian), URL: http://num-meth.srcc.msu.ru/zhurnal/tom_2007/v8r203.html

[7] *The OpenTS parallel programming system*, URL: http://www.opents.net/index.php/en/

[8] S. M. Abramov, I. M. Zagorovskiy, M. R. Kovalenko, G. A. Matveyev, V. A. Roganov. "Migratsiya ot MPI k platforme OpenTS: eksperiment s prilozheniyami PovRay i ALCMD", *Mezhdunarodnaya konferentsiya "Programmnyye sistemy: teoriya i prilozheniya"*. V. 1 (Pereslavl'-Zalesskiy, oktyabr' 2006), Nauka. Fizmatlit, M., pp. 265–275 (in Russian).

[9] A. Moskovsky, V. Roganov, S. Abramov, A. Kuznetsov. "Variable Reassignment in the T++ Parallel Programming Language", *Parallel Computing Technologies*, Proceedings, 9th International Conference, PaCT 2007 (Pereslavl-Zalessky, Russia, September 2007), Lecture Notes in Computer Science, vol. **4671**, ed. V. Malyshkin, Springer, Berlin etc., 2007, pp. 579–588.

[10] M. E. Abramyan. *Parallel'nyye metody resheniya krayevykh zadach, osnovannyye na tekhnologiyakh OpenMP i MPI*, 2015 (in Russian), URL: http://edu.mmcs.sfedu.ru/mod/resource/view.php?id=2165

Sample citation of this publication:

Vladimir Roganov, Valerii Osipov, German Matveev. "Solving the 2D Poisson PDE by Gauss-Seidel method with parallel programming system OpenTS", *Program systems: theory and applications*, 2016, **7**:3(30), pp. 99–107. (*In Russian*). URL: http://psta.psiras.ru/read/psta2016_3_99-107.pdf