УДК 519.688:519.6:514.752 DOI: 10.26661/2413-6549-2019-2-05

ВІЗУАЛІЗАЦІЯ ГЕОМЕТРИЧНИХ ОБЛАСТЕЙ СКЛАДНОЇ ФОРМИ В ПАРАЛЕЛЬНИХ ОБЧИСЛЮВАЛЬНИХ СИСТЕМАХ ЗІ СПІЛЬНОЮ ПАМ'ЯТТЮ

М. С. Ігнатченко, О. В. Кудін

Запорізький національний університет
avk256@gmail.com

Ключові слова:
геометричний об'єкт, R-функція, дискретна модель, паралельний алгоритм, візуалізація.

Практичне застосування чисельних методів для розв'язання крайових задач потребує створення дискретних моделей геометричних областей складної форми. Проблему генерації сіткової моделі можна розбити на дві окремі задачі: 1) створення формального опису вихідної геометричної області та 2) побудову її дискретної моделі на основі цього опису.

Найбільш складним є перший етап особливо для геометричних областей нестандартної форми. Універсальним способом параметричного опису геометричної області будь-якої складності є використання апарату R-функцій, які дозволяють за допомогою логічних операцій інверсії, кон'юнкції та диз'юнкції над елементарними математичними співвідношеннями будувати неявні функції, що однозначно описують границю геометричного об'єкта довільної форми.

Практичне застосування R-функцій є досить складним в силу їх неявної природи. Для візуалізації геометричних областей, описаних за допомогою неявних функцій, необхідно найперше побудувати набір опорних точок, які належать границі вихідної області, а потім для цього набору згенерувати воксельну або гранично-елементну модель. Задача ефективного пошуку точок, що належать границі геометричної області, описаної за допомогою неявної R-функції, на сьогодні є актуальною й вимагає розробки відповідних підходів і алгоритмів.

У статті описується запропонований паралельний алгоритм побудови та візуалізації геометричних моделей областей, описаних неявними функціями, в обчислювальних системах із загальною пам'яттю. Цей алгоритм було реалізовано із застосуванням Стандартної бібліотеки мови програмування C++ (C++11). Було створено відповідне програмне забезпечення й проведено ряд обчислювальних експериментів, які показали ефективність запропонованого алгоритму на комп'ютерах з різними типами процесорів.

VISUALISATION OF GEOMETRIC REGIONS OF COMPLEX SHAPES IN SHARED MEMORY PARALLEL COMPUTERS

M. S. Ignatchenko, O. V. Kudin

Zaporizhzhia National University
avk256@gmail.com

Key words:
geometric object, R-function, discrete model, parallel algorithm, visualization.

Using of the numerical methods for the solution of boundary value problems requires the making of discrete models of geometric regions of complex shape. The problem of generating a mesh can be divided into two independent tasks: 1) creating a formal description of the initial geometric region and 2) building its discrete model based on this description.

The most difficult is the first stage, especially for geometric areas of irregular shape. A universal way to parametrically describe a geometric region of arbitrary shape is to use R-functions, which allow using logical operations of inversion, conjunction, and disjunction over elementary mathematical relations to construct implicit functions that uniquely describe the boundary of an arbitrary geometric object.

The practical application of R-functions is quite complicated due to their implicit nature. To visualize the geometric regions described using implicit functions, you must first find a set of reference points that belong to the boundary of the original region, and then construct a voxel or boundary model for this set. The problem of effective finding points that belong to the boundary of the geometric region described using the implicit R-function is relevant today and requires the development of appropriate approaches and algorithms.

This article describes a parallel algorithm for constructing and visualizing geometric models of regions described by implicit functions in computing systems with shared memory. This algorithm was implemented using the C++ standard library (C++11). A series of computational experiments were carried out, showing the effectiveness of the proposed algorithm on computers with various types of processors.

1. Вступ

Чисельний аналіз крайових задач із використанням, наприклад, методу скінченних елементів передбачає створення дискретних (скінченно-елементних) моделей дво- і тривимірних геометричних областей складної форми. На практиці проблема автоматичної генерації дискретної моделі може бути розбита на дві окремі задачі:

створення формального опису вихідної геометричної області у формі, придатній для подальшої автоматичної обробки із застосуванням комп'ютерної техніки;

побудова дискретної моделі вихідної області за раніше отриманим формальним описом [1-3].

Перша задача є нетривіальною, особливо для геометричних областей нестандартної форми. На практиці для опису формальних моделей геометричних областей найчастіше використовують граничне й твердотільне геометричне моделювання [4]. Головним недоліком цих класичних підходів є досить висока трудомісткість моделювання об'єктів нетипової форми. Найбільш універсальним і природним способом формального опису геометричної області довільної форми Ω є застосування R-функцій, запропонованих академіком В. Л. Рвачовим [5]. Основна ідея цього підходу полягає в побудові такої функції $F(x)$, для якої виконуються, наприклад, такі співвідношення: $F(x) \geq 0$, якщо $x \in \Omega$ ($F(x) = 0$, якщо x знаходиться на границі Ω – $\partial\Omega$), і $F(x) < 0$, якщо $x \notin \Omega$. В. Л. Рвачовим було доведено, що функцію $F(x)$ для будь-якої геометричної області Ω можна сконструювати за допомогою набору елементарних математичних функцій і логічних операцій кон'юнкції, диз'юнкції та інверсії над ними. Проте практичне застосування такого підходу є досить складним, так як функція $F(x)$ є неявною. Тому візуалізація геометричних областей, описаних R-функціями, є нетривіальною задачею. Огляд існуючих алгоритмів візуалізації неявних функцій наведено в роботах [6-12]. Однак раніше розроблені алгоритми є або повільними послідовними, або орієнтовані на побудову тільки воксельных моделей вихідних геометричних областей. Тому розробка ефективних паралельних алгоритмів візуалізації неявних функцій на сьогодні є актуальною. У цій статті описується запропонований паралельний алгоритм побудови гранично-елементних моделей тривимірних геометричних об'єктів, описаних R-функціями, в обчислювальних системах зі спільною пам'яттю.

2. Використання паралельних систем зі спільною пам'яттю для геометричного моделювання

На сьогодні з появою і широким розповсюдженням багатопроцесорних та/або багатоядерних обчислювальних систем великої актуальності набула проблема розробки спеціалізованих паралельних алгоритмів для розв'язання актуальних прикладних задач. На цей час існує досить багато різноманітних типів обчислювальних архітектур, найбільш поширеними серед яких є такі [13]:

– SISD (Single Instruction, Single Data) – комп'ютерні системи, в яких одиночному потоку команд, що виконуються, відповідає єдиний потік даних (звичайні однопроцесорні комп'ютери);

– SIMD (Single Instruction, Multiple Data) – обчислювальні системи з одиночним потоком команд і множинним потоком даних (GPU, MMX та інші);

– MIMD (Multiple Instruction, Multiple Data) – комп'ютери з множинним потоком команд і множинним потоком даних (сучасні багатопроцесорні системи).

Остання архітектура є найбільш поширеною на сьогодні через свою обчислювальну ефективність [14]. MIMD-системи прийнято в загальному випадку поділяти на дві окремі категорії:

1) мультипроцесори (multiprocessor) – комп'ютерні системи зі спільною пам'яттю (найчастіше це комп'ютери з багатоядерними процесорами);

2) мультикомп'ютери (multicomputer) – обчислювальні системи без спільної пам'яті (наприклад, обчислювальні кластери) [15, 16].

Розробка паралельних алгоритмів для мультикомп'ютерів є складним і трудомістким процесом через необхідність реалізації синхронізації доступу до даних, потенційну можливість виникнення взаємоблокувань (deadlock), труднощі у налагодженні тощо [17], тому в даній роботі розглянута паралельна реалізація геометричного моделювання в обчислювальних системах зі спільною пам'яттю.

При візуалізації геометричних областей, заданих неявними функціями, найбільш простим й зручним для практичної реалізації є метод паралельної декомпозиції «Розділяй і володарюй» [18]. Згідно з ним вихідна задача пошуку множини точок, що належать границі $\partial\Omega$ неявно заданої геометричної області Ω, поділяється на певну сукупність підзадач, які розв'язуються паралельно, а отримані результати згодом об'єднуються. Послідовний алгоритм пошуку границі області, заданої R-функцією ($F(x) = 0$), можна описати за допомогою псевдокоду [19] наступним чином:

Алгоритм *Процедура пошуку граничних вузлів геометричній області*

procedure *FindBnPts*(*BoundaryPoints, Box, N*)

BnPts – шуканий вектор координат точок на границі області

$Box = (X_{min}, Y_{min}, Z_{min}, X_{max}, Y_{max}, Z_{max})$ – координати суперобласті (зони пошуку)

$N = (N_x, N_y, N_z)$ – кількість кроків вздовж осей координат

begin
  $H_x \leftarrow (X_{max} - X_{min})/N_x$
  $H_y \leftarrow (Y_{max} - Y_{min})/N_y$
  $H_z \leftarrow (Z_{max} - Z_{min})/N_z$
  while $i \in [0, N_x - 1]$ **do**
    $x_0 \leftarrow X_{min} + i \cdot H_x$
    $x_1 \leftarrow X_{min} + (i+1) \cdot H_x$
    while $j \in [0, N_y - 1]$ **do**
      $y_0 \leftarrow Y_{min} + j \cdot H_y$
      $y_1 \leftarrow Y_{min} + (j+1) \cdot H_y$
      while $k \in [0, N_k - 1]$ **do**
        begin
        $z_0 \leftarrow Z_{min} + k \cdot H_z$
        $z_1 \leftarrow Z_{min} + (k+1) \cdot H_z$
        if $F(x_0, y_0, z_0) \le 0$ **and**
            $F(x_1, y_1, z_1) \ge 0$ **then**
                $Find(x_0, y_0, z_0, x_1, y_1, z_1) \rightarrow BnPts$
  end while
end procedure

Тут під суперобластю зазвичай розуміється деяка кубоїдна область, що задається користувачем, в середині якої завідомо знаходиться вихідний геометричний об'єкт. Процедура $Find$ реалізує пошук на відрізку $(x_0, y_0, z_0) - (x_1, y_1, z_1)$ координати точки, для якої R-функція, що описує вихідну геометричну область, приймає нульове значення.

Отриманий в результаті роботи вищенаведеного алгоритму масив граничних точок потім використовується для візуалізації області або у вигляді воксельного, або у вигляді граничного подання. На рис. 1 наведено отримане граничне подання геометричній області «Атом», яку можна описати за допомогою R-функції такого вигляду:

$$F(x,y,z) = F_1(x,y,z) \vee F_2(x,y,z),$$

де $F_1(x,y,z) = R^2 - x^2 - y^2 - z^2$;

$F_2(x,y,z) = F_3(x,y,z) \vee F_4(x,y,z)$;

$F_3(x,y,z) = a\sqrt{x^2+y^2} - -(x^2+y^2+z^2) - b$;

$F_4(x,y,z) = F_5(x,y,z) \vee F_6(x,y,z)$;

$F_5(x,y,z) = a\sqrt{x^2+z^2} - -(x^2+y^2+z^2) - b$;

$F_6(x,y,z) = a\sqrt{y^2+z^2} - -(x^2+y^2+z^2) - b$;

$R = 9, a = 10, b = 24.$

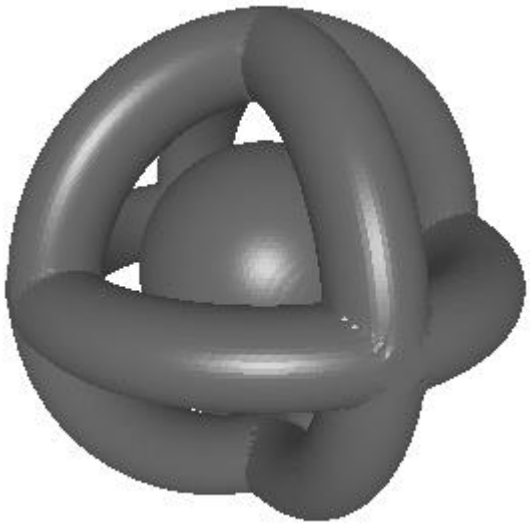

Рис. 1. Геометрична область «Атом»

Візуалізація поверхні вихідної області за допомогою набору опорних точок, що належать її границі, отримана за допомогою алгоритму Marching Cubes [20]. Наведений вище послідовний алгоритм можна модифікувати в паралельний, якщо передбачити розбиття суперобласті, що задається користувачем, на певну кількість окремих підобластей, число яких, наприклад, відповідає наявним в обчислювальній системі процесорам, ядрам або потокам, що паралельно виконуються, а результати їх роботи об'єднати в один результуючий масив граничних вузлів.

Запропонований паралельний алгоритм було реалізовано за допомогою Стандартної бібліотеки мови програмування C++ (C++11 [21]). Обчислювальний експеримент проводився на комп'ютері з процесором AMD Ryzen 7 2700X Eight-Core Processor з тактовою частотою 3.70 ГГц і об'ємом оперативної пам'яті 32 Гбайт під управлінням операційної системи Windows 10. Для пошуку границі області будувалася сітка, що складалася з $300 \times 300 \times 300$ вузлів. Час роботи алгоритму при побудові зображення геометричної області «Атом» при різній кількості задіяних обчислювальних потоків наведено на рис. 2.

Рис. 2. Час виконання паралельного алгоритму візуалізації геометричної області «Атом» при різній кількості задіяних потоків на процесорі AMD Ryzen

На рис. 3 наведено аналогічний графік, отриманий при використанні іншого комп'ютера з процесором Intel (R) Core (TM) i7-3630QM CPU @ 2.40 GHz і 16 ГБ пам'яті, що також працює під управління ОС Windows 10.

З наведених графіків видно, що час роботи паралельного алгоритму логарифмічно зменшується із зростанням кількості використаних потоків до того моменту, коли їх кількість не стане рівною кількості фізичних ядер процесора (AMD Ryzen – 8, Intel i7 – 4). Після чого швидкість роботи практично не змінюється. При практичній реалізації даного алгоритму слід врахувати накладні витрати операційної системи на переключення управління між потоками.

Рис. 3. Час роботи алгоритму на процесорі Intel i7

3. Висновки

Запропонований паралельний алгоритм візуалізації тривимірних геометричних областей, описаних за допомогою R-функцій, є простим і легким в практичній реалізації, а також характеризується високою швидкістю виконання. Отже, його застосування дозволяє дослідникові підвищити ефективність своєї роботи при геометричному моделюванні об'єктів складної форми для подальшої побудови їх дискретних представлень.

Перспективи подальших досліджень пов'язані з адаптацією запропонованого алгоритму для застосування на мультикомп'ютерах.

Література

1. Чопоров С. В., Гребенюк С. Н., Гоменюк С. И. Функциональный подход к геометрическому моделированию технических систем. Запорожье: ЗНУ, 2016. 177 с.
2. Чопоров С. В., Гоменюк С. И., Алатамнех Х. Х., Оспищев К. С. Методы построения дискретных моделей: структурированные и блочно-структурированные сетки. *Вісник Запорізького національного університету. Фізико-математичні науки*. 2016. № 1. С. 272–284.
3. Чопоров С. В., Лисняк А. А., Борисовская Ю. А., Козлова О. С., Снежкова Л. С. Методы построения дискретных моделей: неструктурированные сетки. *Вісник Запорізького національного університету. Фізико-математичні науки*. 2016. № 2. С. 237–250.
4. Голованов Н. Н. Геометрическое моделирование. Москва: Из-во физ.-мат. лит., 2002. 472 с.
5. Рвачев В. Л. Теория R-функций и некоторые ее приложения. Киев: Наук. думка, 1982. 106 с.
6. Pasko A., Adzhiev V., Sourin A. Savchenko V. Function representation in geometric modeling: concepts, implementation and applications. *The visual computer*. 1995. Vol. 11. P. 429–446.
7. Мыльцев А. М., Толок А. В. Математическая модель визуализации динамического массива данных при построении трёхмерных сцен. *Вісник Запорізького державного університету. Фізико-математичні науки*. 2003. № 3. С. 1–6.
8. Максименко-Шейко К. В., Мацевитый А. М., Шейко Т. И. Конструктивные средства метода R-функций для построения примитивов в 3D. *Проблемы машиностроения*. 2005. Т. 8. № 1. С. 59–65.

9. Толок А. В. Функционально-воксельный метод в компьютерном моделировании. Москва: ФИЗМАТЛИТ, 2016. 112 с.
10. Гоменюк С. І., Чопоров С. В., Аль-Атамнех Б. Г. М. Математичне моделювання геометричних об'єктів у паралельних комп'ютерних системах: монографія. Херсон: Вид-чий дім «Гельветика», 2018. 112 с.
11. Чопоров С. В., Гоменюк С. И. Параллельный способ построения сеток треугольных элементов при функциональном. *Вестник Херсонского национального технического университета*. 2015. № 3(54). С. 511–517.
12. Чопоров С. В. Использование технологий параллельных вычислений в методе конечных элементов. *Радіоелектроніка, інформатика, управління*. 2013. № 2(29). С. 88–94.
13. Flynn M. J. Some computer organizations and their effectiveness. *IEEE Transactions on Computers*. 1972. Vol 21 (9). P. 948–960.
14. Xu Z., Hwang K. Scalable Parallel Computing Technology, Architecture, Programming. McGraw-Hill Science/Engineering/Math, 1998. 832 p.
15. Таненбаум Э., Стеен М. Распределенные системы. Принципы и парадигмы. Санкт–Петербург: Питер, 2003. 877 с.
16. Таненбаум Э., Остин Т. Архитектура компьютера. 6-е изд. Санкт–Петербург: Питер, 2013. 816 с.
17. Малышкин В. Э., Корнеев В. Д. Параллельное программирование мультикомпьютеров. Новосибирск: НГУ, 2006. 439 с.
18. Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К. Алгоритмы: построение и анализ. Москва: Вильямс, 2005. 1296 с.
19. Макконнелл Дж. Основы современных алгоритмов. Москва: Техносфера, 2004. 368 с.
20. William E. L., Harvey E. C. Marching Cubes: A high resolution 3D surface construction algorithm. *Computer Graphics*. 1987. Vol. 21 (4). P. 163–169.
21. C++11 Overview. URL: https://isocpp.org/wiki/faq/cpp11#cpp11-specific-goals.

References

1. Choporov, S. V., Grebenyuk, S. N. & Homeniuk, S. I. (2016). Functional approach to geometric modeling of technical systems. Zaporizhzhya: ZNU (in Russian).
2. Choporov, S. V., Homeniuk, S. I., Alatamneh, H. H. & Ospischev, K. S. (2016). Methods for constructing discrete models: structured and block-structured grids. Visnik Zaporizkogo nacionalnogo universitetu. Fiziko-matematichni nauki, No. 1, pp. 272–284 (in Russian).
3. Choporov, S. V., Lisnyak, A. A., Borisovskaya, Y. A., Kozlova, O. S. & Snezkova, L. S. (2016). Discrete Model Building Methods: Unstructured Grids. Visnik Zaporizkogo nacionalnogo universitetu. Fiziko-matematichni nauki, No. 2. pp. 237–250 (in Russian).
4. Golovanov, N. N. (2002). Geometric modeling. Moscow: Izdatelstvo fiziko-matematicheskoj literatury (in Russian).
5. Rvachev, V. L. (1982). Theory of R-functions and Applications. Kyiv: Naukova Dumka (in Russian).
6. Pasko, A., Adzhiev, V., Sourin, A. & Savchenko, V. (1995). Function representation in geometric modeling: concepts, implementation and applications. The visual computer, Vol. 11, pp. 429–446.
7. Mylcev, A. M. & Tolok, A. V. (2003). A mathematical model for visualizing a dynamic data array when building three-dimensional scenes. Visnik Zaporizkogo derzhavnogo universitetu, No. 3, pp. 1–6 (in Russian).
8. Maksimenko-Shejko, K. V., Macevityj, A. M. & Shejko, T. I. (2005). Constructive tools of the R-function method for constructing primitives in 3D. Problemy mashinostroeniya, Vol. 8, Issue 1, pp. 59–65 (in Russian).
9. Tolok, A. V. (2016). Functional voxel method in computer simulation. Moscow: FIZMATLIT (in Russian).

10. Homeniuk, S. I., Choporov, S. V. & Al-Atamneh, B. G. M. (2018). Mathematical modeling of geometric objects in parallel computer systems: monograph. Kherson: Vidavnichij dim “Gelvetika” (in Ukrainian).
11. Choporov, S. V. & Homeniuk, S. I. (2015). Parallel method meshing triangular elements when the functional representation. Vestnik Hersonskogo nacionalnogo tehnicheskogo universiteta, No. 3(54), pp. 511–517 (in Russian).
12. Choporov, S. V. (2013). The using of parallel computing technologies in the finite element method. Radioelektronika, informatika, upravlinnya, No. 2(29), pp. 88–94 (in Russian).
13. Flynn, M. J. (1972). Some computer organizations and their effectiveness. IEEE Transactions on Computers, Vol. 21(9), pp. 948–960.
14. Xu, Z. & Hwang, K. (1998). Scalable Parallel Computing Technology, Architecture, Programming. McGraw-Hill Science/Engineering/Math.
15. Tanenbaum, E. & Steen, M. (2003). Distributed systems. Principles and Paradigms. Saint Petersburg: Piter (in Russian).
16. Tanenbaum, E. & Ostin, T. (2013). Computer architecture: 6th ed. Saint Petersburg: Piter (in Russian).
17. Malyshkin, V. E. & Korneev, V. D. (2006). Parallel programming of multicomputers. Novosibirsk: NGU (in Russian).
18. Kormen, T., Lejzerson, Ch., Rivest, R. & Shtajn, K. (2005). Algorithms: construction and analysis. Moscow: Vilyams (in Russian).
19. Makkonnell, D. (2004). The basics of modern algorithms. Moscow: Tehnosfera (in Russian);
20. William, E. L. & Harvey, E. C. (1987). Marching Cubes: A high resolution 3D surface construction algorithm. Computer Graphics, Vol. 21 (4), pp. 163–169.
21. C++11 Overview. Retrieved from https://isocpp.org/wiki/faq/cpp11#cpp11-specific-goals.