TẠP CHÍ KHOA HỌC VÀ CÔNG NGHỆ ĐẠI HỌC DUY TÂN
DTU Journal of Science and Technology 01(74) (2026) 71-78
ĐẠI HỌC DUY TÂN
DUYTAN UNIVERSITY

Phương pháp số hiệu quả trong giải phương trình Poisson: Các chiến lược tối ưu hóa thời gian tính toán

Efficient Numerical Methods for Solving the Poisson Equation: Strategies for Computational Time Optimization

Lê Thị Quỳnh Trang[a,b*]
Le Thi Quynh Trang[a,b*]

[a]*Viện Nghiên cứu và Phát triển Công nghệ cao, Đại học Duy Tân, Đà Nẵng, Việt Nam*
[a]*Institute of Research and Development, Duy Tan University, Da Nang, 550000, Viet Nam*
[b]*Khoa Môi trường và Khoa học tự nhiên, Trường Công nghệ và Kỹ thuật, Đại học Duy Tân, Đà Nẵng, Việt Nam*
[b]*Faculty of Environmental and Natural Sciences, School of Engineering and Technology, Duy Tan University, Da Nang, 550000, Viet Nam*

(Ngày nhận bài: 31/10/2025, ngày phản biện xong: 19/01/2026, ngày chấp nhận đăng: 05/02/2026)

Tóm tắt

Phương trình Poisson là phương trình vi phân bậc hai thường được sử dụng để tính toán điện thế dựa trên mật độ điện tích của một hệ. Trong lĩnh vực mô phỏng, phương pháp sai phân hữu hạn (FDM) thường được sử dụng để đơn giản hóa hệ phương trình. Gauss-Seidel, Jacobi và SOR là ba phương pháp lặp chủ yếu dùng để giải số hệ phương trình Poisson. Lựa chọn giá trị sai số cực đại cho phép hợp lý hoặc tiến hình song song hóa câu lệnh để chạy trên siêu máy tính có thể rút gọn đáng kể thời gian máy tính giải phương trình nhằm tối ưu hóa chương trình mô phỏng cho những hệ phức tạp.

Từ khóa: phương trình Poisson, Gauss-Seidel, Jacobi, SOR

Abstract

Poisson's equation is a second-order partial differential equation which is used to compute the electric potential based on the charge density of a system. In numerical study, the finite difference method is widely used to simplify the equations. Gauss-Seidel, Jacobi and SOR methods are three common iterative methods for solving Poisson's equation. Choosing suitable values of maximum error or parallelizing the code for running on a supercomputer can drastically reduce computation time in order to optimize the code for complicated systems.

Keywords: Poisson's equation, Gauss-Seidel, Jacobi, SOR

1. Giới thiệu

Trong lĩnh vực vật lý và kĩ sư, để tính toán điện trường **E** thông qua mật độ điện tích, người ta sử dụng công thức:

$$\nabla \mathbf{E} = \frac{\rho}{\varepsilon_0} \quad (1),$$

trong đó ρ là mật độ điện tích và ε_0 là độ điện thẩm của chân không. Trong một số bài toán, phụ thuộc vào các điều kiện vùng biên ngoại

Tác giả liên hệ: Lê Thị Quỳnh Trang
Email: letquynhtrang4@duytan.edu.vn

phức tạp, khác nhau mà chúng ta khó có thể tính toán điện trường từ phương trình 1 một cách trực tiếp mà cần tìm giá trị của điện thế Φ trước để thông qua đó xác định giá trị của điện trường dựa theo mối liên hệ $\mathbf{E} = -\boldsymbol{\nabla}\Phi$. Do đó, từ phương trình biểu thị mối liên hệ giữa điện trường và mật độ của điện tích, ta có thể viết lại dưới dạng phương trình vi phân bậc 2 theo điện thế dưới dạng:

$$\nabla^2\Phi = -\frac{\rho}{\varepsilon_0} \quad (2).$$

Phương trình này được gọi là phương trình Poisson [1]. Ở môi trường không có sự chênh lệch điện tích giữa các hạt mang điện, phương trình 2 trở thành phương trình Laplace [1-2]. Phương trình Poisson là một trong những phương trình cơ bản, thông dụng, thường được sử dụng trong các bài toán liên quan tới điện thế dựa trên mật độ điện tích. Phương trình Poisson khi được phân tích thành phương trình vi phân riêng từng phần theo ba chiều $(\mathrm{x}, \mathrm{y}, \mathrm{z})$ tọa độ không gian hệ trục tọa độ Descartes có dạng:

$$\frac{\partial^2\Phi}{\partial \mathrm{x}^2} + \frac{\partial^2\Phi}{\partial \mathrm{y}^2} + \frac{\partial^2\Phi}{\partial \mathrm{z}^2} = -\frac{\rho(\mathrm{x},\mathrm{y},\mathrm{z})}{\varepsilon_0} \quad (3).$$

Có rất nhiều cách khác nhau được đưa ra để giải phương trình này [3-8]. Báo cáo này đề cập đến các phương pháp số giải phương trình Poisson trong mô phỏng. Đối với các bài toán mô phỏng, phương trình Poisson được vận dụng để tính toán điện trường được tạo ra khi các hạt mang điện tích dịch chuyển. Ở những dạng mô hình này, điện thế ở các vùng biên được xác định trước, hoặc có các điều kiện rõ ràng hơn so với điện trường. Chính vì vậy, sử dụng phương trình Poisson được mang lợi thế hơn nhiều so với phương trình điện trường (phương trình 1). Mục 2 liệt kê một vài phương pháp thông dụng được sử dụng khi giải phương trình Poisson ở các bài toán mô phỏng. Mục 3 đề xuất một số thủ thuật dùng để rút gọn thời gian máy tính xử lý câu lệnh khi giải phương trình Poisson. Kết luận về phương trình Poisson và cách giải của nó được đề cập ở mục 4.

2. Các phương pháp số thường gặp để giải phương trình Poisson

Giống như cách giải phương trình vi phân riêng từng phần khác, một trong những phương pháp số lâu đời và đơn giản nhất để giải các bài toán số cho phương trình Laplace hoặc Poision gọi là phương pháp sai phân hữu hạn (FDM) [7]. Đây là một phương pháp số dùng để tính xấp xỉ nghiệm của phương trình vi phân riêng từng phần bằng việc biến đổi một cách gần đúng các đạo hàm thành các sai phân hữu hạn trong từng miền nghiên cứu. Miền nghiên cứu sẽ được chia thành các ô nhỏ hay các lưới. Độ chính xác của phương pháp này sẽ phụ thuộc vào kích thước và hình dạng của các lưới. Đây là phương pháp dễ thực hiện, dễ học và phù hợp cho các phép mô phỏng tính toán hạt có kích thước miền cố định, hay có dạng hình chữ nhật. Đối với các hệ thống vùng miền có hình dạng phức tạp, phương pháp FDM gặp nhiều hạn chế. Trong trường hợp đó, phương pháp phần tử hữu hạn (FEM) được sử dụng nhiều hơn. FEM cho phép giải các phương trình vi phân đạo hàm trên các miền xác định có hình dạng bất kỳ bằng cách chia nhỏ vùng miền thành các phần tử hữu hạn khác nhau. Việc sử dụng FEM cũng yêu cầu kiến thức chuyên ngành cao hơn và tốn khá nhiều thời gian cho việc tìm hiểu, sử dụng nó. Để tối ưu hóa thời gian và hiệu quả thực hiện, tùy vào mỗi bài toán khác nhau, phương pháp phù hợp được sử dụng sẽ khác nhau.

Đối với những bài toán mô phỏng chuyển động của các hạt plasma có vùng miền xác định có hình dạng đơn giản, phương pháp FDM thường được sử dụng phổ biến. Dựa theo quy tắc 3 điểm (đối với hệ một chiều không gian) và 5 điểm (đối với không gian hai chiều), FDM sẽ thay đổi phương trình Poisson liên tục thành hệ các phương trình rời rạc. Các điểm được chọn để đưa vào mỗi phương trình là các điểm gần kề với điểm được quy ước làm mốc. Quy tắc lấy các điểm này được mô tả như ở Hình 1. Với mỗi

điểm tại vị trí thứ i bất kỳ (ngoại trừ các khu vực biên) luôn có hai điểm gần kề bên trái và phải cấu thành nên một phương trình trong hệ không gian một chiều và có bốn điểm gần kề ở vị trí bên trái, bên phải, phía trên và phía dưới đối với hệ không gian hai chiều. Đối với hệ không gian ba chiều, quy tắc lấy điểm gần kề được tiến hành tương tự cho chiều còn lại. Trong các bài toán giải phương trình ở những phần tiếp theo trong báo cáo này, hệ không gian hai chiều được sử dụng làm ví dụ để cho dễ hiểu. Từ phương trình số (3), áp dụng quy tắc ba điểm gần kề cho đạo hàm bậc hai theo phương x, ta có:

$$\frac{\partial^2 \Phi_{i,j}}{\partial x^2} \approx \frac{\Phi_{i-1,j} - 2\Phi_{i,j} + \Phi_{i+1,j}}{\Delta x^2} \quad (4).$$

Phương trình theo phương y được thực hiện tương tự. Như vậy, phương trình Poisson trong không gian hai chiều khi viết lại dưới dạng phương trình năm điểm có dạng:

$$\frac{\Phi_{i-1,j} - 2\Phi_{i,j} + \Phi_{i+1,j}}{\Delta x^2} + \frac{\Phi_{i,j-1} - 2\Phi_{i,j} + \Phi_{i,j+1}}{\Delta y^2} = -\frac{\rho_{i,j}}{\varepsilon_0} \quad (5).$$

Độ chính xác của phương trình này phụ thuộc vào bước nhảy không gian $\Delta x, \Delta y$. Giá trị bước nhảy không gian càng nhỏ thì độ chính xác của các nghiệm thu được càng cao. Tuy nhiên, mật độ lưới càng dày thì số lượng điểm cần xét, hay số lượng phương trình cần giải trở thành một con số không nhỏ. Đối với các bài toán mô phỏng hạt plasma, giá trị của điện thế tại mỗi điểm $\Phi_{i,j}$ chỉ phụ thuộc vào mật độ điện tích $\rho_{i,j}$ tại điểm đó và điện thế tại bốn điểm gần kề (như phương trình 5). Giả sử Nx, Ny lần lượt là số điểm lưới được sử dụng theo phương x, y của khu vực đang xét. Tập hợp N ($N = Nx \times Ny$) phương trình điện thế cấu thành từ mỗi điểm i,j tạo thành một ma trận (hệ phương trình tuyến tính). Có nhiều cách để giải hệ phương trình này như sử dụng phép khử Gauss. Trong mô phỏng, phương pháp thông dụng thường được sử dụng đó là phương pháp lặp [5-6]. Phương pháp lặp là phương pháp số nhằm giải quyết bài toán bằng cách gán một giá trị ban đầu cho các nghiệm trong hệ phương trình cần giải, sau đó lặp đi lặp lại một chuỗi các bước tính nghiệm phương trình giống nhau sao cho kết quả thu được sau mỗi bước tiến gần sát đến kết quả chính xác cuối cùng. Các phép biến đổi phổ biến khi dùng kĩ thuật lặp trong giải phương trình Poisson bao gồm Jacobi, Gauss-Seidel và Successive over-Relaxation (SOR). Giả sử rằng bước nhảy không gian theo phương x, y bằng nhau ($\Delta x = \Delta y = h$), từ phương trình (5) giá trị điện thế tại mỗi điểm $\Phi_{i,j}$ được viết lại theo biểu thức:

$$\Phi_{i,j} = \frac{1}{4}\left(\Phi_{i-1,j} + \Phi_{i+1,j} + \Phi_{i,j-1} + \Phi_{i,j+1}\right) + \frac{1}{4}\frac{\rho_{i,j}}{\varepsilon_0}h^2 \ . \ (6)$$

Phương pháp Jacobi bắt đầu bằng biệc gán cho các đại lượng cần tìm của hệ phương trình những nghiệm ước đoán ban đầu (có thể chọn bằng không). Giá trị của $\Phi_{i,j}$ ở lần tính toán thứ $k+1$ được cập nhập dựa trên các giá trị của những điểm gần kề ở thời điểm tính toán thứ k [5-8]:

$$\Phi_{i,j}^{k+1} = \frac{1}{4}\left(\Phi_{i-1,j}^{k} + \Phi_{i+1,j}^{k} + \Phi_{i,j-1}^{k} + \Phi_{i,j+1}^{k}\right) + \frac{1}{4}\frac{\rho_{i,j}}{\varepsilon_0}h^2 \ . \ (7)$$

Phép tính này được thực hiện cho mỗi điểm trên lưới của vùng miền đang xét và lặp đi lặp lại nhiều lần cho đến khi sự chênh lệch kết quả giữa hai lần tính toán liền kề nhỏ hơn sai số cho phép. Hay nói cách khác thì phép tính này được thực hiện lặp lại cho đến khi giá trị của $\Phi_{i,j}$ ở lần tính toán thứ $k+1$ và lần thứ k không có sự chênh lệch quá lớn. Khi đó, giá trị cuối cùng của

$\Phi_{i,j}^{k+1}$ chính là nghiệm xấp xỉ của hệ phương trình. Jacobi là một phương pháp số đơn giản, dễ thực hiện nhưng tốn khá nhiều thời gian để hệ tìm ra được nghiệm cuối cùng trong phạm vi sai số cho phép. Phương pháp Gauss-Seidel được thực hiện tương tự như phương pháp Jacobi. Điểm khác biệt ở đây là một khi giá trị nào vừa được cập nhập thì nó sẽ tham gia ngay vào quá trình cập nhập giá trị của các điểm gần kề bên cạnh [5-8]. Cụ thể, giá trị tại $\Phi_{i,j}^{k+1}$ được cập nhập theo biểu thức:

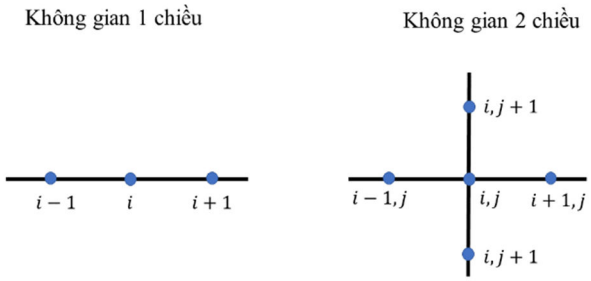
$$\Phi_{i,j}^{k+1} = \frac{1}{4}\left(\Phi_{i-1,j}^{k+1} + \Phi_{i+1,j}^{k} + \Phi_{i,j-1}^{k+1} + \Phi_{i,j+1}^{k}\right) + \frac{1}{4}\frac{\rho_{i,j}}{\varepsilon_0}h^2 \quad . \quad (8)$$

Nhờ vào việc sử dụng ngay lập tức các giá trị vừa được cập nhập của các điểm gần kề, Gauss-Seidel cho phép rút ngắn thời gian ma trận hội tụ để thu được kết quả mô phỏng cuối cùng trong phạm vi sai số cho phép nhanh hơn nhiều so với phương pháp Jacobi. Tuy nhiên phương pháp này vẫn là một phương pháp tốn nhiều thời gian để tính toán.

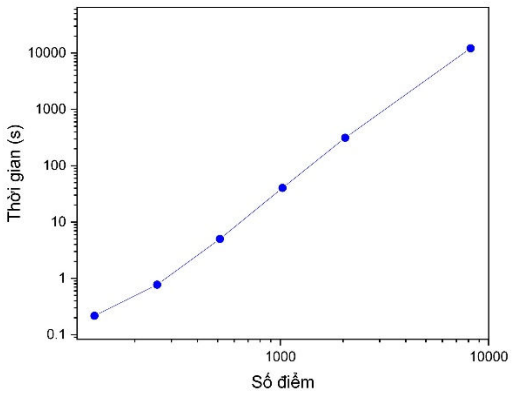
Nhằm tăng tốc độ hội tụ và giảm số bước lặp trong các phương pháp cơ bản, phương pháp SOR được cải thiện dựa trên phương pháp Jacobi và Gauss-Seidel. Trong trường hợp này, hệ số tăng tốc w $(0 < w < 2)$ được thêm vào phương trình (5) để gia tốc thời gian tìm nghiệm của hệ phương trình theo dạng [5-8]:

$$\Phi_{i,j}^{k+1} = (1-w)\Phi_{i,j}^{k} + \frac{1}{4}w\left(\Phi_{i-1,j}^{k+1} + \Phi_{i+1,j}^{k} + \Phi_{i,j-1}^{k+1} + \Phi_{i,j+1}^{k} + \frac{1}{4}\frac{\rho_{i,j}}{\varepsilon_0}h^2\right). \quad (9)$$

Hệ số tăng tốc w phụ thuộc vào hình dạng của vùng miền nghiên cứu, kích thước của các lưới, cũng như loại điều kiện biên đang sử dụng. w được chọn sao cho tốc độ hội tụ của hệ đạt được cực đại, giá trị tối ưu cần chọn tùy thuộc vào số điểm của hệ theo mỗi phương [9]. Nếu $w = 1$ thì phương trình (9) chính là phương trình Gauss-Seidel. Phương pháp SOR giúp rút ngắn thời gian đáng kể để đưa ra nghiệm xấp xỉ gần đúng và đây cũng là phương pháp dễ dàng thực hiện trong nghiên cứu số. Đặc biệt, phương pháp này phù hợp áp dụng đối với các hệ phương trình có cấu trúc đặc biệt. Do đó, trong các mô hình mô phỏng hạt, phương pháp SOR thường được sử dụng nhiều hơn. SOR được biến tấu theo nhiều cách áp dụng khác nhau. SOR có thể được thực hiện theo cách thông thường như lần lượt xét tất cả các điểm có trên lưới của vùng miền xác định theo thứ tự, hoặc phân tách ra các điểm chẵn lẻ riêng biệt để thực hiện [10]. Tuỳ thuộc vào điều kiện biên, hình dạng vùng miền xác định, phương thức khác nhau được sử dụng khi xây dựng chương trình mô phỏng. Ngoài những phương pháp đã liệt kê, vẫn còn nhiều phương pháp khác để giải số phương trình Poisson, hoặc sử dụng các gói hoặc thư viện có sẵn cho các ngôn ngữ lập trình khác nhau (ví dụ như PoisFFT được sử dụng cho C++, PETSc có thể sử dụng cho cả Fortran, Python, C++) [11-12]. Ba phương pháp được đề cập là ba phương pháp thông dụng được dùng cho nhiều ngôn ngữ lập trình. Thời gian và sai số đem đi so sánh trong báo cáo này được sử dụng dựa trên ngôn ngữ Fortran.



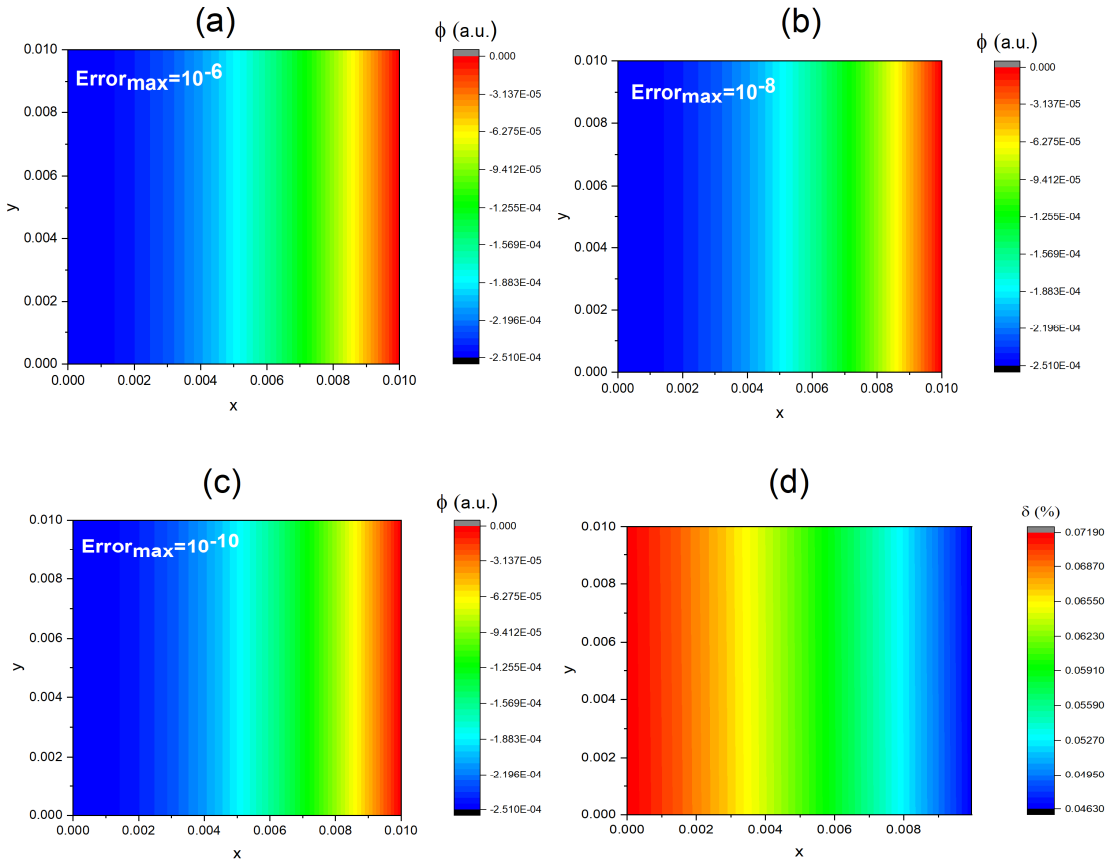
Hình 1.Các điểm gần kề trong không gian một chiều và hai chiều



Hình 2. Mối liên hệ giữa thời gian CPU xử lý chương trình và số lượng điểm lưới cần sử dụng.

3. Tối ưu hóa phương pháp giải phương trình Poisson

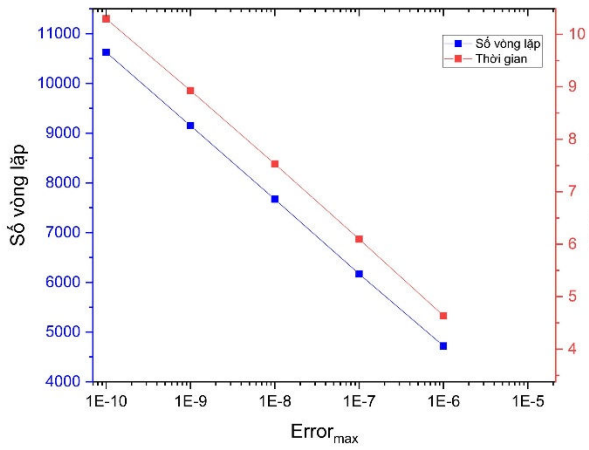
Đối với vùng miền nghiên cứu có kích thước lớn, hoặc có hình dạng phức tạp, số điểm lưới được sử dụng nhiều hơn. Ví dụ như muốn mô phỏng sự chuyển động của hạt plasma trong ở khu vực rìa lò phản ứng nhiệt hạch theo mô hình mô phỏng Particle-in Cell (PIC), thông thường với vùng miền xác định ở không gian hai chiều có diện tích $1\text{m} \times 1\text{m}$, cần sử dụng ít nhất 10.000×10.000 điểm để đảm bảo hệ mô phỏng duy trì tính ổn định và cho ra kết quả có độ chính xác cao, giả sử mật độ các hạt plasma xấp xỉ 10^{-20}m^{-3}. Như vậy hệ phương trình Poisson có xấp xỉ 10^8 phương trình cần giải (bao gồm cả điều kiện biên). Đây là một số lượng lớn phương trình, và tốn khá nhiều thời gian để đưa ra nghiệm cuối cùng cho hệ này. Hình 2 mô tả mối liên hệ giữa số lượng điểm lưới sử dụng và thời gian cần thiết để một CPU giải phương trình theo phương pháp SOR. Giả sử rằng hệ dùng để mô phỏng là một hệ đơn giản, giá trị $\rho_{i,j}$ là hằng số tại mọi điểm, khi số lượng điểm lưới của hệ tăng lên, kéo theo số lượng lớn hệ phương trình cần giải, CPU tốn nhiều thời gian hơn để giải quyết vấn đề. Như vậy, đối với một hệ mô phỏng có điều kiện đầu vào đơn giản, thời gian bình quân để giải hệ phương trình này bằng phương pháp SOR cần vài đến vài chục thậm chí vài trăm giây. Trong trường hợp chỉ yêu cầu một lần thực hiện phép tính để đưa ra nghiệm thì thời gian vài giây không cấu thành vấn đề. Tuy nhiên trong phép mô phỏng PIC, khi mô phỏng chuyển động của các hạt plasma, vị trí hay mật độ của chúng thay đổi và được cập nhập mới theo thời gian, mật độ điện tích $\rho_{i,j}$ biến đổi phức tạp. Phương trình Poisson được triển khai trong mỗi một bước nhảy thời gian xuyên suốt quá trình diễn ra phép mô phỏng. Giả sử hệ mô phỏng cần 10^5 bước nhảy thời gian để hệ tiến tới trạng thái cân bằng, và ổn định, tổng thời gian để CPU xử lý cho phương trình Poisson tạo thành một con số lớn, tiêu tốn thời gian vài ngày. Chưa kể đến mô hình PIC còn tiêu tốn thời gian cho các quá trình khác như cập nhập vị trí, vận tốc của hạt, xử lý va chạm đàn hồi giữa các hạt, v.v... Chính vì vậy, mặc dù phương pháp SOR đã giảm thiểu thời gian đáng kể so với những phương pháp số khác khi giải phương trình Poisson, việc nghiên cứu làm sao để tối ưu hóa, nâng cao hiệu suất của chương trình vẫn là vấn đề được quan tâm.



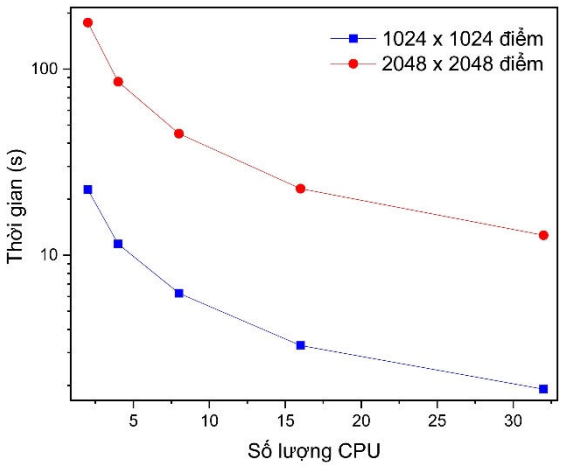
Hình 3. Giá trị điện thế thu được khi sử dụng sai số cực đại cho phép khác nhau.

Như đã đề cập, khi giải phương trình Poisson bằng phương pháp SOR nói riêng cũng như phương pháp lặp nói chung, vòng lặp sẽ được lặp lại cho đến khi giá trị thu được ở hai lần tính toán liền kề không chênh lệch nhau quá lớn. Thông thường, có hai cách để kết thúc vòng lặp để đưa ra nghiệm. Cách đầu tiên là cho hệ chạy chính xác N vòng lặp và dừng lại. Cách này đối với hệ mô phỏng đơn giản thì có thể dự đoán được giá trị của N bằng bao nhiêu. Đối với hệ phức tạp, việc chọn N có thể gây ra quá dư thừa (nếu lớn hơn nhiều với giá trị cần dùng thực để hệ phương trình hội tụ) hoặc chưa đủ lớn để hệ đưa ra được kết quả chính xác mà không chêch lệch quá lớn so với lần tính toán liền kề trước đó. Do đó, cách này thường ít khi được sử dụng. Cách thứ 2 là sử dụng sai số cực đại cho phép ($Error_{Max}$). Giá trị của tất cả các điểm thu được ở hai lần tính toán liên tiếp phải nhỏ hơn sai số cực đại cho phép thì có thể kết thúc vòng lặp và đưa ra nghiệm của hệ phương trình. Sai số cực đại được chọn có giá trị càng nhỏ thì kết quả tính toán mô phỏng thu được có giá trị càng chính xác. Tuy nhiên, sai số cực đại càng nhỏ thì số vòng lặp cần thực hiện càng lớn, dẫn đến thời gian cần thiết cho máy tính xử lý càng dài. Vì vậy, việc chọn lựa giá trị của sai số cực đại cho phép cũng ảnh hưởng tới thời gian tính toán. Thông thường, sai số cực đại cho phép được chọn lựa sao cho kết quả thu được vẫn nằm trong khoảng chấp nhận mà không tiêu tốn quá nhiều thời gian xử lý. Hình 3 so sánh kết quả mô phỏng thu được khi sử dụng các giá trị khác nhau của sai số cực đại cho phép khi sử dụng phương pháp SOR để giải phương trình Poisson. Hệ đang xét sử dụng 256×256 điểm lưới cho hai chiều không gian và giá trị của mật độ điện tích $\rho_{i,j}$ là hằng số cho mọi điểm lưới. Ba trường hợp khác nhau được đem ra so sánh:

$\text{Error}_{\text{Max}} = 10^{-6}, 10^{-8}, 10^{-10}$ được mô tả lần lượt ở Hình 3 a, b, và c. Kết quả điện thế thu được được từ các sai số cực đại cho phép khác nhau này cho thấy về cơ bản, điện thế Φ của hệ thu được đều có xu hướng và giá trị gần như nhau. Hình 3d mô tả tỉ số chênh lệch giá trị giữa hai nghiệm thu được khi sử dụng sai số cực đại cho phép là 10^{-6} và 10^{-10} ($\delta = \frac{|\Phi_{\text{Error}_{\text{Max}}=10^{-6}} - \Phi_{\text{Error}_{\text{Max}}=10^{-10}}|}{|\Phi_{\text{Error}_{\text{Max}}=10^{-10}}|}$). Tỉ số này có giá trị nhỏ hơn 0,1%. Điều này chứng tỏ rằng không có sự sai lệch lớn giữa các nghiệm thu được khi chọn giá trị sai số cực đại khác nhau. Mối liên hệ giữa sai số cực đại khác nhau và số vòng lặp cần thực hiện hay thời gian CPU xử lý thông tin được thể hiện ở Hình 4. Khi sai số cực đại cho phép càng lớn thì số vòng lặp cần thực hiện hay thời gian CPU xử lý thông tin càng giảm xuống. Chính vì vậy, nếu như một hệ mô phỏng không cần đưa ra kết quả vô cùng chính xác tới vài chục số sau dấu thập phân, việc sử dụng sai số cực đại hợp lý cho phép rút ngắn đáng kể thời gian CPU giải hệ phương trình Poisson.



Hình 4. Mối liên hệ giữa số vòng lặp cần thực hiện (màu xanh) và thời gian CPU thực hiện phép tính (màu đỏ) với những giá trị khác nhau của sai số cực đại cho phép ($\text{Error}_{\text{Max}}$).



Hình 5. Thời gian hoàn thành tính toán khi sử dụng tăng dần số lượng CPU đem vào sử dụng ở hệ không gian hai chiều với 1024×1024 (màu xanh) và 2048×2048 (màu đỏ) điểm.

Sự xuất hiện của siêu máy tính cho phép một chuỗi các câu lệnh có thể chạy song song trên nhiều CPU cùng lúc. Từ đó rút ngắn thời gian hay tốc độ máy tính tính toán xử lý câu lệnh. Đối với những bài toán phức tạp, khổng lồ thay vì chạy trên một CPU thông thường, việc song song hóa chương trình mô phỏng sẽ chia nhỏ bài toán ra thành chuỗi câu lệnh giống nhau và cùng đồng thời thực hiện trên chuỗi các máy tính, luồng khác nhau mà không làm thay đổi kết quả thu được. Khi công việc cần xử lý được phân tán tới nhiều máy tính khác nhau để thực hiện, nó sẽ rút ngắn đáng kể thời gian tính toán so với chỉ chạy trên một CPU bình thường. OpenMP và MPI là những thủ thuật phổ biến dùng cho việc song song hóa chương trình. Hình 5 so sánh thời gian cần thiết để đưa ra cùng một kết quả khi sử dụng số lượng CPU khác nhau cho một hệ phương trình hai chiều. Sự giảm của thời gian thực hiện tỷ lệ thuận với sự tăng lên của số CPU đem đi dùng. Điều này tăng khả năng thực thi cho việc mô phỏng các hệ có kích thước lớn hay mô hình phức tạp, giải quyết một trong những khó khăn về thời gian khi xây dựng các mô hình mô phỏng. Tuy nhiên khi cần lưu ý tới số lượng các luồng máy tính, số CPU đem đi dùng để tối ưu hóa hiệu suất sử dụng. Việc sử dụng nhiều CPU cần lưu ý tới thời gian trao đổi thông tin giữa các CPU với nhau. Nếu việc trao đổi này diễn ra giữa quá nhiều CPU cho một hệ kích

thước nhỏ thì tổng thời gian giải quyết vấn đề có thể lâu hơn khi chỉ dùng ít CPU. Chính vì vậy, nên tính toán, thiết kế, phân bổ số lượng CPU hay chọn lựa phương pháp giải phù hợp cho từng bài toán cụ thể riêng biệt.

4. Kết luận

Để giải phương trình Poisson trong chương trình mô phỏng có rất nhiều phương pháp khác nhau, trong đó thông dụng nhất là phương pháp lặp. Phương pháp lặp là phương pháp dựa trên việc sử dụng sai phân hữu hạn nên được sử dụng ở nhiều ngôn ngữ lập trình khác nhau. Jacobi, Gauss-Seidel hay SOR là những phương pháp tiêu biểu trong việc sử dụng vòng lặp để giải phương tình Poisson với kết quả thu được có độ chính xác cao so với giá trị lý thuyết. Tùy vào điều kiện bài toán khác nhau, hoặc yêu cầu của phép tính để chọn phương thức hợp lý. Đối với những bài toán cần chú trọng thời gian máy tính phân tích, xử lý câu lệnh, có thể lựa chọn giá trị sai số cực đại khác nhau phù hợp nhằm giảm thời gian mà không ảnh hưởng tới tổng thể nghiệm thu được của phương trình. Tùy thuộc vào điều kiện cơ sở vật chất, việc xây dựng song song hóa chương trình và chạy trên nhiều CPU cùng lúc ở siêu máy tính có thể giảm thiểu đáng kể thời gian giải phương trình.

Lời cảm ơn

Một số kết quả trong báo cáo này được thực hiện trên "Plasma Simulator" (NEC SX-Aurora TSUBASA) thuộc Viện Nghiên cứu Phản ứng Nhiệt hạch (NIFS, Nhật Bản). Tôi xin được gửi lời biết ơn chân thành nhất tới những người đã hướng dẫn và giúp đỡ tôi hoàn thành nghiên cứu này: GS. Yasuhiro Suzuki (Đại học Hiroshima, Nhật Bản), TS. Hiroaki Ohtani (NIFS, Nhật Bản), TS. Hiroki Hasegawa (NIFS), TS. Toseo Moritaka (NIFS).

Tài liệu tham khảo

[1] Hackbusch, W. (2017). Elliptic differential equations: theory and numerical treatment (Vol. 18). *Springer*.

[2] Venkateshan, S. P., & Swaminathan, P. (2014). Computational methods in engineering (pp. 317-373). *Cambridge, MA, USA: Academic Press*.

[3] Moghaderi, H., Dehghan, M., & Hajarian, M. (2016). A fast and efficient two-grid method for solving d-dimensional Poisson equations. *Numerical Algorithms*, 72(3), 483-537.

[4] Dorr, F. W. (1970). The direct solution of the discrete Poisson equation on a rectangle. *SIAM review*, 12(2), 248-263.

[5] Schäfer, M. (2006). Computational engineering—introduction to numerical methods. *Berlin, Heidelberg: Springer Berlin Heidelberg.*

[6] Griffiths, D. V., & Smith, I. M. (2006). Numerical methods for engineers. *Chapman and Hall/CRC.*

[7] Nagel, J. R. (2011). Solving the generalized Poisson equation using the finite-difference method (FDM). Lecture Notes, Dept. of Electrical and Computer Engineering, University of Utah, 52.

[8] Moiseienko, S., Tuchyna, U., Redchyts, D., Zaika, V., & Vygodner, I. (2021). Comparative analysis of numerical methods for solving linear equation systems for Poisson's equation. *In International Conference on Advanced Mechanical and Power Engineerin*g (pp. 169-177). Cham: Springer International Publishing.

[9] Yang, S., & Gobbert, M. K. (2009). The optimal relaxation parameter for the SOR method applied to the Poisson equation in any space dimensions. *Applied mathematics letters*, 22(3), 325-331.

[10] Kuo, C. C. J., & Levy, B. C. (1989). A two-level four-color SOR method. *SIAM journal on numerical analysis*, 26(1), 129-151.

[11] Zhang, J. (2015). A PETSc-Based Parallel Implementation of Finite Element Method for Elasticity Problems. *Mathematical Problems in Engineering*, 2015(1), 147286.

[12] Fuka, V. (2015). Poisfft–a free parallel fast poisson solver. *Applied Mathematics and Computation*, 267, 356-364.