



## Parallel Computation Using Non-Overlapping Domain Decomposition Coupled with Compact Local Integrated RBF for Navier–Stokes Equations

Nam Pham-Sy & Canh-Dung Tran

To cite this article: Nam Pham-Sy & Canh-Dung Tran (2022) Parallel Computation Using Non-Overlapping Domain Decomposition Coupled with Compact Local Integrated RBF for Navier–Stokes Equations, International Journal of Computational Fluid Dynamics, 36:10, 835-856, DOI: [10.1080/10618562.2023.2229250](https://doi.org/10.1080/10618562.2023.2229250)

To link to this article: <https://doi.org/10.1080/10618562.2023.2229250>



© 2023 The Author(s). Published by Informa UK Limited, trading as Taylor & Francis Group.



Published online: 28 Jun 2023.



Submit your article to this journal [↗](#)



Article views: 139




View related articles [↗](#)



View Crossmark data [↗](#)

# Parallel Computation Using Non-Overlapping Domain Decomposition Coupled with Compact Local Integrated RBF for Navier–Stokes Equations

Nam Pham-Sy and Canh-Dung Tran 

School of Engineering, Faculty of Health, Engineering and Sciences, University of Southern Queensland, Toowoomba, Australia

## ABSTRACT

A non-overlapping domain decomposition-based parallel algorithm coupled with a compact local integrated radial basis function (CLIRBF) method is developed for solving Navier-Stokes equations. For this approach, a problem is divided into subdomains. In each sub-domain, a CLIRBF scheme is applied to solve the Navier-Stokes equations of flows. A relaxation factor is used at the interface between sub-domains to ensure the quick convergence of the present method. The Bitmap termination detection technique is introduced to complete the global termination. The present approach is verified using two fluid flow problems: the lid-driven cavity and the natural convection in concentric annuli flow. The numerical results have demonstrated the efficiency of the present parallel method compared with the corresponding sequential one and other published methods. Especially, super-linear speed-up was achieved for several CPUs. In terms of accuracy, the obtained results are in very good agreement with benchmark results.

## ARTICLE HISTORY

Received 27 September 2022  
Accepted 19 June 2023

## KEYWORDS

Parallel algorithm;  
non-overlapping domain  
decomposition method;  
compact local integrated  
radial basis function;  
Navier–Stokes equations

## 1. Introduction

Domain decomposition (DD) scheme, also called as Schwarz alternating DD method, is a sufficient way to deal with large-scale problems (Schwarz 1869). This approach's idea is to split a considered domain into smaller sub-domains and the problem is then solved in each sub-domain separately. On the artificial boundaries between sub-domains, Dirichlet boundary conditions (BCs) are set up for the start-up time and updated for the next repeat times using the obtained values from neighbouring sub-domains. This approach is applied for a group of DD methods, characterised by the fact that adjacent sub-domains must overlap with each other (Smith, Bjorstad, and Gropp 1996). That yields (i) difficulties in solving problems with irregular domains and (ii) an increase in the overhead of degrees of freedom to each sub-domain.

Non-overlapping DD methods are classified into two sub-groups: the Schur complement and the Steklov–Poincare methods (Quarteroni and Valli 1999). The Schur complement method divides a structure into substructures in which all artificial boundaries are fixed (Haynsworth 1968). The actual value of field

variables on artificial boundaries is determined from the equilibrium equations of forces (Gander and Tu 2014). The problem is then solved independently and separately in substructures. For the Schur method, the computation cost of obtaining solutions on the artificial boundaries is much higher in comparison with the benefit by the parallel calculation on substructures. Meanwhile, the Steklov–Poincare method is to construct an equivalent problem by introducing transmission conditions (Quarteroni and Valli 1999). The newly formed problem is then solved separately in sub-domains. Results, which were obtained from sub-domains, are gathered back into the original domain. This approach is better than the overlapping DD scheme because of its minimised degree of freedoms (Tran, Phillips, and Tran-Cong 2009). Furthermore, the method is potential in parallel computation in which all sub-domains can run concurrently but do not cause the bottle-neck as in the Schur complement method. More details on this approach will be presented as a part of our present work in Section 3.

Radial basis function (RBF) and specially, the integrated RBF (IRBF) have been a well-known approximation method owing to its high convergence

**CONTACT** Canh-Dung Tran  [canh-dung.tran@usq.edu.au](mailto:canh-dung.tran@usq.edu.au)

rate (Fasshauer 1999; Mai-Duy and Tran-Cong 2001) and recently applied to efficiently simulate various complicated polymeric liquid problems from non-Newtonian flows (Tran-Canh and Tran-Cong 2004; Tran et al. 2012) to fibre suspensions (Nguyen and Tran 2022; Nguyen, Tran, and Tran-Cong 2015). However, its stability degrades relatively fast with respect to collocation density because of the ill-condition of the corresponding system matrix. This drawback of IRBF is mitigated by combining it with compact local schemes. Numerical results demonstrated that by appropriately choosing the shape parameter  $\beta$ , IRBF compact schemes have achieved a very high convergence rate while maintaining a low condition number of the system matrix (Hoang-Trieu, Mai-Duy, and Tran-Cong 2012). So, in this present work, the non-overlapping DD coupled compact local integrated radial basis function (CLIRBF) method is developed into one parallel algorithm to efficiently solve Navier–Stokes equations. The accuracy and efficiency of the approach are investigated and evaluated by solving two benchmark problems, the lid-driven cavity fluid flow (LDC) and natural convection (NC), whose obtained results are compared with those of the non-parallel method as well as with benchmark results.

For this paper, the two dimensions – IRBF (2D-IRBF) method and a compact local scheme are firstly reviewed in Section 2. Section 3 represents a non-overlapping Dirichlet–Neumann DD method where the present parallel algorithm based on the combination of CLIRBF and DD method is introduced. Numerical results by solving the lid-driven cavity and NC problems compared with benchmark results are provided and discussed in Sections 4 and 5.

## 2. Review on Compact Local 2D-IRBF Method

### 2.1. 2D-IRBF Scheme

Consider a 2D elliptic problem.

$$\begin{cases} Lu(x) = f, x \in \Omega \\ Bu(x) = g, x \in \partial\Omega \end{cases} \quad (1)$$

where  $\mathbf{x}$  is the position variable;  $u(\mathbf{x})$  is an unknown function;  $L$  is the second order differential operator;  $B$  represents the boundary conditions;  $f$  and  $g$  are known functions with respect to  $x$ ;  $\Omega$  is the considered domain and  $\partial\Omega$  is the boundary.

With the 2D-IRBF discretisation scheme, the second derivatives of function  $u$  are expressed as a combination of RBFs (Mai-Duy and Tran-Cong 2010; Tran-Canh and Tran-Cong 2002)

$$\frac{\partial^2 u(\mathbf{x})}{\partial x_j^2} = \sum_{i=1}^n w_i g_i(\mathbf{x}) = \sum_{i=1}^n w_i(t) G_i^{[2]}(\mathbf{x}) \quad (2)$$

where  $x_j$  is the  $j$ -component of  $x$  ( $j = 1, 2$ );  $\{w_i\}_{i=1}^n$  is the set of weights; and  $\{g_i(\mathbf{x})\}_{i=1}^n$  is the set of RBFs associated with  $n$  centres.

In this work, the centres are chosen to be the grid points and the multiquadric RBF (MQ-RBF) function is used and given by  $G_i^{[2]}(\mathbf{x}) = \sqrt{(\mathbf{x} - \mathbf{c}_i)^2 + a_i^2}$ , where  $\{\mathbf{c}_i\}_{i=1}^n$  and  $\{a_i\}_{i=1}^n$  are MQ-RBF centres and widths, respectively. It is noted that superscript  $^{[1]}$  is used to denote the associated derivative order.

The first-order derivative and the function are directly integrated in Equation (2) with respect to  $x_j$  as follows.

$$\frac{\partial u}{\partial x_j} = \sum_{i=1}^n w_i G_i^{[1]}(\mathbf{x}) + C_1 \quad (3)$$

$$u = \sum_{i=1}^n w_i G_i^{[0]}(\mathbf{x}) + C_1 x_j + C_2 \quad (4)$$

where  $G_i^{[1]}(\mathbf{x}) = \int G_i^{[2]}(\mathbf{x}) dx_j$ ,  $G_i^{[0]}(\mathbf{x}) = \int G_i^{[1]}(\mathbf{x}) dx_j$  and  $C_1$ , and  $C_2$  are integral constants with  $C_i = C_i(x_k)$ ,  $k \neq j$ .

Collocating Equations (2–4) at the grid points  $\{\mathbf{x}_i\}_{i=1}^n$  yields.

$$\frac{\partial^2 \tilde{\mathbf{u}}}{\partial^2 x_j} = \varsigma_{x_j}^{[2]} \tilde{\mathbf{w}}_{x_j} \quad (5)$$

$$\frac{\partial \tilde{\mathbf{u}}}{\partial x_j} = \varsigma_{x_j}^{[1]} \tilde{\mathbf{w}}_{x_j} \quad (6)$$

$$\tilde{\mathbf{u}} = \varsigma_{x_j}^{[0]} \tilde{\mathbf{w}}_{x_j} \quad (7)$$

With  $\tilde{\mathbf{w}}_{x_j} = (w_1, w_2, \dots, w_n, C_1, C_2)_{x_j}^T$ ;  $\tilde{\mathbf{u}} = (u_1, u_2, \dots, u_n)^T$ ;  $\frac{\partial^k \tilde{\mathbf{u}}}{\partial x_j^k} = \left( \frac{\partial^k u_1}{\partial x_j^k}, \frac{\partial^k u_2}{\partial x_j^k}, \dots, \frac{\partial^k u_m}{\partial x_j^k} \right)^T$  where  $u_i = u(\mathbf{x}_i)$  ( $i = 1, 2, \dots, n$ );  $\varsigma_{x_j}^{[2]}$ ,  $\varsigma_{x_j}^{[1]}$  and  $\varsigma_{x_j}^{[0]}$  are known matrices.

It is worth noting that the subscript  $x_j$  denotes the quantity associated with the integration process on the  $x_j$  direction. In a 2D problem, Equation (4) generates two approximations for the function  $u$  and they will be naturally forced to be identical.

For the sake of easy following, in the next sections,  $x$  and  $y$  notations will be used to represent dimensions instead of  $x_1$  and  $x_2$  and  $\mathbf{x}_k$  to represent a grid point  $k$ .

## 2.2. Compact Local IRBF Scheme

In general, a set of points of a local scheme called stencil is used for discretisation. In compact local 2D-IRBF scheme, a '9-point stencil' is described in Figure 1.

For such stencil, Equations (5–7) are explicitly expressed along  $x$ -dimension by

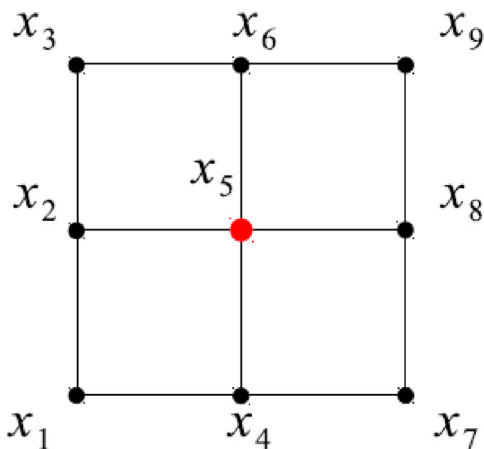
$$\frac{\partial^2 \tilde{u}}{\partial^2 x} = \zeta_x^{[2]} \tilde{\mathbf{w}}_x \quad (8)$$

$$\frac{\partial \tilde{u}}{\partial x} = \zeta_x^{[1]} \tilde{\mathbf{w}}_x \quad (9)$$

$$\tilde{u} = \zeta_x^{[0]} \tilde{\mathbf{w}}_x \quad (10)$$

where

$$\zeta_x^{[2]} = \begin{bmatrix} G_{x_1}^{[2]}(\mathbf{x}_1) & \dots & G_{x_9}^{[2]}(\mathbf{x}_1) & 0 & 0 & 0 & 0 & 0 & 0 \\ G_{x_1}^{[2]}(\mathbf{x}_2) & \dots & G_{x_9}^{[2]}(\mathbf{x}_2) & 0 & 0 & 0 & 0 & 0 & 0 \\ G_{x_1}^{[2]}(\mathbf{x}_3) & \dots & G_{x_9}^{[2]}(\mathbf{x}_3) & 0 & 0 & 0 & 0 & 0 & 0 \\ G_{x_1}^{[2]}(\mathbf{x}_4) & \dots & G_{x_9}^{[2]}(\mathbf{x}_4) & 0 & 0 & 0 & 0 & 0 & 0 \\ G_{x_1}^{[2]}(\mathbf{x}_5) & \dots & G_{x_9}^{[2]}(\mathbf{x}_5) & 0 & 0 & 0 & 0 & 0 & 0 \\ G_{x_1}^{[2]}(\mathbf{x}_6) & \dots & G_{x_9}^{[2]}(\mathbf{x}_6) & 0 & 0 & 0 & 0 & 0 & 0 \\ G_{x_1}^{[2]}(\mathbf{x}_7) & \dots & G_{x_9}^{[2]}(\mathbf{x}_7) & 0 & 0 & 0 & 0 & 0 & 0 \\ G_{x_1}^{[2]}(\mathbf{x}_8) & \dots & G_{x_9}^{[2]}(\mathbf{x}_8) & 0 & 0 & 0 & 0 & 0 & 0 \\ G_{x_1}^{[2]}(\mathbf{x}_9) & \dots & G_{x_9}^{[2]}(\mathbf{x}_9) & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$



**Figure 1.** '9-point stencil' of 2D-IRBF scheme with  $\mathbf{x}_5$  is the point under consideration.

$$\zeta_x^{[1]} = \begin{bmatrix} G_{x_1}^{[1]}(\mathbf{x}_1) & \dots & G_{x_9}^{[1]}(\mathbf{x}_1) & 1 & 0 & 0 & 0 & 0 & 0 \\ G_{x_1}^{[1]}(\mathbf{x}_2) & \dots & G_{x_9}^{[1]}(\mathbf{x}_2) & 0 & 1 & 0 & 0 & 0 & 0 \\ G_{x_1}^{[1]}(\mathbf{x}_3) & \dots & G_{x_9}^{[1]}(\mathbf{x}_3) & 0 & 0 & 1 & 0 & 0 & 0 \\ G_{x_1}^{[1]}(\mathbf{x}_4) & \dots & G_{x_9}^{[1]}(\mathbf{x}_4) & 1 & 0 & 0 & 0 & 0 & 0 \\ G_{x_1}^{[1]}(\mathbf{x}_5) & \dots & G_{x_9}^{[1]}(\mathbf{x}_5) & 0 & 1 & 0 & 0 & 0 & 0 \\ G_{x_1}^{[1]}(\mathbf{x}_6) & \dots & G_{x_9}^{[1]}(\mathbf{x}_6) & 0 & 0 & 1 & 0 & 0 & 0 \\ G_{x_1}^{[1]}(\mathbf{x}_7) & \dots & G_{x_9}^{[1]}(\mathbf{x}_7) & 1 & 0 & 0 & 0 & 0 & 0 \\ G_{x_1}^{[1]}(\mathbf{x}_8) & \dots & G_{x_9}^{[1]}(\mathbf{x}_8) & 0 & 1 & 0 & 0 & 0 & 0 \\ G_{x_1}^{[1]}(\mathbf{x}_9) & \dots & G_{x_9}^{[1]}(\mathbf{x}_9) & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

$$\zeta_x^{[0]} = \begin{bmatrix} G_{x_1}^{[0]}(\mathbf{x}_1) & \dots & G_{x_9}^{[0]}(\mathbf{x}_1) & x_1 & 0 & 0 & 1 & 0 & 0 \\ G_{x_1}^{[0]}(\mathbf{x}_2) & \dots & G_{x_9}^{[0]}(\mathbf{x}_2) & 0 & x_2 & 0 & 0 & 1 & 0 \\ G_{x_1}^{[0]}(\mathbf{x}_3) & \dots & G_{x_9}^{[0]}(\mathbf{x}_3) & 0 & 0 & x_3 & 0 & 0 & 1 \\ G_{x_1}^{[0]}(\mathbf{x}_4) & \dots & G_{x_9}^{[0]}(\mathbf{x}_4) & x_1 & 0 & 0 & 1 & 0 & 0 \\ G_{x_1}^{[0]}(\mathbf{x}_5) & \dots & G_{x_9}^{[0]}(\mathbf{x}_5) & 0 & x_2 & 0 & 0 & 1 & 0 \\ G_{x_1}^{[0]}(\mathbf{x}_6) & \dots & G_{x_9}^{[0]}(\mathbf{x}_6) & 0 & 0 & x_3 & 0 & 0 & 1 \\ G_{x_1}^{[0]}(\mathbf{x}_7) & \dots & G_{x_9}^{[0]}(\mathbf{x}_7) & x_1 & 0 & 0 & 1 & 0 & 0 \\ G_{x_1}^{[0]}(\mathbf{x}_8) & \dots & G_{x_9}^{[0]}(\mathbf{x}_8) & 0 & x_2 & 0 & 0 & 1 & 0 \\ G_{x_1}^{[0]}(\mathbf{x}_9) & \dots & G_{x_9}^{[0]}(\mathbf{x}_9) & 0 & 0 & x_3 & 0 & 0 & 1 \end{bmatrix}$$

and  $y$ -dimension as

$$\frac{\partial^2 \tilde{u}}{\partial^2 y} = \zeta_y^{[2]} \tilde{\mathbf{w}}_y \quad (11)$$

$$\frac{\partial \tilde{u}}{\partial y} = \zeta_y^{[1]} \tilde{\mathbf{w}}_y \quad (12)$$

$$\tilde{u} = \zeta_y^{[0]} \tilde{\mathbf{w}}_y \quad (13)$$

where

$$\zeta_y^{[2]} = \begin{bmatrix} G_{y_1}^{[2]}(\mathbf{x}_1) & \dots & G_{y_9}^{[2]}(\mathbf{x}_1) & 0 & 0 & 0 & 0 & 0 & 0 \\ G_{y_1}^{[2]}(\mathbf{x}_2) & \dots & G_{y_9}^{[2]}(\mathbf{x}_2) & 0 & 0 & 0 & 0 & 0 & 0 \\ G_{y_1}^{[2]}(\mathbf{x}_3) & \dots & G_{y_9}^{[2]}(\mathbf{x}_3) & 0 & 0 & 0 & 0 & 0 & 0 \\ G_{y_1}^{[2]}(\mathbf{x}_4) & \dots & G_{y_9}^{[2]}(\mathbf{x}_4) & 0 & 0 & 0 & 0 & 0 & 0 \\ G_{y_1}^{[2]}(\mathbf{x}_5) & \dots & G_{y_9}^{[2]}(\mathbf{x}_5) & 0 & 0 & 0 & 0 & 0 & 0 \\ G_{y_1}^{[2]}(\mathbf{x}_6) & \dots & G_{y_9}^{[2]}(\mathbf{x}_6) & 0 & 0 & 0 & 0 & 0 & 0 \\ G_{y_1}^{[2]}(\mathbf{x}_7) & \dots & G_{y_9}^{[2]}(\mathbf{x}_7) & 0 & 0 & 0 & 0 & 0 & 0 \\ G_{y_1}^{[2]}(\mathbf{x}_8) & \dots & G_{y_9}^{[2]}(\mathbf{x}_8) & 0 & 0 & 0 & 0 & 0 & 0 \\ G_{y_1}^{[2]}(\mathbf{x}_9) & \dots & G_{y_9}^{[2]}(\mathbf{x}_9) & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\zeta_y^{[1]} = \begin{bmatrix} G_{y_1}^{[1]}(\mathbf{x}_1) & \dots & G_{y_9}^{[1]}(\mathbf{x}_1) & 1 & 0 & 0 & 0 & 0 & 0 \\ G_{y_1}^{[1]}(\mathbf{x}_2) & \dots & G_{y_9}^{[1]}(\mathbf{x}_2) & 1 & 0 & 0 & 0 & 0 & 0 \\ G_{y_1}^{[1]}(\mathbf{x}_3) & \dots & G_{y_9}^{[1]}(\mathbf{x}_3) & 1 & 0 & 0 & 0 & 0 & 0 \\ G_{y_1}^{[1]}(\mathbf{x}_4) & \dots & G_{y_9}^{[1]}(\mathbf{x}_4) & 0 & 1 & 0 & 0 & 0 & 0 \\ G_{y_1}^{[1]}(\mathbf{x}_5) & \dots & G_{y_9}^{[1]}(\mathbf{x}_5) & 0 & 1 & 0 & 0 & 0 & 0 \\ G_{y_1}^{[1]}(\mathbf{x}_6) & \dots & G_{y_9}^{[1]}(\mathbf{x}_5) & 0 & 1 & 0 & 0 & 0 & 0 \\ G_{y_1}^{[1]}(\mathbf{x}_7) & \dots & G_{y_9}^{[1]}(\mathbf{x}_7) & 0 & 0 & 1 & 0 & 0 & 0 \\ G_{y_1}^{[1]}(\mathbf{x}_8) & \dots & G_{y_9}^{[1]}(\mathbf{x}_8) & 0 & 0 & 1 & 0 & 0 & 0 \\ G_{y_1}^{[1]}(\mathbf{x}_9) & \dots & G_{y_9}^{[1]}(\mathbf{x}_9) & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

$$\zeta_y^{[0]} = \begin{bmatrix} G_{y_1}^{[0]}(\mathbf{x}_1) & \dots & G_{y_9}^{[0]}(\mathbf{x}_1) & y_1 & 0 & 0 & 1 & 0 & 0 \\ G_{y_1}^{[0]}(\mathbf{x}_2) & \dots & G_{y_9}^{[0]}(\mathbf{x}_2) & y_2 & 0 & 0 & 1 & 0 & 0 \\ G_{y_1}^{[0]}(\mathbf{x}_3) & \dots & G_{y_9}^{[0]}(\mathbf{x}_3) & y_3 & 0 & 0 & 1 & 0 & 0 \\ G_{y_1}^{[0]}(\mathbf{x}_4) & \dots & G_{y_9}^{[0]}(\mathbf{x}_4) & 0 & y_1 & 0 & 0 & 1 & 0 \\ G_{y_1}^{[0]}(\mathbf{x}_5) & \dots & G_{y_9}^{[0]}(\mathbf{x}_5) & 0 & y_2 & 0 & 0 & 1 & 0 \\ G_{y_1}^{[0]}(\mathbf{x}_6) & \dots & G_{y_9}^{[0]}(\mathbf{x}_6) & 0 & y_3 & 0 & 0 & 1 & 0 \\ G_{y_1}^{[0]}(\mathbf{x}_7) & \dots & G_{y_9}^{[0]}(\mathbf{x}_7) & 0 & 0 & y_1 & 0 & 0 & 1 \\ G_{y_1}^{[0]}(\mathbf{x}_8) & \dots & G_{y_9}^{[0]}(\mathbf{x}_8) & 0 & 0 & y_2 & 0 & 0 & 1 \\ G_{y_1}^{[0]}(\mathbf{x}_9) & \dots & G_{y_9}^{[0]}(\mathbf{x}_9) & 0 & 0 & y_3 & 0 & 0 & 1 \end{bmatrix}$$

So, the conversion matrix is determined as follows

$$\begin{pmatrix} \tilde{\mathbf{u}} \\ \tilde{\mathbf{0}} \\ \tilde{\mathbf{k}} \end{pmatrix} = \begin{bmatrix} \zeta_x^{[0]} & 0 \\ \zeta_x^{[0]} - \zeta_y^{[0]} \\ \kappa_x & \kappa_y \end{bmatrix} \begin{pmatrix} \tilde{\mathbf{w}}_x \\ \tilde{\mathbf{w}}_y \end{pmatrix} = C \begin{pmatrix} \tilde{\mathbf{w}}_x \\ \tilde{\mathbf{w}}_y \end{pmatrix}, \quad (14)$$

where the first sub-matrix  $\tilde{\mathbf{u}} = \zeta_x^{[0]} \tilde{\mathbf{w}}_x$  is used to collocate the function  $u$  over the stencil; the second sub-matrix  $\zeta_x^{[0]} \tilde{\mathbf{w}}_x - \zeta_y^{[0]} \tilde{\mathbf{w}}_y = \tilde{\mathbf{0}}$  is to force nodal values of  $u$  obtained from the integration with respect to  $x$  and  $y$  to be identical; the third sub-matrix  $\kappa_x \tilde{\mathbf{w}}_x + \kappa_y \tilde{\mathbf{w}}_y = \tilde{\mathbf{k}}$  is to express values of the PDE (1) at selected collocation points, which are  $(\mathbf{x}_2, \mathbf{x}_4, \mathbf{x}_8, \mathbf{x}_6)$  in this study (see Figure 1).

The network-weight space (space of weights) into the physical space is achieved by inverting Equation (14) as  $\begin{pmatrix} \tilde{\mathbf{w}}_x \\ \tilde{\mathbf{w}}_y \end{pmatrix} = C^{-1} \begin{pmatrix} \tilde{\mathbf{u}} \\ \tilde{\mathbf{0}} \\ \tilde{\mathbf{k}} \end{pmatrix}$ . In other words, we have.

$$\tilde{\mathbf{w}}_x = C_x \begin{pmatrix} \tilde{\mathbf{u}} \\ \tilde{\mathbf{0}} \\ \tilde{\mathbf{k}} \end{pmatrix} \quad (15)$$

and

$$\tilde{\mathbf{w}}_y = C_y \begin{pmatrix} \tilde{\mathbf{u}} \\ \tilde{\mathbf{0}} \\ \tilde{\mathbf{k}} \end{pmatrix} \quad (16)$$

where  $(C_x, C_y)^T = C^{-1}$ . By substituting Equations (15) and (16) into Equations (8–10) and (11–13), the derivatives of function  $u$  with respect to  $x$  and  $y$  over a local stencil are now expressed in physical space.

### 2.2.1. BCs Applied for CLIRBF Schemes

It can be found that, for the Dirichlet BC, the governing equation yields at each grid point an algebraic equation whose variables are the function values of points in the associated stencil. Applying this procedure for all interior points (of the considered domain) and their associated stencils yields a system of algebraic equations. Since the variables of these equations at boundary points are known as Dirichlet BC, they are moved to the right-hand side, yielding a new system of equations which is now solved using an iterative method.

In the present CLIRBF approach, there are two ways to impose the Neumann BC on the final system of algebraic equations. The first one is to add expressions of Neumann BC to the final equation system. This approach is effective for coarse grids but may cause numerical instability for dense grids due to the high condition number of corresponding system matrices.

The second way is to introduce expressions of Neumann BCs into conversion matrices of associated stencils. Governing equations at the boundary points are then also derived and put into the final algebraic system of equations. For example, take a 9-point stencil as shown in Figure 1 with Neumann BC  $\frac{\partial u}{\partial x} = e(\mathbf{x})$  applied at the boundary nodal points  $\{\mathbf{x}_7, \mathbf{x}_8, \mathbf{x}_9\}$ . The conversion matrix is constructed as.

$$\begin{pmatrix} \tilde{\mathbf{u}} \\ \tilde{\mathbf{0}} \\ \tilde{\mathbf{k}} \\ \tilde{\mathbf{e}} \end{pmatrix} = \underbrace{\begin{bmatrix} \zeta_x^{[0]} & 0 \\ \zeta_x^{[0]} - \zeta_y^{[0]} \\ \kappa_x & \kappa_y \\ \zeta_x^{[1]} & 0 \end{bmatrix}}_C \begin{pmatrix} \tilde{\mathbf{w}}_x \\ \tilde{\mathbf{w}}_y \end{pmatrix} = C \begin{pmatrix} \tilde{\mathbf{w}}_x \\ \tilde{\mathbf{w}}_y \end{pmatrix}, \quad (17)$$

where  $e(\mathbf{x})$  is a given function, and  $\tilde{\mathbf{e}} = [e(\mathbf{x}_7), e(\mathbf{x}_8), e(\mathbf{x}_9)]^T$ .

### 2.2.2. Non-Rectangular Domains

In non-rectangular domains, some points are positioned near the non-rectangular boundary. The stencils associated with these points can be arbitrary as shown in Figure 2. The IRBF approximation for these stencils is carried out in a dual stencil associated with  $x$ -gridlines and  $y$ -gridlines, namely the  $x$ - and  $y$ -stencils.

For example, take the non-rectangular stencil as shown in Figure 2, the  $x$ -stencil is the set  $X$  of grid points and boundary points generated by the intersection of  $x$ -gridlines and the boundary:  $X = \{x_1, x_2, x_3, x_4, x_6, x_7, x_8, x_{10}\}$ . Similarly, the  $y$ -stencil is the set  $Y$  of grid points and boundary points generated by the intersection of  $y$ -gridlines and the boundary:  $Y = \{x_1, x_2, x_3, x_5, x_6, x_7, x_9, x_{10}\}$ .

The  $x$ -stencil is used to approximate unknowns and their derivatives with respect to  $x$ , i.e.  $\frac{\partial^2 u}{\partial x^2}$ ,  $\frac{\partial u}{\partial x}$  and  $u_x$ , while the  $y$ -stencil is to approximate those with respect to  $y$ , i.e.  $\frac{\partial^2 u}{\partial y^2}$ ,  $\frac{\partial u}{\partial y}$  and  $u_y$ , where  $u_x$  and  $u_x$  are determined by Equation (4) with  $x_j \equiv x$  and  $u_y$  with  $x_j \equiv y$ .

In order to construct the conversion matrix, several sets of points are explicitly defined as follows.

$$\begin{cases} S_1 = X = \{x_1, x_2, x_3, x_4, x_6, x_7, x_8, x_{10}\} \\ S_2 = Y \setminus X = \{x_5, x_9\} \\ S_3 = X \cap Y = \{x_1, x_2, x_3, x_6, x_7, x_{10}\} \\ S_4 = \{x_2, x_7\} \\ S_5 = \{x_2\} \end{cases}$$

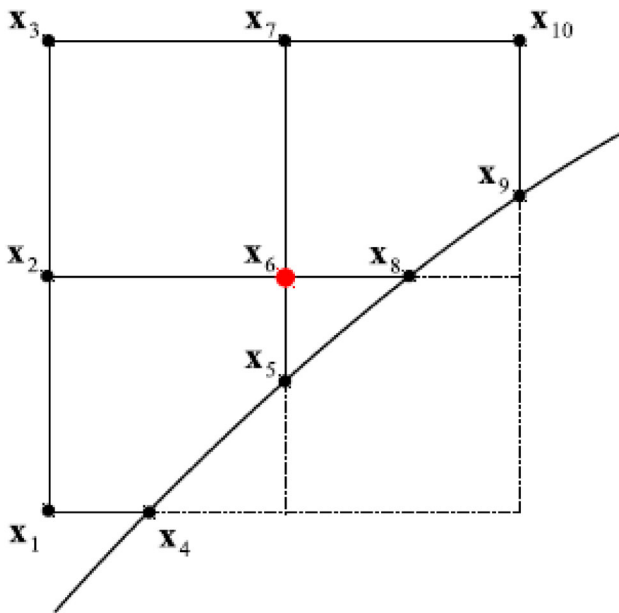


Figure 2. Non-rectangular stencil.

where  $S_1$  is the set of all points in  $x$ -stencil;  $S_2$  is the set of boundary points generated by  $y$ -gridlines crossing the non-rectangular boundary;  $S_3$  is the set of common points between  $x$ -stencil and  $y$ -stencil and  $S_4$  is the set of compact points, which are grid points lying on the cross whose centre is the reference point  $x_5$ .

Suppose that a Neumann BC  $\frac{\partial u}{\partial x} = e(x)$  is imposed at  $x_2$  then  $S_5 = \{x_2\}$  is defined as the set of points with Neumann BC in  $x$ -dimension. The conversion matrix associated with this stencil is determined as follows.

$$\begin{pmatrix} \tilde{u}_{S_1} \\ \tilde{u}_{S_2} \\ \tilde{0}_{S_3} \\ \tilde{k}_{S_4} \\ \tilde{e}_{S_5} \end{pmatrix} = \underbrace{\begin{bmatrix} \zeta_x^{[0]} & 0 \\ 0 & \zeta_y^{[0]} \\ \zeta_x^{[0]} & -\zeta_y^{[0]} \\ \kappa_x & \kappa_y \\ \zeta_x^{[1]} & 0 \end{bmatrix}}_C \begin{pmatrix} \tilde{w}_x \\ \tilde{w}_y \end{pmatrix} = C \begin{pmatrix} \tilde{w}_x \\ \tilde{w}_y \end{pmatrix}, \quad (18)$$

where  $\zeta_x^{[0]} \tilde{w}_x = \tilde{u}_{S_1}$  captures the collocation of  $u_x$  at  $S_1$ ,  $\zeta_y^{[0]} \tilde{w}_y = \tilde{u}_{S_2}$  is the collocation of  $u_y$  at  $S_2$ ,  $\zeta_x^{[0]} \tilde{w}_x - \zeta_y^{[0]} \tilde{w}_y = \tilde{0}_{S_3}$  is the enforcement  $u_x = u_y$  at  $S_3$ ,  $\kappa_x \tilde{w}_x - \kappa_y \tilde{w}_y = \tilde{k}_{S_4}$  is the compact information, which represents the governing equation at  $S_4$ , and  $\zeta_x^{[1]} \tilde{w}_x = \tilde{e}_{S_5}$  is the Neumann BC at  $S_5$ .

## 3. Compact Local IRBF Coupled Non-Overlapping DD Parallel Method

A new approach based on the marriage of Steklov–Poincaré non-overlapping DD scheme and the compact local IRBF method is developed into a parallel algorithm. Figure 3 describes an example for the present method with two sub-domains whose boundary is  $\partial\Omega = \Gamma_1 \cup \Gamma_2$ , and one artificial boundary ( $\Gamma$ ).

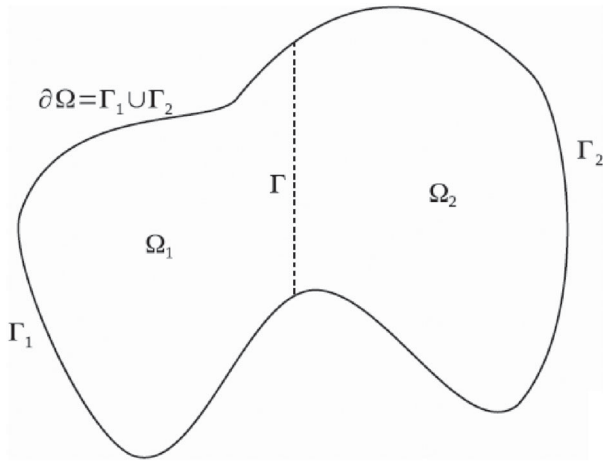
### 3.1. Review on Non-Overlapping Dirichlet-Neumann DD method

The 2D problem Equation (1) is represented with the introduction of artificial boundary  $\Gamma$  as.

$$\begin{cases} Lu = f, \mathbf{x} \in \Omega \\ Bu = g, \mathbf{x} \in \Gamma_1 \cup \Gamma_2 \end{cases} \quad (19)$$

The problem Equation (19) is expressed for a considered domain of two sub-domains  $\Omega_1$  with  $\partial\Omega_1 = \Gamma_1$





**Figure 3.** Non-overlapping DD method in 2D.

$\cup \Gamma$  and  $\Omega_2$  with  $\partial\Omega_2 = \Gamma_2 \cup \Gamma$  as follows.

$$\begin{cases} Lu_1 = f, & \mathbf{x} \in \Omega_1 \\ Bu_1 = g, & \mathbf{x} \in \Gamma_1 \\ u_1 = u_2 & \mathbf{x} \in \Gamma \\ Lu_2 = f & \mathbf{x} \in \Omega_2 \\ Bu_2 = g & \mathbf{x} \in \Gamma_2 \\ \frac{\partial u_2}{\partial n} = \frac{\partial u_1}{\partial n} & \mathbf{x} \in \Gamma \end{cases} \quad (20)$$

The two constraints  $u_1 = u_2$  and  $\frac{\partial u_2}{\partial n} = \frac{\partial u_1}{\partial n}$  are BCs for  $u$  across the artificial boundary  $\Gamma$ . It is proved that the solution of the equivalent problem Equation (20) converges to the solution of the original problem Equation (19) as stated by Quarteroni and Valli (1999). Thus, the present non-overlapping Dirichlet–Neumann DD method can be written for a step  $k$  for each sub-domain as

$$\begin{cases} Lu_1^k = f, & \mathbf{x} \in \Omega_1 \\ Bu_1^k = g, & \mathbf{x} \in \Gamma_1 \\ u_1^k = u_2^{k-1} & \mathbf{x} \in \Gamma \end{cases} \quad (21)$$

and

$$\begin{cases} Lu_2^k = f, & \mathbf{x} \in \Omega_2 \\ Bu_2^k = g, & \mathbf{x} \in \Gamma_2 \\ \frac{\partial u_2^k}{\partial n} = \frac{\partial u_1^k}{\partial n} & \mathbf{x} \in \Gamma \end{cases} \quad (22)$$

In order to improve the convergence of non-overlapping Dirichlet–Neumann DD method, a relaxation factor  $\theta$  is introduced into the transmission condition in Equation (21) as follows,

$$u_1^k = \theta u_1^{k-1} + (1 - \theta)u_2^{k-1}, \mathbf{x} \in \Gamma$$

where  $\theta \in [0, 1]$ .

### 3.2. Review on Parallel Algorithm of Non-Overlapping Dirichlet-Neumann DD Method

It can be found that the DD method presented in Section 3.1 is not suitable for parallel computing because the calculation in sub-domain 2 by Equations (22) requires values on the artificial boundary from sub-domain 1 by Equations (21) in the same iteration step. Fortunately, this issue (data dependence) can be handled by introducing the parallel computation. There are two common ways of parallelisation of non-overlapping DD method.

The first way is to use the black and white colouring technique. By this technique, sub-domains that have no-common artificial boundary, i.e. not adjacent, are marked with black colour. The remaining sub-domains, which are also not adjacent, are marked with white colour. In the first iteration, only black sub-domains are computed in parallel, and in the next iteration, only white sub-domains are computed in parallel and so on. Although the approach maintains the form of the equivalent problem Equation (19), it only allows at most  $N/2$  sub-domains to run in parallel, where  $N$  is the number of sub-domains.

For the second way, a small modification to Equation (22) is carried out as below to allow all sub-domain to run concurrently.

$$\begin{cases} Lu_2^k = f, & \mathbf{x} \in \Omega_2 \\ Bu_2^k = g, & \mathbf{x} \in \Gamma_2 \\ \frac{\partial u_2^k}{\partial n} = \frac{\partial u_1^{k-1}}{\partial n} & \mathbf{x} \in \Gamma \end{cases} \quad (23)$$

where the values of the first-order derivative on the artificial boundary of sub-domain 2 are obtained from sub-domain 1 in the previous time step. The second way is used in the present method.

### 3.3. The Present Compact Local IRBF Based Non-Overlapping DD Parallel Method

The considered domain of problem is divided into non-overlapping sub-domains. In each sub-domain, a sub-problem with either Dirichlet or Dirichlet and Neumann BC will be solved using the compact local IRBF approximation scheme as presented in Section 2. The solution is considered converged once the convergence measures (CM) in each sub-domain and on all artificial boundaries achieve predefined tolerances.

### 3.3.1. Communication and synchronisation

The present parallel algorithm is implemented in MATLAB environment. The parallel communication is achieved by using standard send and receives functions (MathWork 2012). Since these operations are synchronous, the synchronisation and delivery of messages are guaranteed.

### 3.3.2. Termination

As presented before, the algorithm stops when CMs reach given tolerances. CM is the norm-2 of a field variable and defined as follows.

$$CM = \frac{\sqrt{\sum_{i=1}^n (\mathbf{u}_i^{k+1} - \mathbf{u}_i^k)^2}}{\sqrt{\sum_{i=1}^n (\mathbf{u}_i^k)^2}} \quad (24)$$

where  $k$  is the iteration step,  $n$  is the number collocation points and  $u$  is the field variable.

In this method, three CMs are used to verify the convergence of the numerical solution.  $CM^{[u]}$  is the convergence measure of the field variable obtained by two successive time steps within a sub-domain.  $ABCM^{[u]}$  is the convergence measure of the velocity on the artificial boundary between two adjacent sub-domains.  $ABCM^{[\frac{\partial u}{\partial n}]}$  is the convergence measure of the first derivative of velocity on the artificial boundary between two adjacent sub-domains. The solution is considered converged if all three following conditions are met

$$\begin{cases} CM^{[u]} < CM_{tol}, & \forall \mathbf{x} \in \Omega \\ ABCM^{[u]} < ABCM_{tol}^{[u]}, & \forall \mathbf{x} \in \Gamma \\ ABCM^{[\frac{\partial u}{\partial n}]} < ABCM_{tol}^{[\frac{\partial u}{\partial n}]}, & \forall \mathbf{x} \in \Gamma \end{cases} \quad (25)$$

where  $CM_{tol}$ ,  $ABCM_{tol}^{[u]}$  and  $ABCM_{tol}^{[\frac{\partial u}{\partial n}]}$  are the given tolerances for velocity within sub-domains, on artificial boundaries, and the first derivative of velocity on artificial boundaries, respectively.

In parallel computation, each sub-domain is a part of a distributed system. A sub-domain can terminate if there is no other adjacent sub-domain communicating with it. In this work, the Bitmap distributed termination detection (DTD) algorithm (Pham-Sy et al. 2013) is used to achieve an efficient termination.

### 3.3.3. Review on Bitmap DTD Algorithm

The method uses a bitmap (group of binary bits) to store the state of all processes in the system. Each bit has two values 0 and 1, which are correspondingly equivalent to two states (active or passive) of a process. The length of a bitmap is equal to the number of processes in the system. During the computation, the bitmap is spread throughout the system and itself expresses current state of the system. One process, depending on the value of the bitmap, can judge the global state of the whole system. The bitmap with all bits of 1 means all processes are passive.

The present DTD algorithm consists of two parts: (i) termination detection and (ii) synchronous termination. The Termination detection part aims to detect the system's quiescence. Once the quiescence state is detected, the Synchronous Termination is activated to get all processes terminated simultaneously at the same step. It is noted that if one process terminates while some processes are still active, those active processes cannot exchange information with their terminated neighbours and thus cannot finish their work. Consequently, the system will end up with deadlock. So, an important goal of our DTD algorithm is to allow all processes to terminate simultaneously.

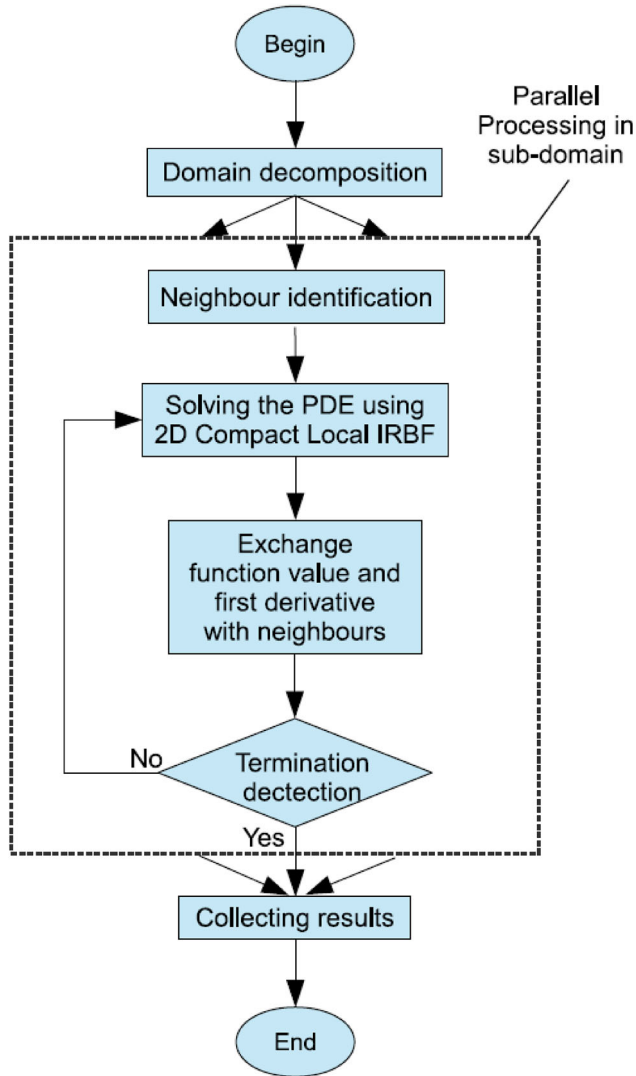
The termination detection algorithm in each process is started by setting a zero bitmap. Take an arbitrary process as an example, this process keeps its bitmap as a record of current state of the system. After each iteration of the underlying computation, if the process becomes passive, the bit corresponding to its enumeration number will be set to 1, and then the bitmap will be exchanged with its neighbours. When this process receives a bitmap from one of its neighbours, it will update its own bitmap by doing a binary union between its bitmap and received bitmap. Finally, the process checks the value of its bitmap. If the bitmap with all bits of 1, the process detects termination. Otherwise, the algorithm is repeated. The bitmap DTD algorithm allows any sub-domain to detect the termination. More details on Bitmap DTD algorithm can be found in Pham-Sy et al. (2013)

### 3.3.4. The Present Numerical Algorithm

The parallel algorithm of the present method as described in Flowchart (see Figure 4) consists of two parts: a sequential part and a parallel part.

The sequential part includes two blocks of processing, the DD at the beginning and the collecting results





**Figure 4.** Flowchart of non-overlapping Dirichlet-Neumann DD method based parallel algorithm.

at the end. The block of DD is to create sub-domains and their relative coordinates in the whole domain. The collecting results block is to gather data from all sub-domains and then merges them into one global solution.

In the parallel part, each sub-domain firstly needs to identify its neighbours. The Navier–Stokes equations are then solved together with BCs in each sub-domain using 2D Compact Local IRBF scheme (see Section 2.2). The obtained solution is used to approximate the first-order normal derivative of the field variable on artificial boundaries. Both the field variable and its normal derivative on artificial boundaries are sent to neighbouring sub-domains. These data are used to determine CM and to check the convergence condition as mentioned in expressions (25). The algorithm will

be completed by the termination detection, which is to find whether the calculation in a current sub-domain is finished and to terminate accordingly.

## 4. Numerical Examples

The efficiency of the present method is demonstrated by solving two benchmark problems: the LDC fluid flow and the NC in concentric annuli. The obtained results are compared with those of the non-parallel method as well as with benchmark results to investigate and evaluate the accuracy of the present method of a parallel computation.

The numerical computations were performed on high-performance computing (HPC) cluster, which comprises: 48 compute nodes, each with  $2 \times$  quad-core 2.7 GHz AMD Opteron CPUs and 16 GB of memory and 1 visualisation node consisting of Sun X4440 system with  $4 \times$  six-core 2.4 GHz Opteron CPUs, 64 GB of memory, and NVidia graphics card. Also, there is 80 TB of shared storage.

### 4.1. LDC Fluid Flow

The LDC fluid flow is described in dimensionless form, where the square cavity has three fixed walls and a sliding top lid. The cavity is filled with a viscous fluid which is driven by the lid with a velocity of 1 from left to right (see Figure 5).

The governing equations of the problem expressed in stream-function and vorticity formulation are given as.

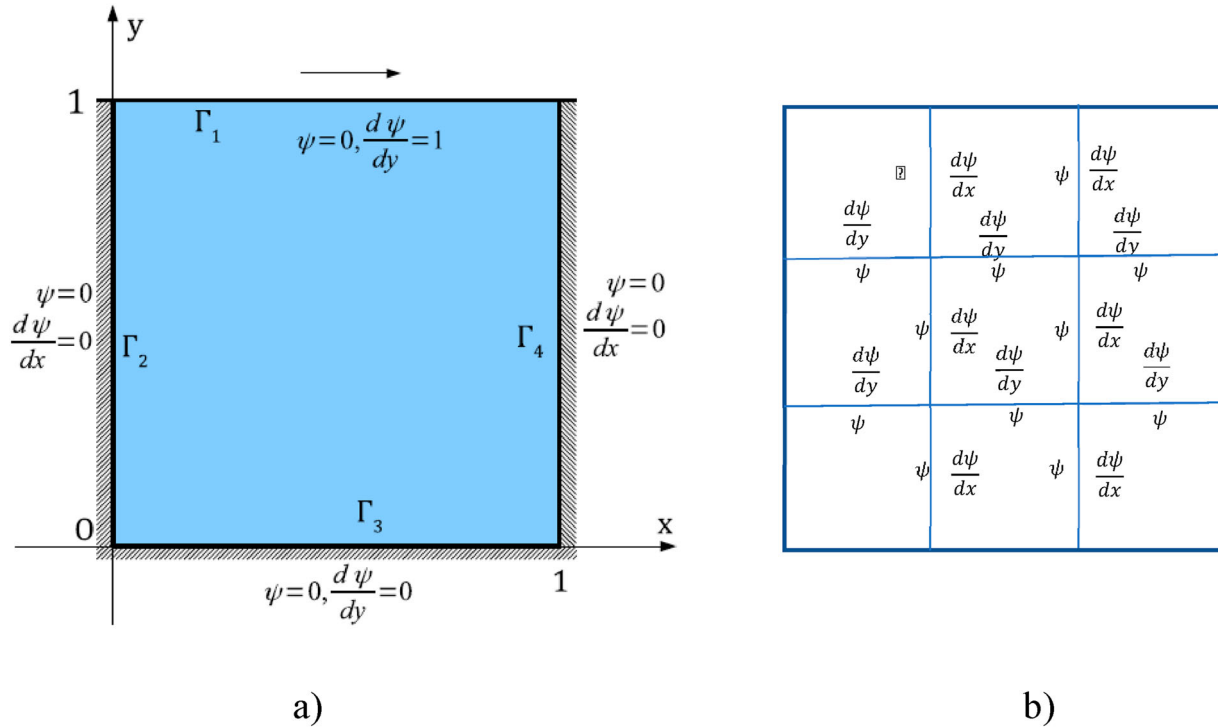
$$\frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} = -\omega, \quad (26)$$

$$\frac{1}{Re} \left( \frac{\partial^2 \omega}{\partial x^2} + \frac{\partial^2 \omega}{\partial y^2} \right) = \frac{\partial \omega}{\partial t} + \left( \frac{\partial \psi}{\partial y} \frac{\partial \omega}{\partial x} - \frac{\partial \psi}{\partial x} \frac{\partial \omega}{\partial y} \right), \quad (27)$$

where  $\psi$  is the stream-function variable and  $\omega$  the vorticity variable. The Reynolds ( $Re$ ) number is defined as  $Re = \frac{UL}{\nu}$ , where  $U$  is the speed of the lid;  $L$  is the side of the cavity and  $\nu$  is the kinematic viscosity of the fluid.

The relationship of  $\psi$  and  $\omega$  with velocity components  $u$  and  $v$  is given by

$$u = \frac{\partial \psi}{\partial y}; v = -\frac{\partial \psi}{\partial x}, \omega = \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y}.$$



**Figure 5.** The LDC fluid flow problem. (a) Geometry and BCs and (b) for parallel computing, the domain is divided into several equal rectangular sub-domains: an example of nine sub-domains with BCs on artificial boundaries.

The BCs of the LDC problem are determined in Figure 5 as follows.

$$\psi = 0, \quad \frac{\partial \psi}{\partial y} = 1 \quad \forall (x, y) \in \Gamma_1; \quad (28)$$

$$\psi = 0, \quad \frac{\partial \psi}{\partial x} = 0 \quad \forall (x, y) \in \Gamma_2 \cup \Gamma_3 \cup \Gamma_4. \quad (29)$$

For parallel computation, the considered domain (Figure 5) is divided into a range of equal rectangular sub-domains  $\{2, 4, 9, \dots, 81\}$ .

As the Dirichlet–Neumann non-overlapping DD method is applied, there is an alternation from the Dirichlet BCs on one side of artificial boundaries to the Neumann BCs on another side. For the load balance, while the Dirichlet BCs are set on the top and right artificial boundaries, the Neumann BCs are installed on the bottom and left artificial boundaries for each sub-domain. An example of nine sub-domains with BCs on artificial boundaries is described in Figure 5(b).

In each sub-domain, the problem is solved through the following steps.

- (a) Guess the initial BCs on the artificial boundaries.

- (b) Solve Equation (26) using the compact local 2D-IRBF method as presented in Section 2 for a rectangle domain to obtain the value of the stream-function variable  $\psi$  at collocation points within the sub-domain.
- (c) Approximate the vorticity variable  $\omega$  at collocation points on the artificial boundaries, and then solve Equation (27) using the compact local 2D-IRBF method to obtain the value of the vorticity variable  $\omega$  at collocation points within the sub-domain.
- (d) Exchange the Dirichlet and/or Neumann BCs over the artificial boundaries between sub-domains.
- (e) Check the termination conditions using expressions (25). If the termination conditions are met then terminate. Otherwise, update BCs on the artificial boundaries and go back to step (b).

Since the BCs of  $\omega$  are unknown, they need to be determined to solve Equation (27). In this implementation, the BCs of  $\omega$  are approximated from Equation (26) using the global 1D-IRBF method (Mai-Duy and Tran-Cong 2001).

**Table 1.** Parallel computation of the LDC problem with  $Re = 100$  and 400.

$n_x \times n_y$	CPUs	$\Delta t$	$CM_{Tot}$	$CM_p$	$\Delta t$	$CM_{Tot}$	$CM_p$	$\Delta t$	$CM_{Tot}$	$CM_p$
		Re = 100			Re = 400			Re = 1000		
101 × 101	4	1.E-3	1.E-6	3.2612E-7	1.E-3	1.E-6	3.4120E-7	1.E-4	1.E-6	2.8976E-7
	9	1.E-3	1.E-6	2.6812E-7	1.E-3	1.E-6	2.6732E-7	1.E-4	1.E-6	3.5392E-7
	16	1.E-3	1.E-6	2.5326E-7	1.E-3	1.E-6	2.5307E-7	1.E-4	1.E-6	2.8731E-7
	25	1.E-3	1.E-6	2.2448E-7	1.E-3	1.E-6	2.0506E-7	1.E-4	1.E-6	2.4154E-7
	36	1.E-3	1.E-6	2.0544E-7	1.E-3	1.E-6	1.3151E-7	1.E-4	1.E-6	1.8338E-7
151 × 151	4	1.E-3	1.E-6	3.4122E-7	1.E-3	1.E-6	3.7022E-7	1.E-5	1.E-6	2.9676E-7
	9	1.E-3	1.E-6	2.7182E-7	1.E-3	1.E-6	3.3321E-7	1.E-5	1.E-6	3.6542E-7
	16	1.E-3	1.E-6	2.6828E-7	1.E-3	1.E-6	3.3147E-7	1.E-5	1.E-6	2.8731E-7
	25	1.E-3	1.E-6	2.4384E-7	1.E-3	1.E-6	2.8603E-7	1.E-5	1.E-6	2.3874E-7
	36	1.E-3	1.E-6	2.2544E-7	1.E-3	1.E-6	2.8931E-7	1.E-5	1.E-6	1.7498E-7
	49	1.E-3	1.E-6	2.0872E-7	1.E-3	1.E-6	2.4277E-7	1.E-5	1.E-6	1.7053E-7
	64	1.E-3	1.E-6	1.5316E-7	1.E-3	1.E-6	2.5118E-7	1.E-5	1.E-6	1.5181E-7
209 × 209	9	1.E-4	1.E-6	2.9116E-7	1.E-4	1.E-6	3.2546E-7	1.E-6	1.E-6	1.8643E-7
	16	1.E-4	1.E-6	3.2836E-7	1.E-4	1.E-6	3.8783E-7	1.E-6	1.E-6	1.9179E-7
	25	1.E-4	1.E-6	3.3027E-7	1.E-4	1.E-6	3.1972E-7	1.E-6	1.E-6	1.7723E-7
	36	1.E-4	1.E-6	2.3864E-7	1.E-4	1.E-6	2.2997E-7	1.E-6	1.E-6	1.5456E-7
	64	1.E-4	1.E-6	2.0216E-6	1.E-4	1.E-6	2.7347E-7	1.E-6	1.E-6	1.4622E-7

$n_x \times n_y$ , grid of collocation points; CPUs, number of CPUs (number of sub-domains);  $\Delta t$ , time step;  $CM_{tol}^{[u]} = 10^{-6} ABCM_{tol}^{[u]}$  ( $ABCM_{tol}^{[u]}$  or  $ABCM_{tol}^{[\frac{\partial u}{\partial n}]}$ ): tolerance of convergence measure at interfaces;  $CM_p$ , average convergence measure of the whole analysis domain.

To demonstrate the convergence achieved in the present parallel method, fluid of  $Re$  numbers (100, 400 and 1000) is considered in the simulation using several uniform grids  $n_x \times n_y$  (101 × 101 points; 151 × 151 points, and 209 × 209 points),  $\theta = 0.45$  and  $\beta = 2$ . The time step ( $\Delta t$ ) is chosen in the range from  $10^{-3}$  to  $10^{-5}$  based on  $Re$  value and grid size (smaller time steps are for finer grids or/and higher  $Re$ ). Results by the present parallel method using a range of sub-domains (4, 9, 16, 25, 36 and 64) in Table 1 show the convergences of the three grids of collocation points have reached with the order of  $10^{-6}$  for the average convergence measure of the whole analysis domain ( $CM_p$ ).

Also, Figure 6 depicts profiles of the velocities  $u$  and  $v$  along the vertical and horizontal centrelines, respectively at Reynolds numbers 100, 400 and 1000 with grids of 101 × 101 points (6(a,b)), 209 × 209 points (Figure 6(c,d)) and using the present parallel method with 36 sub-domains. These results are in very good agreement with the benchmark solution by Ghia, Ghia, and Shin (1982) using a grid of 129 × 129 points. Meanwhile, grid of 151 × 151 points is selected to thoroughly discuss on the accuracy and efficiency of the present parallel method.

The LDC fluid flow problem by the present parallel method is investigated with several Reynolds numbers {100, 400, 600, 1000} using a grid of 151 × 151 points,  $\theta = 0.45$  and  $\beta = 2$ ; and converges with given tolerances  $CM_{tol}^{[u]} = 10^{-6}$ ,  $ABCM_{tol}^{[u]} = 10^{-6}$ ,  $ABCM_{tol}^{[\frac{\partial u}{\partial n}]} =$

$10^{-6}$ , for velocity within sub-domains, velocity on artificial boundaries and the first-order derivatives on artificial boundaries, respectively. The contours of stream-function  $\psi$  and vorticity  $\omega$  are presented in Figures 7 and 8 while the velocity profiles are shown in Figure 9.

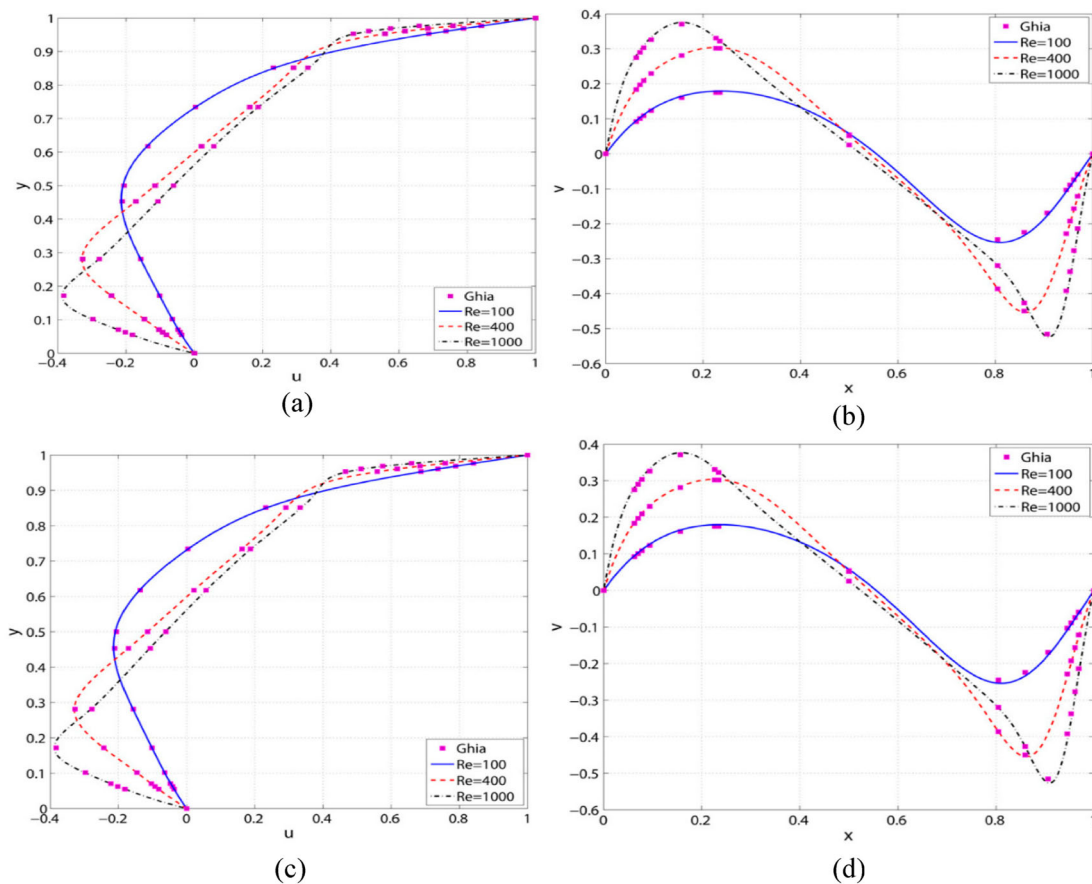
- On the accuracy of the present parallel method, the contours of stream-function and vorticity in Figures 7 and 8 using the present method with 16 sub-domains using 16 CPUs, show a great agreement with benchmark results by Ghia, Ghia, and Shin (1982) and Botella and Peyret (1998). While the velocity profiles  $u$  along the vertical centreline and  $v$  along the horizontal centreline in Figure 9 also go well through the corresponding values given by Ghia, Ghia, and Shin (1982).
- The efficiency of the parallel method is often evaluated by the speed-up  $S$  and the efficiency  $E$ , which are determined as follows,

$$S = \frac{T_s}{T_p}; \quad E = \frac{S}{p} \quad (30)$$

where  $T_s$  is the computation time on a single CPU,  $T_p$  is the computation time on parallel CPUs,  $p$  is the number of parallel CPUs.

The efficiency of the present parallel method is demonstrated using a grid of 151 × 151 with a range of numbers of CPU {1, 2, 4, 9, 16, 25, 36, 49, 64, 81}.

It can be seen in Figure 10(c), the throughput of the present parallel method is very high. Furthermore, the



**Figure 6.** The LDC fluid flow. Profiles of the  $u$  velocity along the vertical centre lines (a,c) and the  $v$  velocity along the horizontal centre lines (b,d) by the present parallel method using 36 CPUs at several  $Re$  numbers ( $Re = 100, 400$ , and  $1000$ ) and grids ( $101 \times 101$  and  $295 \times 295$ ) in comparison with the corresponding Ghia's results. (a) Grid  $101 \times 101$ , (b) Grid  $101 \times 101$ , (c) Grid  $295 \times 295$ , (d) Grid  $295 \times 295$ .

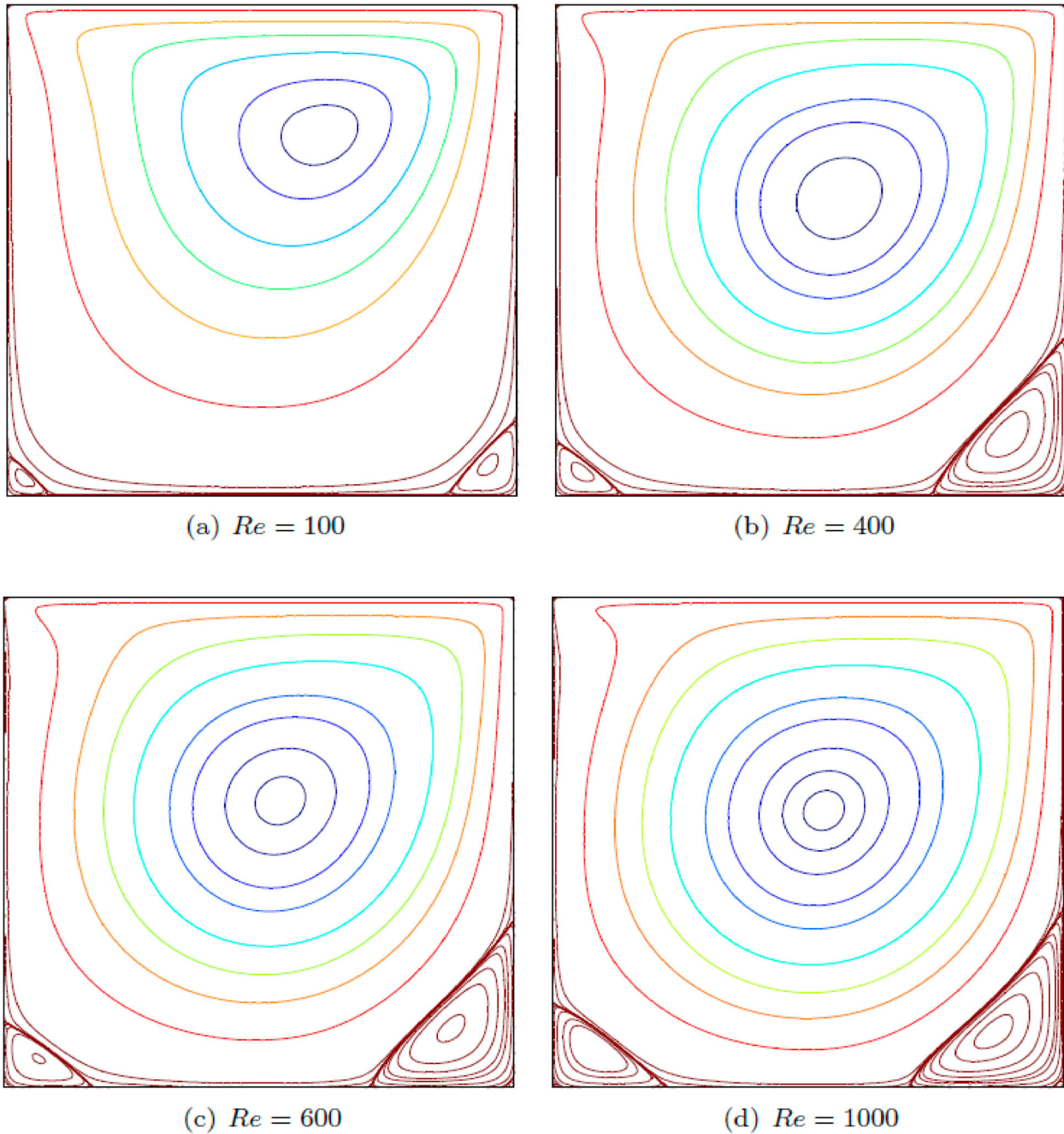
speed-up grows steadily with the increase of number of CPUs (see Figure 10(b)), this shows a good scalability of the present parallel method.

For a parallel method, the threshold defines an upper-bound, over which the efficiency of the parallel computation commences to reduce. It is worth noting that the speed-up of the present parallel method for the LDC fluid flow problem deteriorates as the number of CPUs exceeds a certain threshold. For example, the threshold is 64 CPUs for  $Re = 400$  and  $600$ ; and 49 CPUs for  $Re = 1000$  (see Table 3). Also, in this example, it appears that the CPU threshold decreases with increasing  $Re$  number.

The numerical results given in Figure 10 and Tables 2 and 3 also depict that the present method achieves a very high efficiency, for example the efficiencies are 104.43; 112.61; 106.81 and 102.78 using 25, 36, 49, 64 CPUs, respectively, for  $Re = 600$ .

Furthermore, a super-linear speed-up is observed with 25 CPUs for  $Re = 100$ ; with {16, 25, 36, 49, 64} CPUs for  $Re = 400$  (see Table 1); and with {25, 36, 49, 64} CPUs for  $Re = 600$ ; and with {16, 25, 36, 49, 64} CPUs for  $Re = 1000$  (see Table 3). A reason for this high efficiency is the insignificant increase of number of iterations in the present parallel computation in comparison with the sequential one (using 1 CPU) while for an iteration the computation time to solve Equation (26) in each sub-domain using the compact local 2D-IRBF method reduces massively as shown in Table 3 and the communication time takes a small amount of total computation time using bitmap DTD algorithm as discussed below and in Table 4.

To investigate the ratio of the communication time to the total execution time on each CPU, the time profile of the case using 25 CPUs is recorded and provided



**Figure 7.** The LDC fluid flow problem. Stream-function ( $\psi$ ) contours of the flow for four Reynolds numbers ( $Re = 100, 400, 600$  and  $1000$ ) by the present parallel method using 16 CPUs with the specifications: grid  $151 \times 151$ ,  $\Delta t = 10^{-3}$ ,  $CM_{tol}^{[u]} = 10^{-6}$ ,  $ABCM_{tol}^{[u]} = 10^{-6}$ ,  $ABCM_{tol}^{[\frac{\partial u}{\partial n}]} = 10^{-6}$ ,  $\theta = 0.45$  and  $\beta = 2$ .

in Table 4. The obtained results show that the percentage of the communication time in the total time ranges from 12.38% to 24.29%.

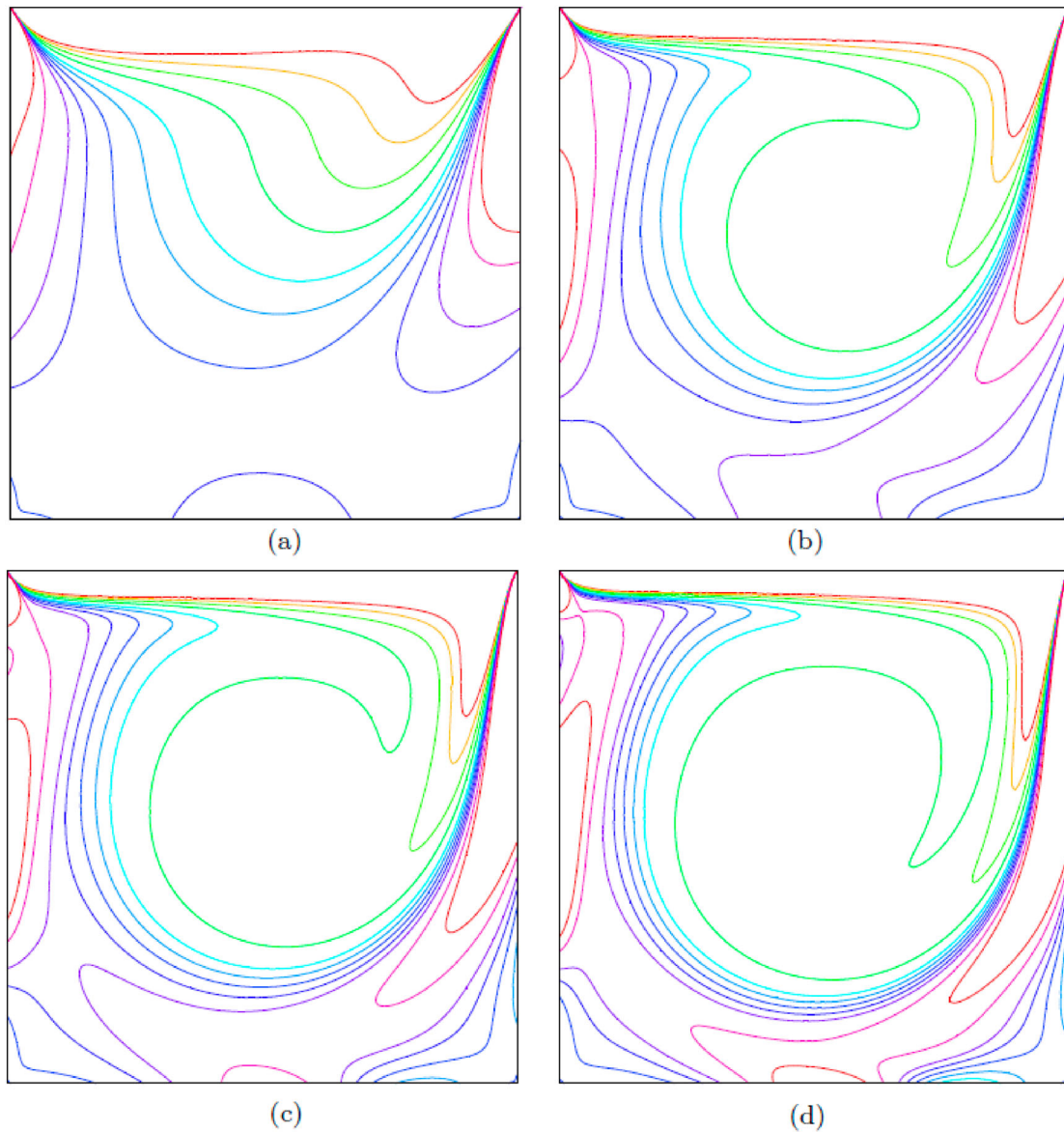
For an objective comparison on the efficiency of the present parallel computation over the sequential computation, a normalised computation time is defined as the time required for each CPU to complete computation task in an ideal parallel program and given by

$S = \frac{T_s}{T_p}$ . Hence, the efficiency  $E$  expressed in Equation (30) can now be determined by.

$$E = \frac{S}{p} \quad (31)$$

A bar graph of the efficiency of the parallel and the sequential algorithms using the present method is provided in Figure 11. The graph presents a quantitative comparison between the communication time and





**Figure 8.** The LDC fluid flow problem. Vorticity ( $\omega$ ) contours of the flow for several *Reynolds* numbers ( $Re = 100, 400, 600$  and  $1000$ ) by the present parallel method using 16 CPUs. Other parameters are given in Figure 7.

computation time by the parallel computation as well as the comparison between the present parallel computation using 25 CPUs and the associated sequential one, in which super-linear efficiency is observed.

#### 4.2. NC in Concentric Annuli

The problem of NC in concentric annuli which plays an important role in various engineering problems such as heat distribution, reactor design and engine design. For this problem, a fluid flow is enclosed in an annulus including a concentric inner circular cylinder which is surrounded by an outer square cylinder

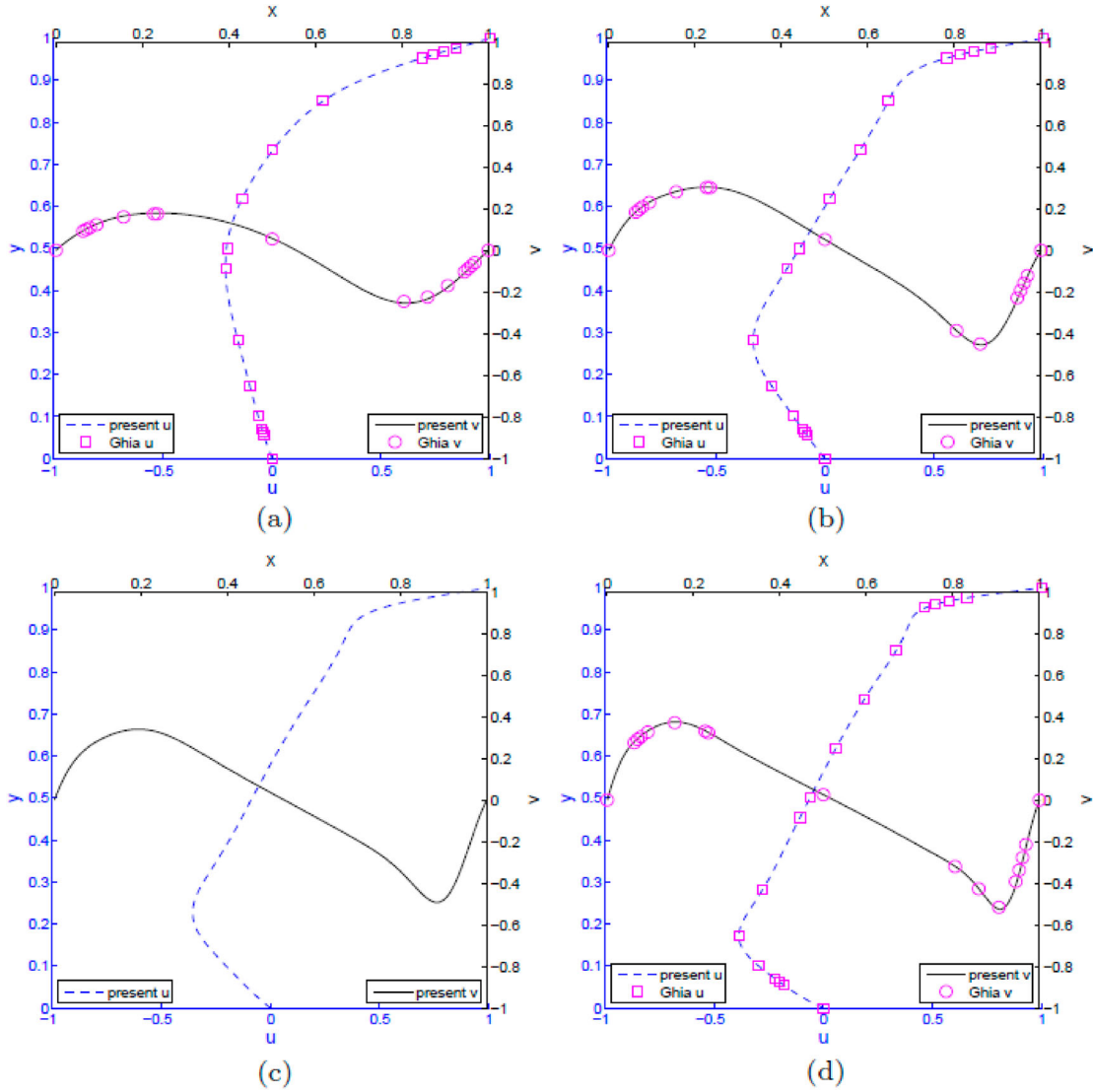
(Figure 12). The outer wall (cold wall) and inner wall (hot wall) are kept at constant temperatures  $T = 0$  and  $T = 1$ , respectively. By Boussinesq approximation, the temperature difference of the two walls generates NC.

Since the inner boundary is non-rectangular, the present CLIRBF approximation uses a Cartesian grid with added irregular boundary points as described in Figure 13.

The dimensionless governing equations for the NC problem are given by.

$$\frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} = -\omega, \quad (32)$$





**Figure 9.** The LDC fluid flow problem. Profiles of the  $u$  velocity along the vertical centreline (dashed – lines) and the  $v$  velocity along the horizontal centreline (solid lines) for several  $Reynolds$  numbers ( $Re = 100, 400, 600$  and  $1000$ ) by the present parallel method in comparison with the corresponding Ghia’s results (for  $u$  velocity and for  $v$  velocity). The parameters of the problem are given in Figure 7.

$$\sqrt{\frac{Pr}{Ra}} \left( \frac{\partial^2 \omega}{\partial x^2} + \frac{\partial^2 \omega}{\partial y^2} \right) = \frac{\partial \omega}{\partial t} + \left( u \frac{\partial \omega}{\partial x} + v \frac{\partial \omega}{\partial y} - \frac{\partial T}{\partial x} \right) \quad (33)$$

$$\frac{1}{\sqrt{PrRa}} \left( \frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} \right) = \frac{\partial T}{\partial t} + \left( u \frac{\partial T}{\partial x} + v \frac{\partial T}{\partial y} \right) \quad (34)$$

where  $Pr$  and  $Ra$  are the Prandtl number and the Rayleigh number, respectively and are defined as

$$Pr = \frac{\nu}{\alpha}, Ra = \frac{g\beta(\Delta T)L^3}{\nu\alpha}$$

where  $\nu$  is the kinematic viscosity,  $\alpha$  is the thermal diffusivity,  $g$  is the acceleration due to gravity,  $\beta$  is the thermal expansion coefficient,  $L$  is the characteristic

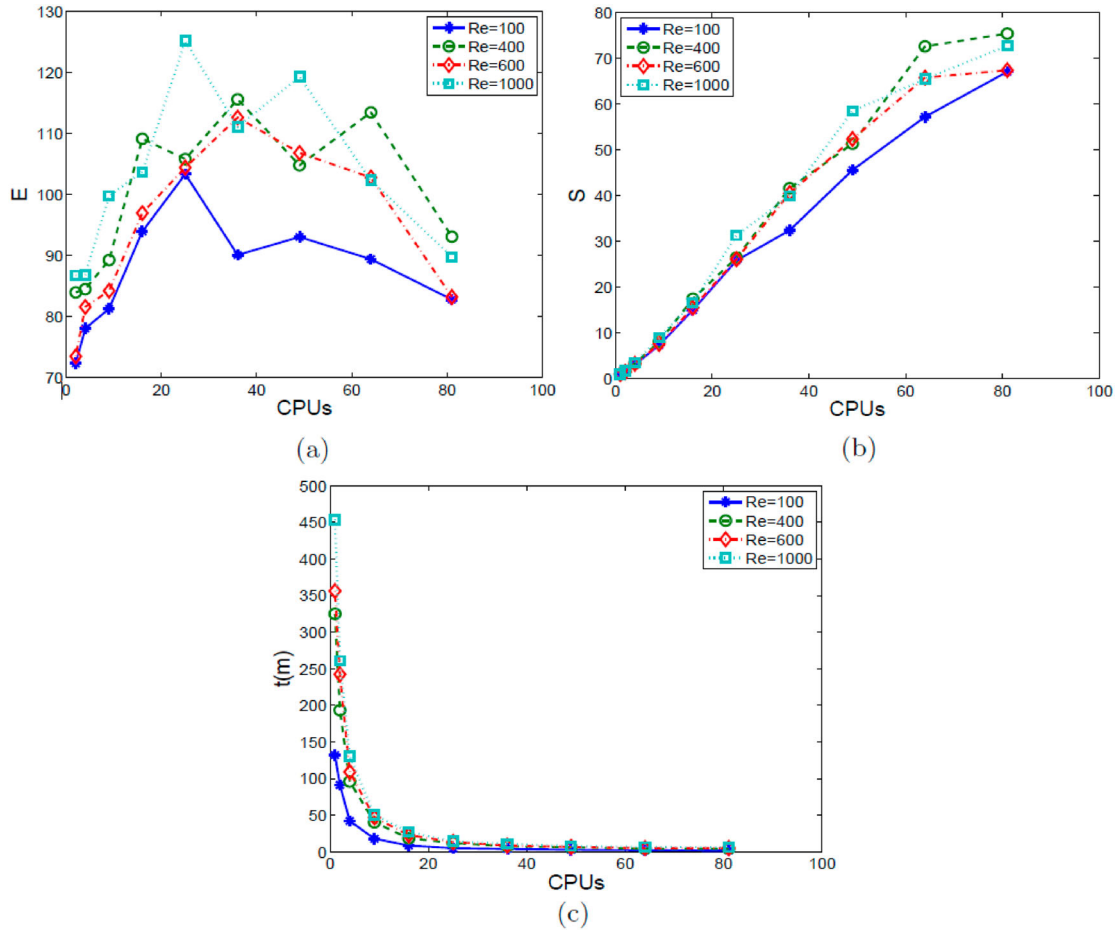
length and  $\Delta T$  is the difference between wall temperature and quiescent temperature. The problem is investigated using the following parameters

$$\frac{L}{2R} = 2.5, Pr = 0.71, Ra = \{10^4, 10^5, 10^6\}$$

where  $L$  is the length of sides of the outer square profile and  $R$  is the radius of the inner circular profile.

BCs of the NC problem presented in Figure 12 are written as follows.

$$\begin{aligned} \psi = 0, \quad \frac{\partial \psi}{\partial n} = 0, \quad T = 0 \quad & \text{on outer square;} \\ \psi = 0, \quad \frac{\partial \psi}{\partial x} = 0, \quad T = 1 \quad & \text{on inner square.} \end{aligned} \quad (35)$$



**Figure 10.** The LDC fluid flow problem. Performance of the present parallel algorithm for several Reynolds numbers (100, 400, 600 and 1000) with a grid of  $151 \times 151$  points: the efficiency, speed-up and throughput as a function of the number of CPUs. Other parameters are given in Tables 2 and 3.

**Table 2.** The LDC fluid flow problem.

CPU	Re = 100				Re = 400			
	$N_i$	$T_p$	S	E	$N_i$	$T_p$	S	E
1	8814	32.25	1.00	100.00	22107	324.98	1.00	100.00
2	13486	91.41	1.45	72.34	28594	193.52	1.68	83.97
4	13793	42.37	3.12	78.04	30770	96.15	3.38	84.50
9	14330	18.07	7.32	81.30	31931	40.45	8.03	89.27
16	16332	8.80	15.03	93.96	33649	18.61	17.46	109.14
25	15096	5.12	25.84	103.34	37848	12.29	26.45	105.81
36	17665	4.08	32.44	90.10	34088	7.81	41.62	115.60
49	16212	2.90	45.59	93.04	37932	6.33	51.35	104.79
64	16257	2.31	57.22	89.41	33964	4.48	72.60	113.44
81	15989	1.97	67.05	82.78	38383	4.31	75.40	93.08

Parallel performance with grid  $151 \times 151$ ,  $\Delta t = 10^{-3}$ ,  $CM_{tol}^{[u]} = 10^{-6}$ ,  $ABCM_{tol}^{[u]} = 10^{-6}$ ,  $ABCM_{tol}^{[\frac{\partial u}{\partial n}]} = 10^{-6}$ ,  $\theta = 0.45$  and  $\beta = 2$ . CPUs, number of CPUs (sub-domains);  $N_i$ , number of iterations;  $T_p$ , computation time (minutes) on parallel CPUs,  $T_p \equiv T_s$  when CPUs = 1; S: speed-up; E: efficiency.

Figure 14 suggests one way to partition the considered domain into 24 non-overlapping sub-domains. Each sub-domain is assigned a number from 1 to 24 as process-index. The process-index is to position sub-domains in the parallel system and to communicate with others, especially with their neighbour sub-domains.

About the BCs of sub-domains, the top and right artificial boundaries are of Dirichlet type and the bottom and left artificial boundaries are of Neumann type. A detailed example is described in Figure 15.

In each sub-domain, the algorithm for the NC problem using the present method is given as follows.

**Table 3.** The LDC fluid flow problem.

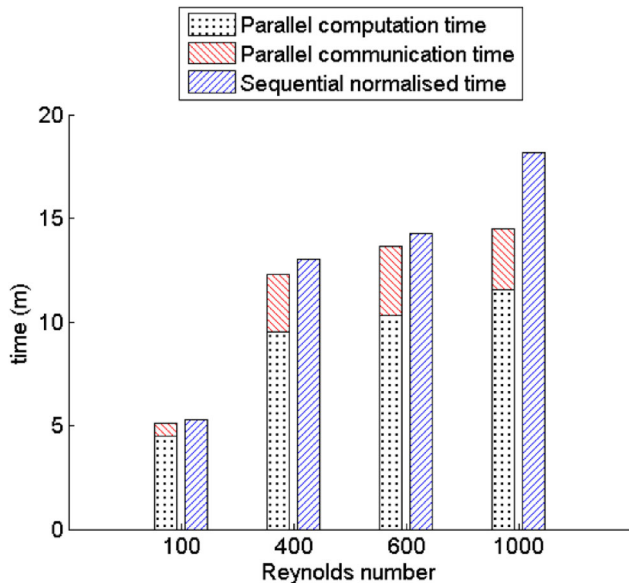
CPU	Re = 600				Re = 1000			
	$N_i$	$T_p$	$S$	$E$	$N_i$	$T_p$	$S$	$E$
1	25824	356.29	1.00	100.00	30442	453.65	1.00	100.00
2	33026	242.43	1.47	73.48	38044	261.37	1.74	86.78
4	35653	109.17	3.26	81.59	43619	130.61	3.47	86.83
9	35595	47.00	7.58	84.22	40853	50.51	8.98	99.80
16	38769	22.97	15.51	96.96	48079	27.34	16.60	103.72
25	36903	13.65	26.11	104.43	45085	14.49	31.30	125.20
36	38148	8.79	40.54	112.61	53924	11.35	39.98	111.06
49	40018	6.81	52.34	106.81	45231	7.76	58.44	119.27
64	40340	5.42	65.78	102.78	54754	6.93	65.49	102.33
81	46869	5.29	67.41	83.22	57140	6.24	72.73	89.79

Parallel performance with grid  $151 \times 151$ ,  $\Delta t = 10^{-3}$ ,  $CM_{tol}^{[u]} = 10^{-6}$ ,  $ABCM_{tol}^{[u]} = 10^{-6}$ ,  $ABCM_{tol}^{[\omega]} = 10^{-6}$ ,  $\theta = 0.45$  and  $\beta = 2$ . CPUs, number of CPUs (sub-domains);  $N_i$ , number of iterations;  $T_p$ , computation time (minutes) on parallel CPUs,  $T_p \equiv T_s$  when CPUs = 1;  $S$ , speed-up;  $E$ , efficiency.

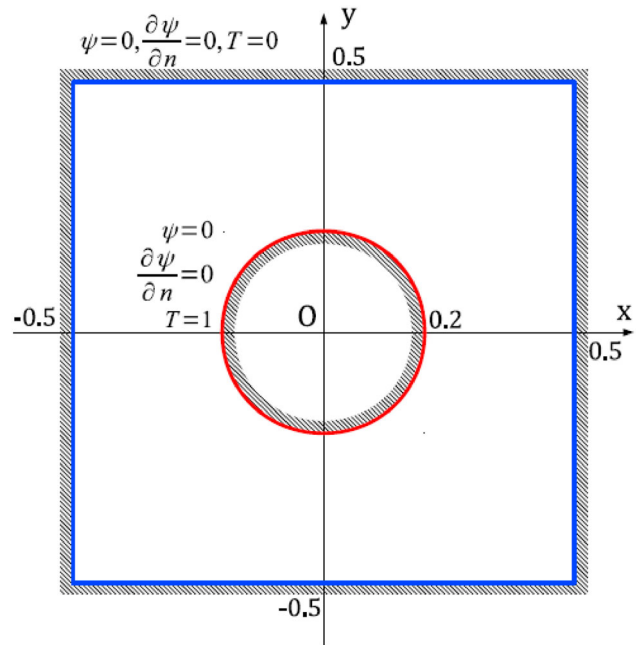
**Table 4.** The LDC fluid flow problem: Comparisons on the total computation time between the present parallel method and sequential scheme; and between computation time and communication time of the present parallel method.

Re	Parallel				Sequential		
	$T_{cmm}$	$T_{cmp}$	$T_p$	$\%T_{cmm}$	$T_s$	$T_n$	$E$
100	0.63	4.49	5.12	12.38	132.25	5.29	103.34
400	2.78	9.50	12.29	22.66	324.98	13.00	105.81
600	3.31	10.33	13.65	24.29	356.29	14.25	104.43
1000	2.92	11.58	14.49	20.13	453.65	18.15	125.20

Parallel program runs using 25 CPUs.  $Re$ , Reynolds number;  $T_{cmm}$ , communication time (minute);  $T_{cmp}$ , computation time (minute);  $T_p$ , total computation time (minutes) on parallel CPUs;  $\%T_{cmm}$ , percentage of communication time in  $T_p$ ;  $T_s$ , computation time (minutes) by sequential program;  $T_n$ , normalised time;  $E$ , efficiency

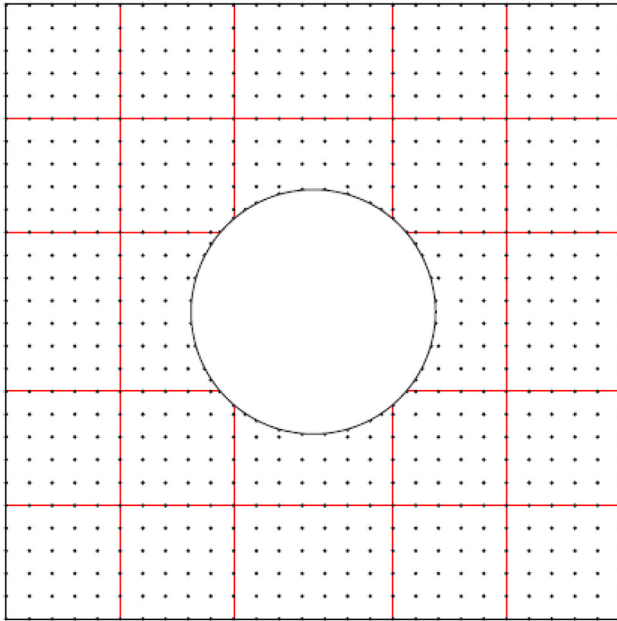


**Figure 11.** The LDC fluid flow problem. Comparison of execution times between parallel computation with 25 CPUs and the sequential computation using the present CLIRBF method.

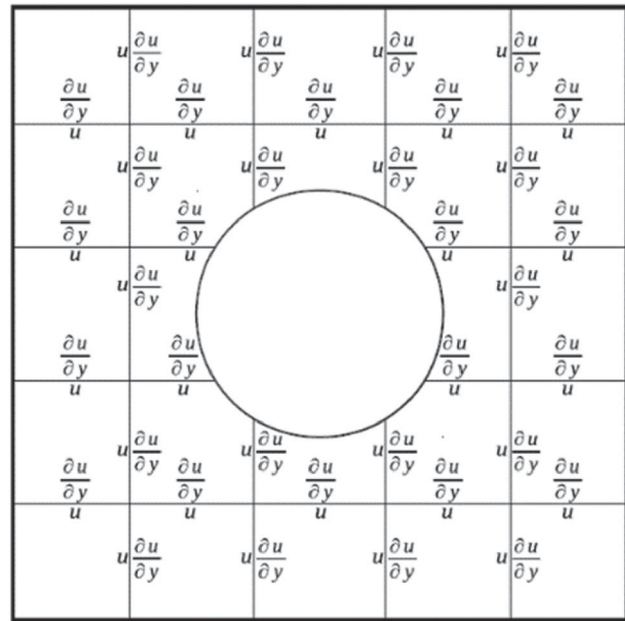


**Figure 12.** The NC in concentric annuli problem. Geometry and BCs are described.

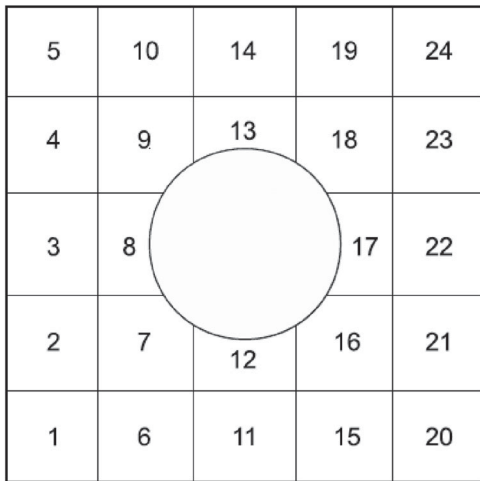
- (a) Set the initial values for  $\psi$ ,  $\omega$ ,  $T$ . Guess the initial values of BCs on artificial boundaries.
- (b) Solve Equation (32) for  $\psi$ .
- (c) Approximate  $u$ ,  $v$  and solve Equation (34) for  $T$ .
- (d) Approximate  $\omega$  on boundaries and solve Equation (33) for  $\omega$ .
- (e) Exchange the values of Dirichlet and/or Neumann BCs of  $\psi$ ,  $\omega$  and  $T$  over artificial boundaries.



**Figure 13.** The NC in concentric annuli problem. A sample grid of  $28 \times 28$  points with 24 sub-domains, one sub-domain is processed by one CPU.



**Figure 15.** The NC in concentric annuli problem with BCs on artificial boundaries of sub-domains.



**Figure 14.** The NC in concentric annuli problem with sub-domain formation and enumeration.

- (f) Check the termination conditions using (25). If the conditions are satisfied then terminate. Otherwise, update BCs on ABs and go to step (b).

As presented in step (d) of the algorithm, the value of  $\omega$  on boundary needs to be approximated. As described in Figure 13, since some boundary points on the inner cylinder do not coincide with grid points, equivalent formulas instead of Equation (40) are used

to determine  $\omega$  on artificial boundaries by (Le-Cao, Mai-Duy, and Tran-Cong 2009).

$$\omega_b = - \left[ 1 + \left( \frac{t_x}{t_y} \right)^2 \right] \frac{\partial^2 \psi_b}{\partial x^2} \text{ for boundary points on } x\text{-grid lines; and.}$$

$$\omega_b = - \left[ 1 + \left( \frac{t_y}{t_x} \right)^2 \right] \frac{\partial^2 \psi_b}{\partial y^2} \text{ for boundary points on } y\text{-grid lines,}$$

where  $t_x$  and  $t_y$  are the  $x$ - and  $y$ -components of the unit vector tangential to the boundary, respectively and the value of  $\omega$  on the inner cylinder can thus be approximated in only one dimension  $x$  or  $y$ . Such problem was solved using several different non-parallel methods by Moukalled and Acharya 1996; Kim et al. 2008; Hussain and Hussein 2010; Ngo-Cong et al. 2012.

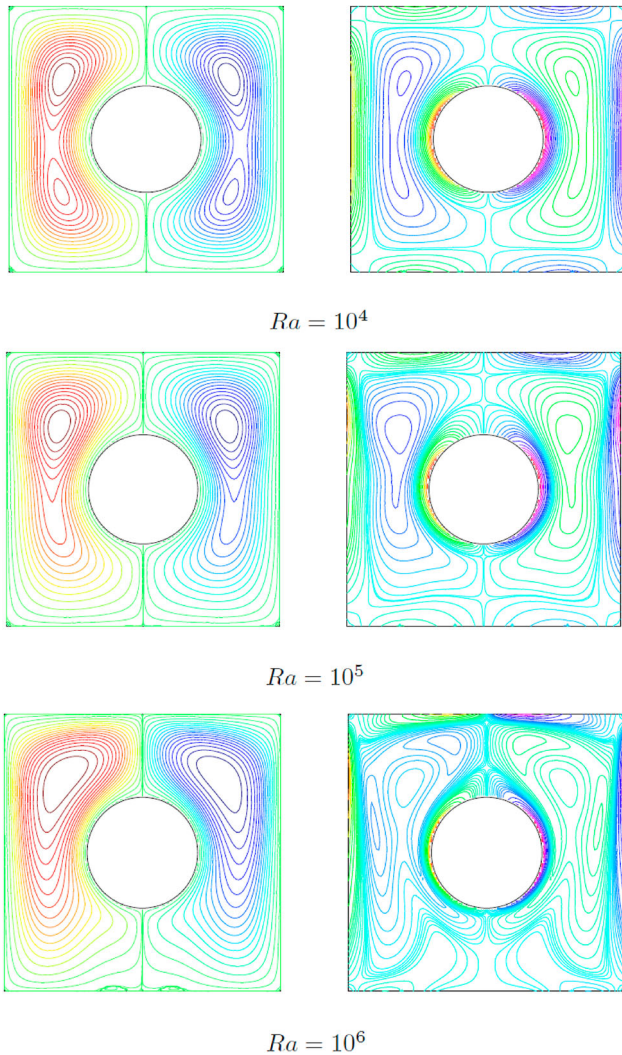
Like to consider the LDC problem, three different uniform grids, including  $92 \times 92$  points;  $120 \times 120$  points, and  $156 \times 156$  points, have been used to consider the convergence of the present parallel method using a range of sub-domains (2, 4, 8, 16, 24, 32, 48, 60, 72),  $\theta = 0.25$  and  $\beta = 2$ ,  $\Delta t = 10^{-4}$ ; and converges with given tolerances  $CM_{tol}^{[u]} = 10^{-9}$ ,  $ABCM_{tol}^{[u]} = 10^{-8}$  and  $ABCM_{tol}^{[\frac{\partial u}{\partial n}]} = 10^{-8}$  for the problem of NC in concentric annuli. The numerical experiments using the three grids of collocation points show that the present method has converged with the order  $10^{-8}$



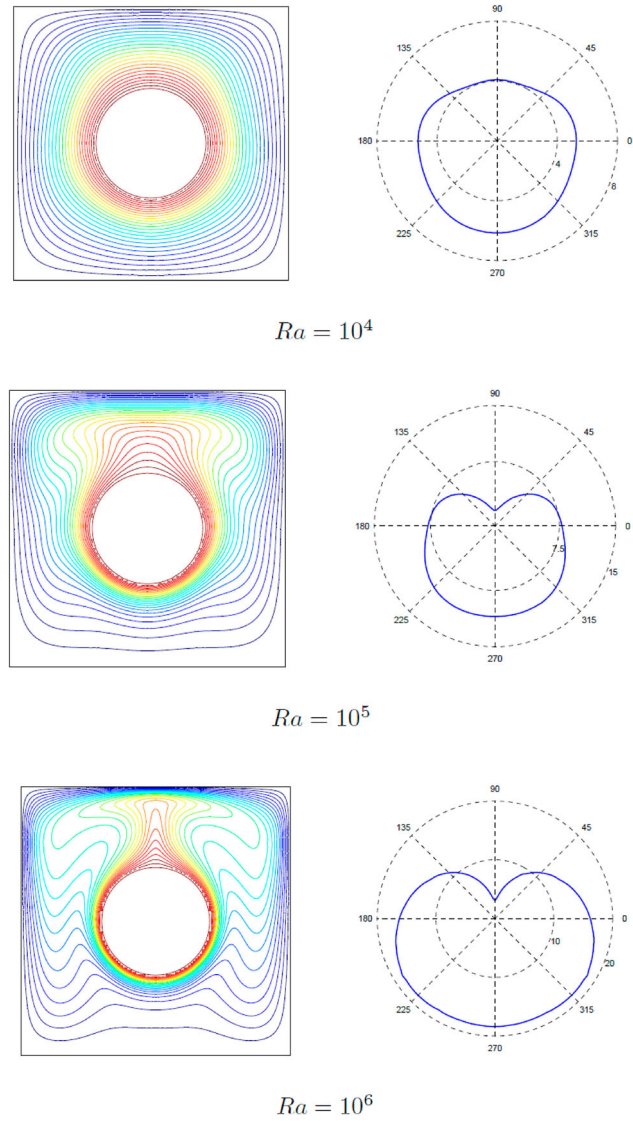
of average convergence measure of the whole analysis domain ( $CM_p$ ) for fluid with 3 Rayleigh numbers  $Ra = \{104, 105, 106\}$ .

The following is the detailed discussion on the accuracy and efficiency of the present method for the case using a grid of  $120 \times 120$  collocation points with 24 sub-domains.

Figures 16 and 17 show the stream-function ( $\psi$ ), vorticity ( $\omega$ ); temperature ( $T$ ) contours and normal derivative of temperature. Numerical results by the present CLIRBF parallel method shown are in very good agreement with those by other works cited above, using the same rheological parameters (Moukalled



**Figure 16.** The NC in concentric annuli problem. Stream-function ( $\psi$ ) contours (on the left) and vorticity ( $\omega$ ) contours (on the right) of the flow for 3 Rayleigh numbers  $Ra = \{10^4, 10^5, 10^6\}$  using 24 sub-domains using grid of  $120 \times 120$ ,  $\Delta t = 10^{-4}$ ,  $CM_{tol}^{[u]} = 10^{-9}$ ,  $ABCM_{tol}^{[u]} = 10^{-8}$ ,  $ABCM_{tol}^{[\frac{\partial u}{\partial n}]} = 10^{-8}$ ,  $\theta = 0.25$  and  $\beta = 2$ .



**Figure 17.** The NC in concentric annuli problem. Temperature ( $T$ ) contours (on the left) and normal derivative of temperature  $\frac{\partial T}{\partial n}$  along the inner circle (on the right) of the flow for 3 Rayleigh numbers  $Ra = \{10^4, 10^5, 10^6\}$  using 24 sub-domains. Other parameters are given in Figure 16.

and Acharya 1996; Kim et al. 2008; Le-Cao, Mai-Duy, and Tran-Cong 2009; Hussain and Hussein 2010; Ngo-Cong et al. 2012).

In order to verify the accuracy of the method, the average Nusselt numbers ( $\bar{Nu}$ ) associated with the outer and the inner walls are calculated to compare with those by other published works.

$$\bar{Nu} = \oint \frac{\partial T}{\partial n} ds \tag{36}$$

As presented in Table 5, the average Nusselt number on the hot wall by the present method is in great agreement with other results obtained by different

**Table 5.** The NC in concentric annuli problem.

Ra	10 <sup>4</sup>	10 <sup>5</sup>	10 <sup>6</sup>
Present method	3.23	4.91	8.90
Ngo-Cong et al. (2012) <sup>a</sup>	3.23	4.92	8.90
Le-Cao, Mai-Duy, and Tran-Cong (2009) <sup>b</sup>	3.21	4.89	8.85
Hussain and Hussein (2010) <sup>c</sup>	3.40	5.13	9.39
Kim et al. (2008) <sup>d</sup>	3.41	5.14	9.39
Shu and Zhu (2002) <sup>e</sup>	3.24	4.86	8.90
Moukalled and Acharya (1996) <sup>f</sup>	3.33	5.08	9.37

Comparison of average Nusselt numbers  $\bar{Nu}$ . Present results are obtained by our parallel algorithm with 24 CPUs.

<sup>a</sup>Using Local moving least square 1D-IRBFN.

<sup>b</sup>Using 1D-IRBFN.

<sup>c</sup>Using Finite volume method.

<sup>d</sup>Using Immersed boundary method.

<sup>e</sup>Using Differential quadrature method.

<sup>f</sup>Using Finite volume method.

**Table 6.** The NC in concentric annuli problem.

CPUs	$N_i$	$\bar{Nu}_o$	$\bar{Nu}_i$	$t(m)$	$S$	$E$
1	118288	3.2253	3.2245	1635.58	1.00	100.00
2	120297	3.2253	3.2249	446.17	3.67	183.29
4	120275	3.2253	3.2249	232.23	7.04	176.08
8	125224	3.2253	3.2247	124.47	13.14	164.26
16	124170	3.2252	3.2251	51.05	32.04	200.25
24	132465	3.2253	3.2253	33.48	48.85	203.56
32	131055	3.2253	3.2247	27.62	59.21	185.02
48	130681	3.2253	3.2262	18.37	89.05	185.52
60	135502	3.2253	3.2251	15.35	106.55	177.58
72	136952	3.2254	3.2244	13.35	122.48	170.11

Parallel performance with  $Ra = 10^4$ , grid  $120 \times 120$ ,  $\Delta t = 10^{-4}$ ,  $CM_{tol}^{[u]} =$

$10^{-9}$ ,  $ABC_{tol}^{[u]} = 10^{-8}$ ,  $ABC_{tol}^{[\frac{\partial u}{\partial n}]}$  =  $10^{-8}$ ,  $\theta = 0.25$  and  $\beta = 2$ . CPUs, number of CPUs (sub-domains);  $N_i$ , number of iterations;  $\bar{Nu}_o$ , average Nusselt number on the outer square (cold wall);  $\bar{Nu}_i$ , average Nusselt number on the inner circle (hot wall);  $t(m)$ , elapsed time (minutes);  $S$ , speed-up;  $E$ , efficiency.

non-parallel numerical methods with single domain. It is noted that the average Nusselt number in some of the works cited are calculated for half of the domain and thus in this work, the actual average Nusselt number needs to be divided by 2 for comparison purpose.

Tables 6–8 present numerical results of the NC problem in a concentric annulus using the present parallel computation method with a range of number of CPUs for three Rayleigh numbers  $Ra = \{10^4, 10^5, 10^6\}$ .

It can be seen that the variation of the average Nusselt numbers on the outer square and the inner circle with respect to several numbers of CPUs is insignificant. For example, the maximum differences in Nusselt numbers are around 0.08%, 0.11% and 0.13% for  $Ra = 10^4$ ,  $Ra = 10^5$  and  $Ra = 10^6$ , respectively (see Tables 6–8). This confirms the stability of the present parallel computation with different numbers of CPUs.

Furthermore, in this example, the super-linear speed-up is also observed with a whole range of

**Table 7.** The NC in concentric annuli problem.

CPUs	$N_i$	$\bar{Nu}_o$	$\bar{Nu}_i$	$t(m)$	$S$	$E$
1	386141	4.9053	4.9065	4542.72	1.00	100.00
2	392906	4.9053	4.9072	1509.73	3.01	150.45
4	401923	4.9053	4.9073	764.73	5.94	148.51
8	420629	4.9047	4.9063	420.99	10.79	134.88
16	422929	4.9054	4.9069	190.56	23.84	148.99
24	448339	4.9045	4.9072	115.11	39.46	164.43
32	443774	4.9040	4.9059	90.69	50.09	156.54
48	446269	4.9047	4.9100	62.49	72.69	151.44
60	446010	4.9030	4.9060	52.31	86.84	144.73
72	445250	4.9035	4.9055	45.80	99.19	137.76

Parallel performance with  $Ra = 10^5$ , grid  $120 \times 120$ , other parameters can be seen in Table 6.

**Table 8.** The NC in concentric annuli problem.

CPUs	$N_i$	$\bar{Nu}_o$	$\bar{Nu}_i$	$t(m)$	$S$	$E$
1	759280	8.8726	8.9015	9093.03	1.00	100.00
2	759000	8.8673	8.9034	2882.77	3.15	157.71
4	785182	8.8676	8.9034	1483.05	6.13	153.28
8	846039	8.8652	8.8990	833.02	10.92	136.45
16	895329	8.8686	8.9021	404.71	22.47	140.42
24	902945	8.8658	8.9043	245.91	36.98	154.07
32	902989	8.8655	8.8991	181.62	50.07	156.45
48	897475	8.8717	8.9075	126.41	71.93	149.85
60	900293	8.8668	8.9019	100.34	90.62	151.04
72	904160	8.8624	8.8956	88.91	102.28	142.05

Parallel performance with  $Ra = 10^6$ , grid  $120 \times 120$ , other parameters can be seen in Table 6.

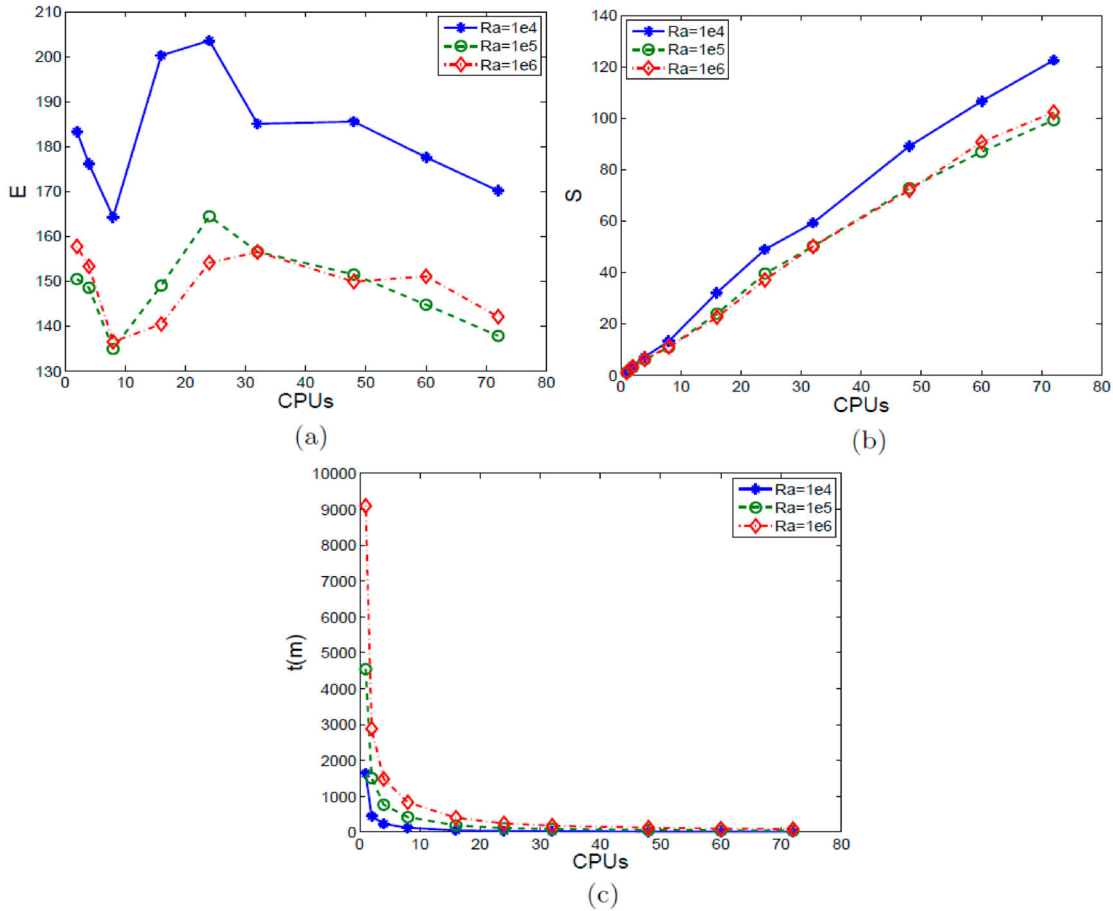
**Table 9.** Condition number of system matrix in LDC and NC problems with respect to number of CPUs (sub-domains)

LDC grid $151 \times 151$		NC grid $120 \times 120$	
CPUs	$CN_{\psi}$	CPUs	$CN_{\psi}$
1	10724.71	1	4563.29
2	4143.51	2	4316.68
4	2680.41	4	4036.09
9	1191.68	8	883.07
16	688.34	16	346.73
25	429.04	24	251.40
36	297.15	32	230.75
49	230.75	48	122.08
64	171.32	60	93.48
81	137.00	72	106.50

numbers of CPUs from 2 to 112 for all three considered Rayleigh numbers as found in Tables 6–8 (column E). One reason for the super-linearity can be explained by the significant reduction of condition number with the increase in the number of CPUs as seen in Table 9 for both the LDC and NC problems. For example, the condition number for the NC problem reduces from 4563.29 down to 106.50 when increasing the number of CPUs from 1 to 72 sub-domains with a  $120 \times 120$  grid.

Moreover, the efficiency and throughput of the present method are high as shown in Figure 18(a,c),





**Figure 18.** The NC in concentric annuli problem. Performance of the present parallel algorithm for several Rayleigh numbers: the efficiency, speed-up and throughput with respect to the number of CPUs. Other parameters are given in Table 6.

**Table 10.** The NC in concentric annuli problem.

Ra	Parallel				Sequential			E
	$T_{cmm}$	$T_{cmp}$	$T_p$	$\%T_{cmm}$	$T_s$	$T_n$		
$10^4$	10.07	23.41	33.48	30.07	1635.58	68.15	203.56	
$10^5$	42.03	73.09	115.11	36.51	4542.72	189.28	164.43	
$10^6$	64.95	180.96	245.91	26.41	9093.03	378.88	154.07	

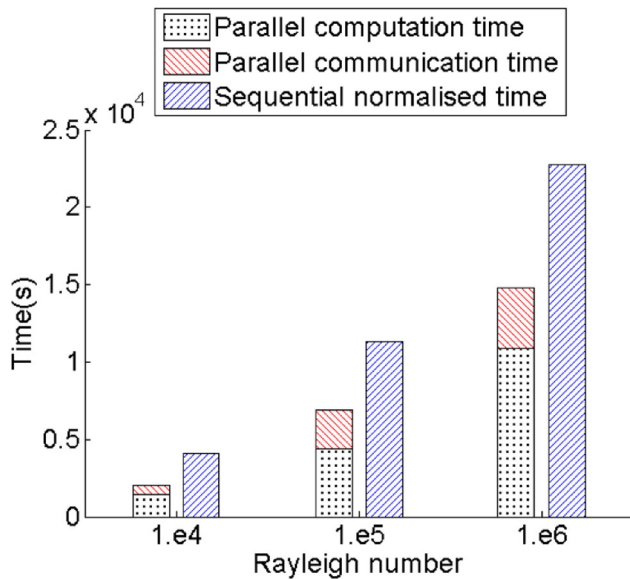
Comparison between the computation time and communication time in parallel program, and the total time between parallel computation and sequential computation. Parallel program using 24 CPUs.  $Ra$ , Rayleigh number;  $T_{cmm}$ , communication time (minute);  $T_{cmp}$ , computation time (minute);  $T_p$ , total computation time (minutes) on parallel CPUs;  $\%T_{cmm}$ , percentage of communication time in  $T_p$ ;  $T_s$ , computation time (minutes) by sequential program;  $T_n$ , normalised time;  $E$ , efficiency.

respectively. The speed-up for all three Rayleigh numbers presented in Figure 18(b) linearly increases with respect to the number of CPUs. Like in the LDC problem, this confirms the high scalability of our present parallel method.

The comparison of the communication time vs the computation time in parallel computing as well as the total execution time in the present parallel program using 24 CPUs vs the sequential one is given in Table 10. A bar graph is also provided in Figure 19 for better visualisation of these comparisons.

### 5. Conclusions

In this paper, the non-overlapping Dirichlet–Neumann DD method coupled with a 2D-CLIRBF method to solve the Navier–Stokes equations has been successfully developed. For the accuracy, the achieved results by the present parallel method for the LDC fluid flow and the NC in concentric annuli are in very good agreement with benchmark results. Meanwhile, the time efficiency and throughput are very high. Especially a super-linear speed-up is achieved with a wide range of numbers of CPUs. This super-linearity can be



**Figure 19.** The NC in concentric annuli problem: A bar graph for the comparison of execution times between the sequential computation and parallel computation using 24 CPUs.

partly explained by the impressive decrease of the condition number of the system matrix of sub-problems and by the insignificant increase in number of iterations required for the solution to converge. Furthermore, the variation of solutions with varying number of CPUs is very small. In addition, the method also showed good scalability as the speed-up grows consistently as a function of number of CPUs with given grids.

### Disclosure statement

No potential conflict of interest was reported by the author(s).

### ORCID

Canh-Dung Tran  <http://orcid.org/0000-0002-1011-4226>

### References

- Botella, O., and R. Peyret. 1998. "Benchmark Spectral Results on the Lid-Driven Cavity Flow." *Computers and Fluids* 27: 421–433. doi:10.1016/S0045-7930(98)00002-4
- Fasshauer, G. E. 1999. "Solving Partial Differential Equations by Collocation with Radial Basis Functions: Multilevel Methods and Smoothing." *Advances in Computational Mathematics* 11: 139–159. doi:10.1023/A:1018919824891
- Gander, M. J., and X. Tu. 2014. "Lecture Notes in Computational Science and Engineering." *Domain Decomposition Methods in Science and Engineering* 98: 597–605. doi:10.1007/978-3-319-05789-7\_57

- Ghia, U., K. N. Ghia, and C. T. Shin. 1982. "High-Re Solutions for Incompressible Flow Using the Navier-Stokes Equations and a Multigrid Method." *Journal of Computational Physics* 48: 387–411. doi:10.1016/0021-9991(82)90058-4
- Haynsworth, E. V. 1968. *On the Schur Complement*. Basel Mathematical Notes, BMN 20, 17.
- Hoang-Trieu, T.-T., N. Mai-Duy, and T. Tran-Cong. 2012. "Several Compact Local Stencils Based on Integrated RBFs for Fourth-Order ODEs and PDEs." *CMES Computer Modeling in Engineering and Sciences* 84 (2): 171–203.
- Hussain, S. H., and A. K. Hussein. 2010. "Numerical Investigation of Natural Convection Phenomena in a Uniformly Heated Circular Cylinder Immersed in Square Enclosure Filled with Air at Different Vertical Locations." *International Communications in Heat and Mass Transfer* 37 (8): 1115–1126. doi:10.1016/j.icheatmasstransfer.2010.05.016
- Kim, B., D. Lee, M. Ha, and H. Yoon. 2008. "A Numerical Study of Natural Convection in a Square Enclosure with a Circular Cylinder at Different Vertical Locations." *International Journal of Heat and Mass Transfer* 51 (7–8): 1888–1906. doi:10.1016/j.ijheatmasstransfer.2007.06.033
- Le-Cao, K., N. Mai-Duy, and T. Tran-Cong. 2009. "An Effective Integrated-RBFN Cartesian-Grid Discretization for the Stream Function–Vorticity–Temperature Formulation in Nonrectangular Domains." *Numerical Heat Transfer, Part B: Fundamentals* 55 (6): 480–502. doi:10.1080/10407790902827470
- Mai-Duy, N., and T. Tran-Cong. 2001. "Numerical Solution of Differential Equations Using Multiquadric Radial Basis Function Networks." *Neural Networks* 14: 185–199. doi:10.1016/S0893-6080(00)00095-2
- Mai-Duy, N., and T. Tran-Cong. 2010. "A Numerical Study of 2D Integrated RBFNs Incorporating Cartesian Grids for Solving 2D Elliptic Differential Problems." *Numerical Methods for Partial Differential Equations* 26: 1443–1462. doi:10.1002/num.20502
- MathWork. 2012. *Parallel Computing Toolbox User's Guide R2012b (6.1)*. Natick, Massachusetts: MathWorks Inc.
- Moukalled, F., and S. Acharya. 1996. "Natural Convection in the Annulus Between Concentric Horizontal Circular and Square Cylinders." *Journal of Thermophysics and Heat Transfer* 10 (3): 524–531. doi:10.2514/3.820
- Ngo-Cong, D., N. Mai-Duy, W. Karunasena, and T. Tran-Cong. 2012. "Local Moving Least Square – One-Dimensional IRBFN Technique: Part I – Natural Convection Flows in Concentric and Eccentric Annuli." *MES Computer Modeling in Engineering and Sciences* 83 (3): 275–310.
- Nguyen, H., and C. D. Tran. 2022. "Simulation of Non-Dilute Fibre Suspensions Using RBF-Based Macro–Micro Multi-scale Method." *Korea-Australia Rheology Journal* 34 (1): 1–15. doi:10.1007/s13367-022-00022-1
- Nguyen, H. Q., C.-D. Tran, and T. Tran-Cong. 2015. "A Multi-scale Method Based on the Fibre Configuration Field, IRBF and DAVSS for the Simulation of Fibre Suspension Flows." *CMES – Computer Modeling in Engineering and Sciences* 109 (4): 361–403.

- Pham-Sy, N., C.-D. Tran, T.-T. Hoang-Trieu, N. Mai-Duy, and T. Tran-Cong. 2013. "Compact Local IRBF and Domain Decomposition Method for Solving PDEs Using a Distributed Termination Detection Based Parallel Algorithm." *CMES Computer Modeling in Engineering and Sciences* 92 (1): 1–31.
- Quarteroni, A., and A. Valli. 1999. *Domain Decomposition Methods for Partial Differential Equations*. Oxford: Clarendon Press.
- Schwarz, H. A. 1869. "Ueber Einige Abbildungsaufgaben." *Journal Fur Die Reine Und Angewandte Mathematik* 70: 105–120.
- Shu, C., and Y. D. Zhu. 2002. "Efficient Computation of Natural Convection in a Concentric Annulus Between an Outer Square Cylinder and an Inner Circular Cylinder." *International Journal for Numerical Methods in Fluids* 38: 429–445. doi:10.1002/flid.226
- Smith, B. F., P. E. Bjorstad, and W. D. Gropp. 1996. *Domain Decomposition: Parallel Multilevel Methods for Elliptic Partial Differential Equations*. Cambridge: Cambridge University Press.
- Tran, C. D., N. Mai-Duy, K. Le-Cao, and T. Tran-Cong. 2012. "A Continuum-Microscopic Method Based on IRBFs and Control Volume Scheme for Viscoelastic Fluid Flows." *CMES: Computer Modeling in Engineering and Sciences* 85 (6): 499–519.
- Tran, C.-D., D. G. Phillips, and T. Tran-Cong. 2009. "Computation of Dilute Polymer Solution Flows Using BCF-RBFN Based Method and Domain Decomposition Technique." *Korea-Australia Rheology Journal* 21 (1): 1–12.
- Tran-Canh, D., and T. Tran-Cong. 2002. "Computation of Viscoelastic Flow Using Neural Networks and Stochastic Simulation." *Korea-Australia Rheology Journal* 14 (2): 161–174.
- Tran-Canh, Dung, and T. Tran-Cong. 2004. "Element-Free Simulation of Dilute Polymeric Flows Using Brownian Configuration Fields." *Korea-Australia Rheology Journal* 26 (1): 1–15.