The multi-core optimization of the unbalanced calculation in the clean numerical simulation of Rayleigh-Bénard turbulence

Lu Li[1], Zhiliang Lin[2], and Yan Hao[1]

[1] Asia Pacific R & D Center Intel, Shanghai, China
[2] Shanghai Jiaotong University,Shanghai, China

Abstract. The so-called clean numerical simulation (CNS) is used to simulate the Rayleigh-Bénard (RB) convection system. Compared with direct numerical simulation (DNS), the accuracy and reliability of investigating turbulent flows improve largely. Although CNS can well control the numerical noises, the cost of calculation is more expensive. In order to simulate the system in a reasonable period, the calculation schemes of CNS require redesign. In this paper, aiming at the CNS of the two-dimension RB system, we first propose the notions of equal difference matrix and balance point set which are crucial to model the unbalanced calculation of the system under multi-core platform. Then, according to the notions, we present algorithms to optimize the unbalanced calculation. We prove our algorithm is optimal when the core number is the power of 2 and our algorithm approaches the optimal when the core number is not the power of 2. Finally, we compare the results of our optimized algorithms with others to demonstrate the effectiveness of our optimization.

Keywords: turbulence, clean numerical simulation, unbalanced calculation

1 Introduction

It is of broad interest to understand the evolution of non-equilibrium systems involves energy exchange through the system boundary with the surroundings, for example, Rayleigh-Bénard (RB) system. With the help of direct numerical simulation (DNS), we can get high resolution, both spatially and temporally. However, because of the numerical noises, e.g. truncation error and round-off error, are inevitable in DNS, the solution reliability remains controversial. Fortunately, the so-called clear numerical simulation (CNS) can well control such kind of numerical uncertainty. Aiming at investigating the laminar-turbulent transition of the two-dimension Rayleigh-Bénard (RB) system, the numerical noise of CNS can be well controlled even much lower than the microscopic thermal compared fluctuation [11]. Although CNS is more accurate, the cost of calculation is more expensive. In order to simulate the system in a reasonable period, the calculation schemes of CNS require optimization. In this paper, we first propose

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93713-7_70

the notions of equal difference matrix and balance point set which are crucial to model the unbalanced calculation under multi-core platform. Then, according to the notions, we present algorithms to optimize the unbalanced calculation.

One significant property of our proposed optimization algorithm is provable to optimal when the core number is the power of 2 and to approach the optimal when the core number is not the power of 2 theoretically. We first model the amount of the calculation as equal difference matrix, which is the basis to understand and optimize the unbalanced calculation. The equal difference matrix reveals the relationship of the calculation amount between different grid points in spectral space quantitatively. Based on the properties of the equal difference matrix, we propose the notion of balance point set, which is the key to optimize the unbalanced calculation.

The other significant property of our proposed algorithm is high efficiency. For arbitrary cores under multi-core platform, our algorithm can complete the assignment of the calculation to different cores on the stage of the initialization procedure. On one hand, the arrangement does not disturb the calculation of CNS. It is straightforward to integrate our method into the original CNS. On the other hand, the arrangement is only done once; the time overhead of the assignment is little.

This paper is organized as follows: In Section 2, we discuss relevant methods in literature. In section 3, first, we describe the CNS of the two-dimensional Rayleigh-Bénard system. Then, we establish the model of the calculation amount of the simulation and present the general notions of equal difference matrix and balance point set. Based on the proposed notions, we present the algorithms to optimize the unbalanced calculation under multi-core platform. In Section 4, we use several concrete examples to demonstrate the effectiveness of our proposed algorithms, followed by the conclusion in Section 5. In the Appendix, we give the proofs adopted by our algorithms.

2 Related Work

Direct numerical simulation (DNS) [1, 15, 3] provides an effective way to understand the evolution of non-equilibrium system involving energy exchange through the system boundary with the surroundings. However, because of the inevitable numerical noises, e.g. truncation error and round-off error, the solution reliability provided by DNS is very controversial [23]. For example, Lorenz discovered the dynamic systems governed by the Navier-Stokes equations are chaotic due to the butterfly effect no only depending on the initial conditions [12] but also on numerical algorithms [13]. Furthermore, [20, 16] reported some spurious turbulence evolution cases provided by DNS.

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93713-7_70

Fortunately, the inevitable numerical noises can be well controlled by clean numerical simulation (CNS)[7, 21, 8, 10, 6, 9]. CNS adopts arbitrary-order Taylor series method (TSM) [2] and arbitrary multiple-precision (MP) data [4] to reduce the round-off error and truncation error, respectively. Lin et al. [11] simulated Saltzman's model of Rayleigh-Bénard convection by means of CNS with considering the propagation of the inherent micro-thermal fluctuation. Compared with DNS, CNS can investigate turbulent flows with well-improved reliability and accuracy.

Numerically, introducing TSM and MP, the data and the calculation amount of CNS is much larger than DNS. In order to simulate the system in a reasonable period, CNS requires large amount of computing resources from a high performance computing (HPC) cluster. However, because of the complexity of the parallel computing there exists difficulty in fully utilizing the capacity and scalability of the HPC cluster [24, 25, 22]. In order to achieve high performance, it is necessary to re-design the CNS calculation scheme from various aspects including parallel algorithm, data arrangement, etc. Even though Lin et al. [11] have optimized the data arrangement to reduce the amount of simulation data by utilizing the symmetry of the two-dimensional Rayleigh-Bénard system.

3 Methods

3.1 2-D CNS Rayleigh-Bénard

The two-dimensional Rayleigh-Bénard system has been extensively studied [17, 19, 5, 18, 14, 26]. Following Saltzman [19], the corresponding non-dimensional governing equations in the form of stream function ψ with the Boussinesq approximation is

$$\begin{cases} \frac{\partial}{\partial t}\nabla^2\psi + \frac{\partial(\psi,\nabla^2\psi)}{\partial(x,z)} - \frac{\partial\theta}{\partial x} - C_a\nabla^4\psi = 0 \\ \frac{\partial\theta}{\partial t} + \frac{\partial(\psi,\theta)}{\partial(x,z)} - \frac{\partial\psi}{\partial x} - C_b\nabla^2\theta \end{cases} \tag{1}$$

where t denotes time, θ denotes the temperature departure from a linear variation background, x and z represent the horizontal and vertical spatial coordinates, $C_a = \sqrt{\mathrm{Pr}\,/Ra}$ and $C_b = 1/\sqrt{\mathrm{Pr}\,Ra}$ with the Prandtl number $\mathrm{Pr} = \nu/\kappa$, where ν is the kinematic viscosity, κ is the thermal diffusivity, and the Rayleigh number $Ra = g\alpha H^3\Delta T/\nu\kappa$, where H is the distance between the two parallel free surfaces, g is the gravity acceleration and α is the thermal expansion coefficient of the fluid, ΔT is the prescribed constant temperature difference.

As described by Saltzman [19], we use the double Fourier expansion modes to expand the stream function ψ and temperature departure θ as follows:

$$\begin{cases} \psi(x,z,t) = \sum\limits_{m=-\infty}^{+\infty}\sum\limits_{n=-\infty}^{+\infty}\Psi_{m,n}(t)\exp[2\pi Hi(\frac{m}{L}x + \frac{n}{2H}z)] \\ \theta(x,z,t) = \sum\limits_{m=-\infty}^{+\infty}\sum\limits_{n=-\infty}^{+\infty}\Theta_{m,n}(t)\exp[2\pi Hi(\frac{m}{L}x + \frac{n}{2H}z)] \end{cases} \tag{2}$$

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93713-7_70

where m and n are the wave numbers in the x and z directions, $\Psi_{m,n}(t)$ and $\Theta_{m,n}(t)$ are the expansion coefficients of the stream function and temperature components with wave numbers (m, n). Separate the real part from the imaginary part:

$$\begin{cases} \Psi_{m,n} = \Psi_{1,m,n} - i\Psi_{2,m,n} \\ \Theta_{m,n} = \Theta_{1,m,n} - i\Theta_{2,m,n} \end{cases} \tag{3}$$

Following Lin et al. [11], denote the time increment as δt and $f(j)$ as the value of $f(t)$ at $t = j\Delta t$. We use the P th-order Taylor series to expand $\Psi_{i,m,n}$ and $\Theta_{i,m,n}$ as follows:

$$\begin{cases} \Psi_{i,m,n}^{(j+1)} = \Psi_{i,m,n}(t_j + \Delta t) = \Psi_{i,m,n}^{(j)} + \sum_{k=1}^{P} \beta_{i,m,n}^{j,k} (\Delta t)^k \\ \Theta_{i,m,n}^{(j+1)} = \Theta_{i,m,n}(t_j + \Delta t) = \Theta_{i,m,n}^{(j)} + \sum_{k=1}^{P} \gamma_{i,m,n}^{j,k} (\Delta t)^k \end{cases} \tag{4}$$

where

$$\begin{aligned} &\beta_{1,m,n}^{j,k+1} \\ &= \Bigg(\sum_{p=-M}^{M} \sum_{q=-N}^{N} C_{m,n,p,q} \frac{\alpha_{p,q}^2}{\alpha_{m,n}^2} \sum_{l=0}^{k} [\beta_{1,p,q}^{j,l} \beta_{1,m-p,n-q}^{j,k-l} \\ &\quad + \beta_{2,p,q}^{j,l} \beta_{2,m-p,n-q}^{j,k-l}] + \frac{l^* m}{\alpha_{m,n}^2} \gamma_{2,m,n}^{j,k} \\ &\quad - C_a \alpha_{m,n}^2 \beta_{1,m,n}^{j,k} \Bigg) / (1+k) \end{aligned} \tag{5}$$

$$\begin{aligned} &\beta_{2,m,n}^{j,k+1} \\ &= \Bigg(\sum_{p=-M}^{M} \sum_{q=-N}^{N} C_{m,n,p,q} \frac{\alpha_{p,q}^2}{\alpha_{m,n}^2} \sum_{l=0}^{k} [\beta_{1,p,q}^{j,l} \beta_{2,m-p,n-q}^{j,k-l} \\ &\quad + \beta_{2,p,q}^{j,l} \beta_{1,m-p,n-q}^{j,k-l}] + \frac{l^* m}{\alpha_{m,n}^2} \gamma_{1,m,n}^{j,k} \\ &\quad - C_a \alpha_{m,n}^2 \beta_{2,m,n}^{j,k} \Bigg) / (1+k) \end{aligned} \tag{6}$$

$$\begin{aligned} &\gamma_{1,m,n}^{j,k+1} \\ &= \Bigg(- \sum_{p=-M}^{M} \sum_{q=-N}^{N} C_{m,n,p,q} \sum_{l=0}^{k} [\, \beta_{1,p,q}^{j,l} \gamma_{1,m-p,n-q}^{j,k-l} \\ &- \beta_{2,p,q}^{j,l} \gamma_{2,m-p,n-q}^{j,k-l}] + l^* m \beta_{2,m,n}^{j,k} \\ &- C_a \alpha_{m,n}^2 \gamma_{1,m,n}^{j,k} \Bigg) / (1+k) \end{aligned} \tag{7}$$

$$\begin{aligned} &\gamma_{2,m,n}^{j,k+1} \\ &= \Bigg(- \sum_{p=-M}^{M} \sum_{q=-N}^{N} C_{m,n,p,q} \sum_{l=0}^{k} [\, \beta_{1,p,q}^{j,l} \gamma_{2,m-p,n-q}^{j,k-l} \\ &- \beta_{2,p,q}^{j,l} \gamma_{1,m-p,n-q}^{j,k-l}] + l^* m \beta_{1,m,n}^{j,k} \\ &- C_a \alpha_{m,n}^2 \gamma_{2,m,n}^{j,k} \Bigg) / (1+k) \end{aligned} \tag{8}$$

with $C_{m,n,p,q} = \pi l^*(mq - np)$, $l^* = 2\pi H/L$ and $\alpha_{m,n}^2 = l^{*2} m^2 + \pi^2 n^2$.

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93713-7_70

Numerically, For point (p, q) in the Eq. (5) (6) (7) (8), $-M \leq m - p \leq M$ and $-N \leq n - q \leq N$ should be satisfied. So that let F_i, where $i = \{1, 2, 3, 4\}$, present the first term of left hand side of the four equations; $f_i(m, n)$, where $i = \{1, 2, 3, 4\}$, present the inside part of the 2-D summations, the formulas of the Eq. (5) (6) (7) (8) can be optimized to

$$F_i = (\sum_{p=m-M}^{M} \sum_{q=n-N}^{N} f_i(m, n))/(1 + k) \tag{9}$$

In the spectral space, to calculate $F_i, i = \{1, 2, 3, 4\}$ of the point $(0, 0)$, the 2-D summation height is from $-M$ to M and the width is from $-N$ to N. In comparison, for the point (M, N), which means that $m = M$ and $n = N$, the 2-D summation height is from 0 to M and the width is from 0 to N. The calculation amount of the 2-D summation for the point $(0, 0)$ is 4 times more than that for the point (M, N). Under multi-core platform, different gird points in Eq. (9) are assigned to different cores, the unbalanced calculation exists.

For CNS calculation, the unbalanced calculation is even worse. In one hand, the CNS calculation adopts arbitrary-precision binary floating-point computation (MPFR) [4] as the data type in order to avoid round-off error. However, the calculation amount using MPFR is much more than adopting double as the data type. In the other hand, the P th-order Taylor series aiming to avoid truncation error, also increases the amount of calculation. After adding the two new features, the amount of the calculation is much more than that of the scheme using double data type, which leads to the unbalanced calculation getting worse.

3.2 Equal Difference Matrix

Although the four equations $F_i, (i = 1, 2, 3, 4)$ are coupled, the calculations for the points in the mesh grid of two-dimensional spectral space are independent to each other because we adopt explicit calculation scheme for each sub-step in each iteration. Here the sub-step means Taylor step $(p = 1, 2, .., P)$. In order to analyze the unbalanced calculation for F_i, we obtain the amount of the calculation of each grid point in the spectral space. Let the grid size be $M * N$, for the point (m, n), the calculation amount of $F_i, i = (1, 2, 3, 4)$ is $(2M - m) * (2N - n) * (\sum_{i=1}^{4} \sum_{p=1}^{k} t_{i,p})$, where k is the order of the Taylor expansion $(k = 1, ..., P)$, $2M - m$ and $2N - n$ are the summation times for the height and width of the point (m, n), t_i is the calculation for $f_i, i = (1, 2, 3, 4)$, which are constants for a given computing environment.

We arrange all the calculation amounts into a matrix according to each grid index. Eq. (10) gives the calculation amount matrix, $t_i, i = (1, 2, 3, 4)$ are omitted so as to simplify the analysis.

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93713-7_70

$$\begin{bmatrix} 4MN & 2M(2N-1) & ... & 2M(N+1) \\ 2N(2M-1) & (2M-1)(2N-1) & ... & (N+1)(2M-1) \\ ... & ... & ... & ... \\ 2N(M+1) & (M+1)(2N-1) & ... & (N+1)(M+1) \end{bmatrix}_{m*n} \tag{10}$$

For a given row j, the row array is $\{a_n\} = \{2N(2M-j), (2N-1)(2M-j)..., (N+1)(2M-j)\}$, the difference between two adjacent items is $a_{j+1} - a_j = 2M - j$, which is a constant for the given row j. So that the row j is an equal difference array with $2M - j$ as the common difference. Similarly, for a given column i in Eq. (10), the column i is an equal difference array with $2N - i$ as the common difference. As a general extension to the equal difference array, we define the matrix of Eq. (10) as equal difference matrix.

Based on the equal difference matrix, we can model the issue of the unbalanced calculation as follows: Given an equal difference matrix EM with size $M * N$ and the core number r, the calculation of Eq. (9) under multi-core platform means dividing EM into r different sets for different r cores, the differences of the sums of the divided sets should be as small as possible.

3.3 Balance Point Set

Inspired by the Theorem 1 in the Appendix, we consider the relationship between the sums of point sets E_0 and E_1 shown in Fig. 1. $E_0 = \{a_1, a_2, a_3, a_4\}$ and $E_1 = \{b_1, b_2, b_3, b_4\}$ are in the equal difference matrix EM. $\{a_1, a_2, a_3, a_4\}$ are the four corners of the matrix. b_1 and b_2 are mirror symmetry about $n = N/2$. b_1 and b_3 are mirror symmetry about $m = M/2$. b_2 and b_4 are mirror symmetry about $m = M/2$. The 4 points form a rectangle, the center is $(M/2, N/2)$. We give point set $E_2 = \{c_1, c_2, c_3, c_4\}$ shown in Fig. 1 as an intermediate variable. c_1, c_2, b_1 and b_2 are in the same row. c_3, c_4, b_3 and b_4 are in the same row.
First we consider the relationship between the sum of E_0 and E_2. According to Theorem 1 in the Appendix,

$$a_1 + a_3 = c_1 + c_3 \tag{11}$$

$$a_2 + a_4 = c_2 + c_4 \tag{12}$$

So that

$$\sum_{i=1}^{4} a_i = \sum_{i=1}^{4} c_i \tag{13}$$

Similarly, we can get the relationship between the sum of E_1 and E_2

$$\sum_{i=1}^{4} b_i = \sum_{i=1}^{4} c_i \tag{14}$$

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93713-7_70

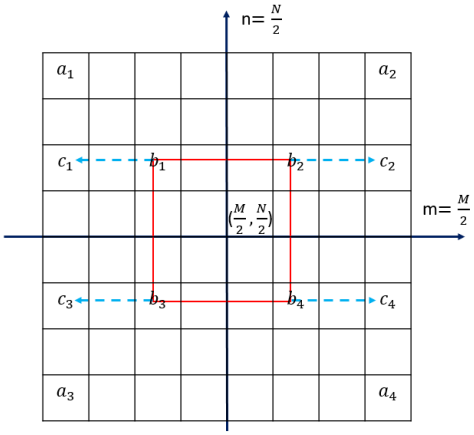

Fig. 1. The relationship between $E_0 = \{a_1, a_2, a_3, a_4\}$, $E_1 = \{b_1, b_2, b_3, b_4\}$ and $E_2 = \{c_1, c_2, c_3, c_4\}$ in equal difference matrix. E_0 is the set containing the 4 corner elements. The 4 elements of E_1 form a rectangle, the center is $(M/2, N/2)$. E_2 extends the x axis of E_1 to the border of the equal difference matrix.

So that

$$\sum_{i=1}^{4} a_i = \sum_{i=1}^{4} b_i \tag{15}$$

Eq. (15) indicates that the sum of an arbitrary 4 points in EM that form a horizontal rectangle with $(M/2, N/2)$ as center is a constant. Thus the equal difference matrix can be divided into $(M * N/4)$ sets, the sum of each set is the same. We define each set as balanced point set. The excellent property provides the basis to solve the unbalanced calculation.

Let $(M/2, N/2)$ be the origin, row $M/2$ be the horizontal axis, column $N/2$ be the vertical axis, The equal difference matrix EM is divided into four quadrants. After an arbitrary point (i, j) is assigned to the second quadrant, the balanced point set where $b_{i,j}$ can be calculated as follows:

$$E_{i,j} = \begin{cases} b_{i,j} & b_{i,N-j+1} \\ b_{M-i+1,j} & b_{M-i+1,N-j+1} \end{cases} \tag{16}$$

where the four elements form the balanced point set $E_{i,j}$, the first row points are in the second and first quadrant respectively, the second row points are in the third and fourth quadrant respectively. Using the attribute of the equal difference array, the relationship between the 4 elements is as follows:

$$b_{i,j} > b_{i,N-j+1}, b_{M-i+1,j} > b_{M-i+1,N-j+1} \tag{17}$$

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93713-7_70

3.4 Unbalanced Calculation Optimization

Based on the balanced point set, the unbalanced calculation of different cores under multi-core platform can be optimized in two cases. Let the equal difference matrix be EM with size $M * N$. Without loss of generality, the size M and N are powers of 2 defined as $M = 2^u$ and $N = 2^v$.

Case 1 We first consider the core number of the multi-core platform is r, where r is the power of 2. This means EM is divided into r different sets. When $r = 2^u * 2^v/4$, the division of the balanced point sets $E_{i,j}$ satisfies, where $1 \leq i \leq M/2$ and $1 \leq j \leq N/2$. When $r = 2^u * 2^v/(4 * 2^k)$, where $1 \leq k \leq u + v - 2$, we can merge arbitrary 2^k $E_{i,j}$ together as a division part for a core because the sum of of each $E_{i,j}$ is a constant. So that for each core, the calculation amount is the same. That is to say, the unbalanced calculation between different cores is solved theoretically. We develop the determined Algorithm (1) so as to programme conveniently.

Algorithm 1 Assignment strategy for case 1

Require: The equal different matrix size, 2^u and 2^v; The number of the cores under multi-core platform, 2^k;
Ensure: The set of balanced point sets for the l th core, L_l, where $1 \leq l \leq 2^k$;
1: Arrange all the balanced point sets $\{E_{i,j}\}$ in a row-major order, $\{E_{i*2^v+j}\}$;
2: Calculate the number of the balanced point sets in L_l, $S = 2^u * 2^v/(4 * 2^l)$;
3: Merge the set of the balanced point set for L_l is $\{E_{S*l+1}, E_{S*l+2}, ...E_{S*(l+1)}\}$;
4: **return** L_l;

Case 2 Case 1 demonstrates when the core number is $r = \{2, 4, 2^k, ..., 2^p * 2^q/4\}$, the unbalanced calculation can be solved completely. When the core number r is not the power of 2, we can follow the same strategy as Algorithm (1) while handling the reminder carefully. We also arrange all the balanced point sets into a row-major order, unlike Algorithm (1) assigns continuous indexes for set L_l, where L_l represents the l th divided part for l th core, the assignment strategy changes to $L_l = \{E_l, E_{l+r}, ...\}$, where r is the number of the core. There are $(M*N)\%r$ sets with size $M*N//r+1$ and $r-(M*N)\%r$ sets with size $M*N//r$. The number of the items in L_l is not equal to each other, but the maximum difference is 1. Based our assignment strategy, we only need to consider the last $(M * N)\%r$ balanced point sets. In order to minimize the maximum difference among different L_l, these balanced point sets composed of 4 elements should be split to different cores. From the perspective of 2-D equal difference matrix, the origin is $(M/2, N/2)$, the assignment strategy mentioned above allocates the balanced point sets those are far from the origin first. As the assignment goes on, the variances of the balanced point sets are gradually lower. So that the

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93713-7_70

last $(M * N)\%r$ balanced point sets have relatively low variances. If $3r/4 < (M * N)\%r$, which means that there must be 4 elements in at least one core, the algorithm that keeps as the original does not slow down the performance. If $r/2 < (M * N)\%r \leq 3 * r/4$, there must be more than $s/4$ cores are empty when processing the last $(M*N)\%r$ balanced point sets. For each of the last $(M*N)\%r$ balanced point sets, namely, E_{i_1,j_1}, based on Eq. (17) we take out the element $b_{i_1,N-j_1+1}$, every three extracted elements form a set and the set is dispatched to a free process. Based on the low variance among the elements the calculation time nearly reduces to the 3/4 of the original. If $r/4 < (M * N)\%r \leq r/2$, these balanced point sets can be split into 2 parts, for set E_{i_1,j_1}, according to Eq. (17), one set is $\{b_{i_1,N-j_1+1}, b_{M-i+1,j}\}$, the other is $\{b_{i,j}, b_{M-i+1,N-j+1}\}$, each part contains 2 elements. For the last $(M * N)\%r$ balanced point sets the calculation time nearly halves. Similarly, if $(M * N)\%r \leq r/4$, these balanced point sets can be split into 4 parts, each element is assigned to different L_l. Algorithm (2) shows the detailed description.

Algorithm 2 Assignment strategy for case 2

Require: The equal different matrix size, M and N; The number of the cores under multi-core platform, r;
Ensure: The set of balanced point set for the lth core, L_l;
1: Arrange all the balanced point sets $\{E_{i,j}\}$ in a row-major order, $\{E_{i*N+j}\}$;
2: Calculate the remainder $(M * N)\%r$;
3: For the aliquot part, calculate the number of the balanced point set in L_l, $L_l = \{E_l, E_{l+r}, ...\}$;
4: **if** $r/2 < (M * N)\%r \leq 3r/4$ **then**
5: Split each balanced point set of E_{i_1,j_1} into 2 sets, take out the element $b_{i_1,N-j_1+1}$, every three extracted elements form new set.
6: Merge the split E_{i_1,j_1} to L_l;
7: **end if**
8: **if** $r/4 < (M * N)\%r \leq r/2$ **then**
9: Split each balanced point set of E_{i_1,j_1} into 2 sets, one set is $\{b_{i_1,N-j_1+1}, b_{M-i+1,j}\}$, the other is $\{b_{i,j}, b_{M-i+1,N-j+1}\}$
10: Merge the split to L_l;
11: **end if**
12: **if** $(M * N)\%r < r/4$ **then**
13: Split each balanced point set of E_{i_1,j_1} into 4 sets, each set contains 1 elements;
14: Merge the split E_{i_1,j_1} to L_l;
15: **end if**
16: **return** L_l;

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93713-7_70

4 Evaluation

4.1 Experimental Environment

Our cluster is composed of 8 homogenous nodes. The configuration of each node is shown in Table 1. We compare our algorithm with two other elementary assignment algorithms. One assigns the EM elements $[i * \frac{N*M}{r}, (i+1) * \frac{N*M}{r})$ to the i th core(we name it "i++"), where M, N is the grid width and height, r is the number of the cores. The other assigns the elements $\{i, i+r, ..., i+n*r\}$ to the i th core (We name it "i+r"), where $\{i+n*r | n \in N, i+n*r < M*N\}$. Acoording to Lin et al. [11], we set $M = N = 127$, the significand of $MPRF$ to 100-bit and the number of the Taylor expansion terms to 10.

We first compare the unbalanced calculation using the maximum elapse time difference of the $\{F_i\}, i = (1, 2, 3, 4)$ in eq. (9) among the three element assignment algorithms under a fixed P th-order Taylor expansion. Then we compare the elapsed time and the speedup ratio for the iterations of $\Psi_{i,m,n}$ and $\Theta_{i,m,n}$ in eq. (4).

Configuration	Setting
CPU	Intel Xeon E5-2699 v4@2.2GHz(2*22 cores)
Memory	128GB DDR4
Hard Disk	500GB SSD
MPI	Intel MPI Version 5.1.3
Network	InfiniBand
Operating System	CentOS 7.2 x86_64

Table 1. The configuration of the nodes.

4.2 Unbalanced Calculation

The maximum time difference between different cores to calculate $\{F_i\}, i = (1, 2, 3, 4)$ for different P th Taylor series can be used to quantitatively describe the unbalanced calculation. Fig. 2 and Fig. 3 show the comparison of the maximum time difference of all the cores among the three algorithms when we use 4 nodes of the cluster. For a fixed P th Taylor expansion, the maximum time difference of our algorithm is much smaller than "i++" and "i+r" no matter whether the core number is the power of 2 or not. In general, our algorithm is quite effective to reduce the unbalanced calculation.

4.3 Speedup Ratio

In addition, Fig.4 and Fig.5 show the comparison of the elapsed time of the calculation of $\Psi_{i,m,n}$ and $\Theta_{i,m,n}$ in eq. (4) in a iteration among the three algorithms.

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93713-7_70

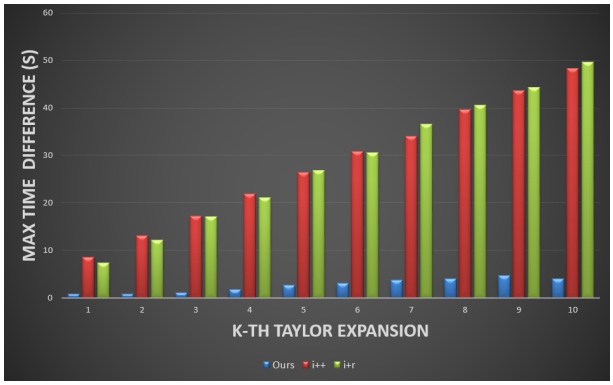

Fig. 2. The comparison of the maximum time difference among our algorithm, "i++" and "i+r". The core number is 128.

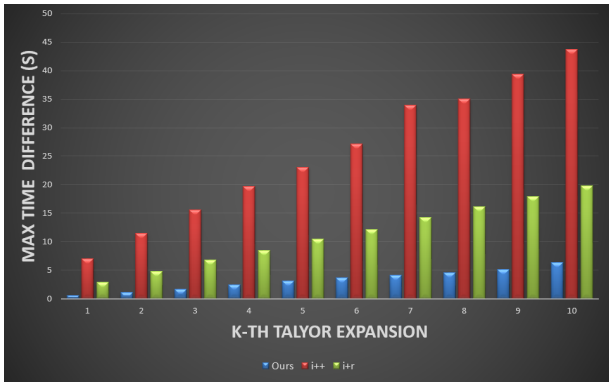

Fig. 3. The comparison of the maximum time difference among our algorithm, "i++" and "i+r". The core number is 176 which is the total CPU cores of 4 nodes.

For a fixed core number, the elapsed time of our algorithm is shorter than the other two algorithms due to the more balanced calculation of our algorithm and low cost for core assignment. Table. 2 gives the speedup ratios between our algorithm and "i++","i+r". In general, our algorithm can accelerate the calculation.

Core Number	32	44	64	88	128	176	256	352
Our algorithm VS "i++"	1.288 X	1.290 X	1.309 X	1.281 X	1.266 X	1.371 X	1.460 X	2.950 X
Our algorithm VS "i+r"	1.067 X	1.087 X	1.198 X	1.077 X	1.269 X	1.213 X	1.269 X	1.238 X

Table 2. The speedup ratios under different core numbers.

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93713-7_70

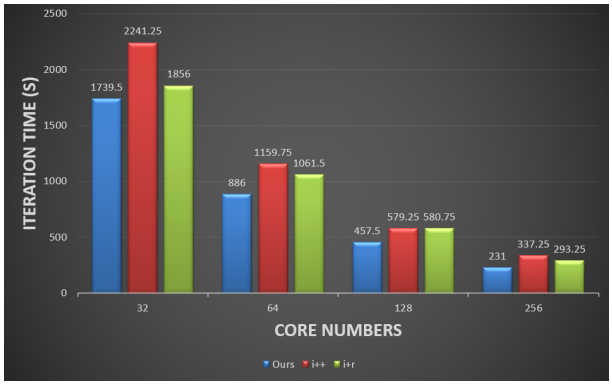

Fig. 4. The comparison of the elapsed time of an iteration among our algorithm, "i++" and "i+r". The core number is the power of 2.

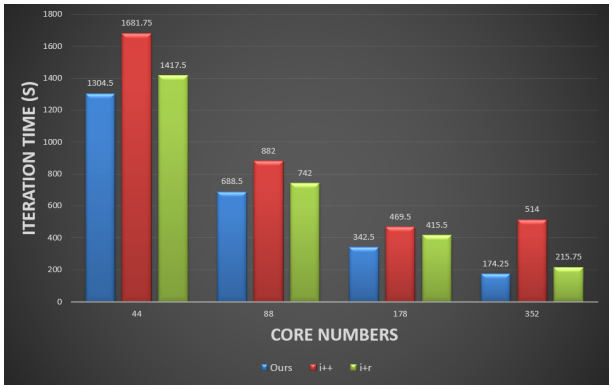

Fig. 5. The comparison of the elapsed time of an iteration among our algorithm, "i++" and "i+r". We use all the cores in the nodes.

5 Conclusion

In this paper, we propose an algorithm to optimize the unbalanced calculation for the CNS calculation of the two-dimensional Rayleigh-Bénard turbulence. We first establish the model of the unbalanced calculation. Then we introduce the notions of equal difference matrix and balance point set. Based on these notions, we introduce the algorithm to optimize the unbalanced calculation under multi-core platform. We prove our algorithm is optimal when the core number is the power of 2 and our algorithm approaches to the optimal when the core number is not the power of 2. Finally, we compare our algorithms with "i++" and "i+r" algorithms and demonstrate our algorithm is effective.

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93713-7_70

6 Appendix

6.1 Theorem 1

Given an arbitrary equal difference array, namely $\{a_n\} = \{a_1, a_2, ..., a_n\}$, the common difference is d. For $i, j \in N$ and $1 \leq i, j \leq n$. $a_{n-i+1} + a_i = a_{n-j+1} + a_j$. Use the general formula to expanse a_n:

$$a_n = a_1 + (n-1) * d \tag{18}$$

So a_{n-i+1} is:

$$a_{n-i+1} = a_1 + (n-i) * d \tag{19}$$

So $a_{n-i+1} + a_i$ is:

$$a_{n-i+1} + a_i = a_1 + (n-i) * d + a_1 + (i-1) * d = 2 * a_1 + (n-1) * d \tag{20}$$

The Eq. (20) is independent with i. Hence we have

$$a_{n-i+1} + a_i = a_{n-j+1} + a_j \tag{21}$$

where $i, j \in N$ and $1 \leq i, j \leq n$. From the perspective of the geometry, if we consider the equal difference array as a line segment, point $n - i + 1$ and point i is symmetrical about point $(n + 1)/2$, which is the center of the line segment. The Eq. (21) expresses that the sums of the two symmetrical points about the center are the same.

References

1. Avila, K., Moxey, D., de Lozar, A., Avila, M., Barkley, D., Hof, B.: The onset of turbulence in pipe flow. Science **333**(6039), 192–196 (2011)
2. Barrio, R., Blesa, F., Lara, M.: Vsvo formulation of the taylor method for the numerical solution of odes. Computers & mathematics with Applications **50**(1), 93–111 (2005)
3. Blackburn, H.M., Sherwin, S.: Formulation of a galerkin spectral element–fourier method for three-dimensional incompressible flows in cylindrical geometries. Journal of Computational Physics **197**(2), 759–778 (2004)
4. Fousse, L., Hanrot, G., Lefèvre, V., Pélissier, P., Zimmermann, P.: Mpfr: A multiple-precision binary floating-point library with correct rounding. ACM Transactions on Mathematical Software (TOMS) **33**(2), 13 (2007)
5. Getling, A.V.: Rayleigh-Bénard convection: structures and dynamics, vol. 11. World Scientific (1998)
6. Li, X., Liao, S.: On the stability of the three classes of newtonian three-body planar periodic orbits. Science China Physics, Mechanics & Astronomy **57**(11), 2121–2126 (2014)
7. Liao, S.: On the reliability of computed chaotic solutions of non-linear differential equations. Tellus A **61**(4), 550–564 (2009)

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93713-7_70

8. Liao, S.: On the numerical simulation of propagation of micro-level inherent uncertainty for chaotic dynamic systems. Chaos, Solitons & Fractals **47**, 1–12 (2013)
9. Liao, S., Li, X.: On the inherent self-excited macroscopic randomness of chaotic three-body systems. International Journal of Bifurcation and Chaos **25**(09), 1530023 (2015)
10. Liao, S., Wang, P.: On the mathematically reliable long-term simulation of chaotic solutions of lorenz equation in the interval [0, 10000]. Science China Physics, Mechanics and Astronomy **57**(2), 330–335 (2014)
11. Lin, Z., Wang, L., Liao, S.: On the origin of intrinsic randomness of rayleigh-bénard turbulence. SCIENCE CHINA Physics, Mechanics & Astronomy **60**(1), 014712 (2017)
12. Lorenz, E.N.: Deterministic nonperiodic flow. Journal of the atmospheric sciences **20**(2), 130–141 (1963)
13. Lorenz, E.N.: Computational periodicity as observed in a simple system. Tellus A **58**(5), 549–557 (2006)
14. Niemela, J., Sreenivasan, K.R.: Turbulent convection at high rayleigh numbers and aspect ratio 4. Journal of fluid mechanics **557**, 411–422 (2006)
15. Orszag, S.A.: Analytical theories of turbulence. Journal of Fluid Mechanics **41**(2), 363–386 (1970)
16. Pugachev, A.O., Ravikovich, Y.A., Savin, L.A.: Flow structure in a short chamber of a labyrinth seal with a backward-facing step. Computers & Fluids **114**, 39–47 (2015)
17. Rayleigh, L.: Lix. on convection currents in a horizontal layer of fluid, when the higher temperature is on the under side. The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science **32**(192), 529–546 (1916)
18. Roche, P.E., Castaing, B., Chabaud, B., Hébral, B.: Prandtl and rayleigh numbers dependences in rayleigh-bénard convection. EPL (Europhysics Letters) **58**(5), 693 (2002)
19. Saltzman, B.: Finite amplitude free convection as an initial value problemi. Journal of the Atmospheric Sciences **19**(4), 329–341 (1962)
20. Wang, L.P., Rosa, B.: A spurious evolution of turbulence originated from round-off error in pseudo-spectral simulation. Computers & Fluids **38**(10), 1943–1949 (2009)
21. Wang, P., Li, J., Li, Q.: Computational uncertainty and the application of a high-performance multiple precision scheme to obtaining the correct reference solution of lorenz equations. Numerical Algorithms **59**(1), 147–159 (2012)
22. Yang, D., Yang, H., Wang, L., Zhou, Y., Zhang, Z., Wang, R., Liu, Y.: Performance optimization of marine science and numerical modeling on hpc cluster. PloS one **12**(1), e0169130 (2017)
23. YAO, L.S., Hughes, D.: Comment on computational periodicity as observed in a simple system, by edward n. lorenz (2006a). Tellus A **60**(4), 803–805 (2008)
24. Zhang, L., Zhao, J., Jian-Ping, W.: Parallel computing of pop ocean model on quad-core intel xeon cluster. Computer Engineering and Applications **45**(5), 189–192 (2009)
25. Zhao, W., Song, Z., Qiao, F., Yin, X.: High efficient parallel numerical surface wave model based on an irregular quasi-rectangular domain decomposition scheme. Science China Earth Sciences **57**(8), 1869–1878 (2014)
26. Zhou, Q., Xia, K.Q.: Thermal boundary layer structure in turbulent rayleigh–bénard convection in a rectangular cell. Journal of Fluid Mechanics **721**, 199–224 (2013)

ICCS Camera Ready Version 2018
To cite this paper please use the final published version:
DOI: 10.1007/978-3-319-93713-7_70