Isomorphic New Parallel Division Methods and Parallel Algorithms for Giant Matrix Transpose

ZHOU Qi-hai
Information Technology Application Research Institute, Southwestern University Of Finance and Economics, Chengdu, Sichuan, China; School of Economic Information Engineering, Southwestern University Of Finance and Economics, Chengdu, Sichuan, China
Email: zhouqh@swufe.edu.cn

LI Yan
Information Technology Application Research Institute, Southwestern University Of Finance and Economics, Chengdu, Sichuan, China; School of Economic Information Engineering, Southwestern University Of Finance and Economics, Economics, Chengdu, Sichuan, China
Email: Liyan77@163.com

***Abstract*—In this paper, the weakness of the traditional matrix division methods for giant matrix transpose is pointed; Specific to the nature of the giant matrixes' transpose and based on the characteristics of asynchronous PRAM parallel computing model, some new division (such as booklet belt division, closed-loop division and Checkerboard-belt-shaped compound division) methods are proposed for improving the traditional existing matrix division methods, and their new parallel transpose algorithms for giant matrixes are advanced; The correspondence pressure between machines computational capability and enormously computing quantity in the parallel transpose process are reduced, while the parallel processing operating efficiency are enhanced, and the cost of parallel processing realization should be brought down by these new parallel algorithms (which could be realized on the cluster of workstations) based on division methods used fully the symmetrical characteristic of the giant matrix transpose operation.**

***Index Terms*—Isomorphic, Giant matrix transpose, Booklet belt division, Closed-loop division, Checkerboard-belt-shaped compound division, Asynchronous PRAM model, Cluster of workstations**

I. INTRODUCTION

Along with the economy and society's development, the amount of information rapidly growths which the people need to process. We are in an era of a knowledge explosion. Matrix tabulation is an effective and important tool used to describe and process information. But with the increasing amount of information, the matrixes need to deal with of which a considerable part are huge matrixes. The traditional approach can't be able to adequately satisfy us to the needs of rapidly dealing with complex information. Under this background, people propose a new idea -- parallel processing to deal with complex information. Literatures [1-6] have conducted the research separately from the different angle to the matrix transpose operation's parallelization processing question. Up to day, the existing study of large-scale parallel computing matrix transpose, almost all based on the traditional division method -- straight belt division or the checkerboard division. However, these two kinds of division method using for the matrix transpose operation have a common shortcoming: Have not noted the symmetrical characteristics of the matrix transpose operation, many elements need assign operations for exchange mutually in different region. So various processor need to correspond frequently among them in the parallel processing.

In the realization of the complex issues of parallel processing, how to discompose appropriately a complex question into some sub-questions is an extremely important and basic step. It has a direct bearing on the realization of parallel processing and parallel processing efficiency. In order to clear the stemming from this reason, the author has made the improvement to the traditional matrix division strategy, several new kinds of new division strategy in view of the matrix transposition operation were proposed in literature [7]. These new kinds of division strategy have ingeniously used the symmetrical characteristic of the matrix transpose operation. In these division methods, assign the elements needing to exchange mutually in the identical region. In parallel processing, it can effectively reduce the intensity of communication among the processors, and solve restrictions on parallel processing because of correspondence blocking. Now, the further research from the asynchronous PRAM parallel computation model angle to the giant matrix transpose operation parallelization processing question has been studied in this paper, and some parallel algorithms for the giant matrix transpose operation which realize on the cluster of workstations has constructed still.

II. PRAM MODEL SUMMARY [8]

© 2010 ACADEMY PUBLISHER
doi:10.4304/jcp.5.2.169-177

A. Describing the PRAM Model

PRAM (Parallel Random Access Machine) model, namely parallel random access machine, also called the shared memory of the SIMD model which is one kind of abstract parallel computing model. In this kind of model, assuming there is a shared buffer memory of which the capacity is infinitely great. The number of processors is limit or unlimit. Those processors have the same function and can take the simple arithmetic operation and the logical judge. Exchange the data mutually in any time in various processors through the sharing memory cell. The PRAM model's merit is: It is particularly suited to the expression of parallel algorithms, analysis and comparison; simple to use, many such as: communication between the processor, memory management and synchronization process, such as low-level details of the parallel computer are hidden in the model; easy-to-design Algorithms and can be slightly modified to run on different machines in parallel; also has the possibility to join the question in the PRAM model such as synchronization and communication. Therefore, this model has been widely used in the algorithm sector.

B. Asynchronous PRAM Model (APRAM)

Model characteristic: Phase PRAM model is an asynchronous PRAM model, simply record it as APRAM, is composed by P processor, its characteristic is that each processor has all its bureau to save, the partial clock and the partial procedure; Correspondence process between the processor shares the global memory; the non-overall situation clock, various processors asynchronous carry out respective instruction independently; any time dependence between the processor need to rely on the synchronization Barrier joined in various processors' procedure; An instruction can complete in a non-identified (unbounded) but limited period of time.

In APRAM model instruction type: In the APRAM model has four kinds of instructions:

1) The overall situation reads, reads the overall situation memory cell's in content to save in the person bureau;

2) The partial operation, playing execution operation to the data saved in the bureau, its result is saved in the person bureau;

	Processor 1	Processor 2	…	Processor p
Phase1	Read x_1 Read x_2 *	Read x_3 * Write to B		Read x_n * *
synchronization Barrier	Write to A	Write to C		Write to D
Phase2	Read B *	Read A *		Read C *
synchronization Barrier	Write to B	Write to D		
Phase3	* Read D	Read C		write to B Read A *
synchronization Barrier	*			Write to B

Figure 1. Asynchronous computation in APRAM mold with Barriers (where * expresses partial operation)

3) The overall situation writes, read the content saved in the person bureau in the overall situation memory cell;

4) The synchronization, the synchronization is a logical spot in computation. In this spot, various processors must wait for other processor, and then can continue to carry out its local program (shown as Fig. 1).

In this paper, constructing parallel algorithm for the matrix transpose operation is based on the symmetrical booklet belt division method and the closed-loop division method. These two kinds of division methods have considered fully the matrix transpose operation's symmetrical characteristic. Put the elements which are symmetrical about the principal diagonal in the identical processor, because we will only need to exchange those elements in the matrix transpose operation. Therefore carries on the algorithm design according to these two kinds of division method, the merit is: In the processor parallel processing, various processors do not need to exchange data with each other. Therefore the algorithm constructed in this paper does not need to set up the synchronization Barrier. It can avoid the disruption in the parallel processing process, because various processors don't wait for the correspondence (as shown in Fig. 2).

Processor 1	Processor 2	…	Processor P
Read A1	Read A2		Read AP
*	*		*
*	*		*
Write to A1	Write to A2		Write to AP

Figure 2. Asynchronous computation in APRAM mold without Barrier (where * expression partial operation)

Ⅲ. THE DIVISION METHOD FORGIANT MATRIX

A. Two Typical Traditional Division Methods

Both band division method and chess-board division method are two typical traditional division methods in matrix parallel processing.

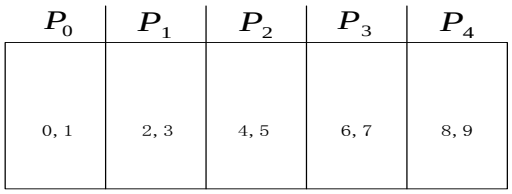
(1) Band division method

The characteristic of band division method is: dividing the given giant matrix into some vertical (or level) bands including some whole entire row or some whole entire columns. Each of them will be assigned to the different processor. These columns (or rows) can be continuous or isometric phases. The former called as continuous block strip, the latter called as cycle band division, as shown in Fig. 3 (where P_x represents processor x, "i, j" represents separately the row number and column number of the element a_{ij} of the given matrix).

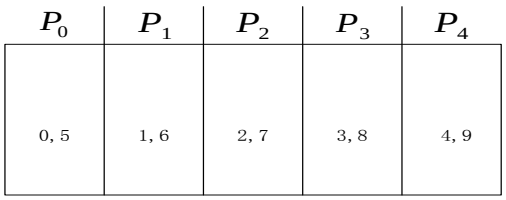
(2) Chess-board division method

The characteristic of chess-board division method is: the given giant matrix is divided into a number of sub-matrixes. Each of them is assigned to one different processor. Each of processors has non-complete rows or columns (as shown in Fig. 4).

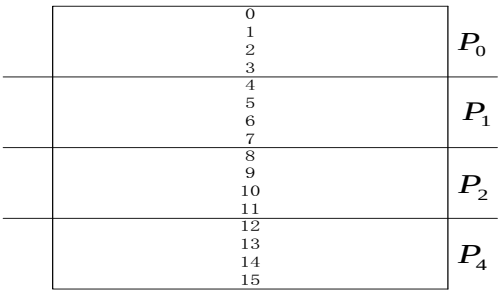
© 2010 ACADEMY PUBLISHER



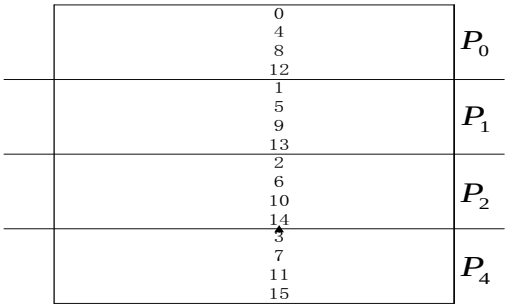
A) Continuous vertical band division



B) Cycle vertical band division



C) Continuous level band division



D) Cycle level band division

Figure 3. The band division for a 16×16 matrix

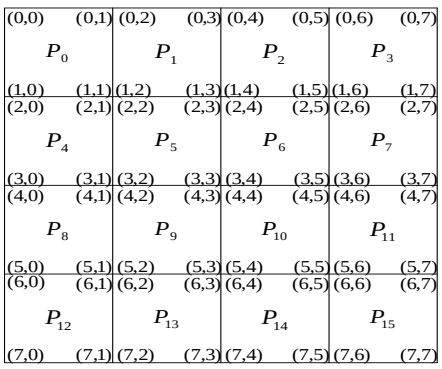


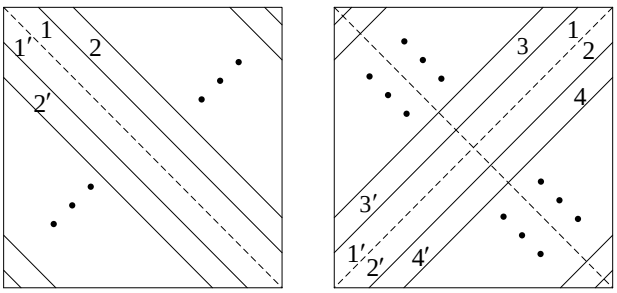
Figure 4. A case of chessboard division for a matrix

B. New Methods for Dividing Giant Matrix

We can obtain the transposed matrix A^T of the given matrix A by exchanging the element a_{ij} and a_{ji} in the given matrix A (where i=1,2,3, …, n; j=1,2,3, …, n). The above viewpoint is merely from the partial angle to understand the relationship between matrix A and its transposed matrix A^T. The traditional division methods are precisely based on this traditional viewpoint. In fact, we can view to use the relationship between matrix A and its transposed matrix A^T from a broader vision. Two important isomorphic natures should be found at least: 1) The row i of A is just the column i of A^T, (where i=1,2,3,…,n); 2) As for the division form for dividing the elements of a matrix, we can conduct the research from the different angle of view. So, using the isomorphic natures about the relationship between matrix A and its transposed matrix A^T, some isomorphic new methods of parallel computing division for special large numerical matrixes' transposing are created in this paper as follows.

(1) Oblique bell division methods

The elements of matrix A can be distributed into some oblique belts which are parallel with the main (or vice) diagonal line (as shown in Fig. 5). So A^T can be obtained by exchanging the elements in two oblique belts which are symmetric about the main (or vice) diagonal line. Based on this new isomorphic idea, oblique belt divisions are given as following.



A) Right oblique belt division B) Left oblique belt division

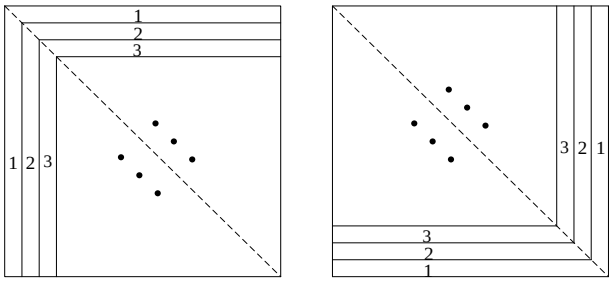
Figure 5. Oblique belt division of a matrix

The oblique belt division contains two different division methods according to its oblique direction: the right oblique belt division method and the left oblique belt division method. A large matrix could be divided into some symmetrical oblique belts (named as $Belt_i$) which are parallel with the main (or vice) diagonal line of the given matrix (as shown in Fig. 5). In the right oblique belt division method, the oblique $Belt_i$ contains two symmetrical oblique belts (i. e. $belt_i$ and $belt_{i'}$) which are parallel with the main diagonal line. One locate the right of the main diagonal line, the other locate the left of the main diagonal line. The bandwidth of the two oblique belts must be equal. But the bandwidth of $Belt_i$ is not necessary to equal with the bandwidth of $Belt_j$ (where $i \neq j$). In the left oblique belt division method, one oblique belt contains two parts which are symmetrical about the main diagonal line. One locate the right of the main diagonal line, the other locate the left of the main diagonal line. The bandwidth of the two oblique belts must be equal. But the bandwidth of $Belt_i$ is not necessary to equal with the bandwidth of $Belt_j$ (where $i \neq j$). However, in order to balance each processor's workload, the following tactics for oblique belt division should be used:

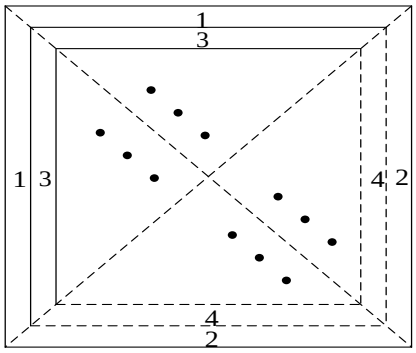
1) Oblique belt division with equal bandwidth

© 2010 ACADEMY PUBLISHER

If the bandwidth of all oblique belts is equal to each other, then every processor could save and process one symmetric oblique belts group at the same time, which is composed of four belts: $belt_i$, $belt'_i$, $belt_{p-i+1}$ and $belt'_{p-i+1}$ (where $belt_i$ and $belt'_i$ are symmetric about the main diagonal line, $belt_{p-i+1}$ and $belt'_{p-i+1}$ are symmetric too). When carries on the transpose operation to a matrix, the symmetrical elements about the main diagonal line must be carried on exchanging mutually. The asynchronous parallel assigned operations for exchanging of each group symmetrical elements included in the $belt_i$, $belt'_i$, $belt_{p-i+1}$ and $belt'_{p-i+1}$ should be planed in the same processor. Therefore in parallel processing matrix transpose operation process, only need to carry on asynchronous parallel exchanging the symmetric elements in each processor's interior, but does not need to exchange the data between various processors. So the enormous correspondence costs of exchanging the data between various processors can be reduced greatly. In addition, in order to balance the various processors' workload, take the following allocation strategy: Long belts and short belts combined as one group, of which the elements' total equals and matches to another group mutually and approximately, should be set in the same one processor for exchanging assignment.



A) Band-length-reducing type B) Band-length-raising type



C) Triangle-symmetric-band type

Figure 6. Right angle symmetrical belt division of a matrix

2) Oblique belts division with unequal bandwidth

i) Two mutual symmetrical oblique belts have the same bandwidth, but there are some asymmetrical oblique belts of which bandwidth are not need to equal. However the total of elements included in each oblique belt group is approximately equal. Then each processor could save and process two symmetrical belts' group at the same time; ii) Of course, if all oblique belt's bandwidth is not equal, but the number of elements included in each equal distance-total (which is constant equal to p+1) belt group is approximately equal, which is composed of $belt_i$ and $belt_{p-i+1}$ (or its symmetrical equal distance-total belt group consisted of $belt'_i$ and $belt'_{p-i+1}$), then processor P_i could save and process four belts (i.e. $belt_i$, $belt'_i$, $belt_{p-i+1}$ and $belt'_{p-i+1}$) without exchanging date with other processor and communicating among many processors.

Therefore, the new oblique belts division method could improve the efficiency of the giant matrix transpose operation, and will be " the bigger, the more".

(2) Right angle bell division methods

Right angle belt division includes three types about the shape and direction shown as Fig. 6:

Type 1: Band-length-reducing right angle belt division (which means: The longer length of belt is, the more the number of belt is); Type 2: Band-length-raising right angle belt division (which means: The shorter the length of belt is, the less the number of belt is); Type 3: Triangle-symmetric-band-length-raising right angle belt division (which means: It is the combination of the halves from the above two types).

In the three type of right angle belt division, the bandwidth satisfies the following condition: $1 \le bandwidth \le k < n$. It is obvious that all elements locating on the both sides of one right angle belt should be assigned to the same processor for exchanging. In order to balance each processor workload, should introduce the following strategy in right-angle belt division method:

1) Right angle belt division with equal bandwidth

If all right angle belts' bandwidth is equal, then each processor could save and process a distance-total right angle belts group at the same time, which is composed of two belts: $belt_i$ and $belt_{p-i+1}$ (the total of the distances from $belt_i$ and $belt_{p-i+1}$ to the high end of the main diagonal line is constant equal to p+1). Therefore in parallel processing matrix transpose operation process, only need to carry on exchanging the symmetric elements in each processor's interior, but does not need to exchange the data between various processors. So it can reduce correspondence costs enormously.

2) Right angle belts division with unequal bandwidth

i. If some right angle belts' bandwidth is not equal, but the number of elements included in each right angle belt is approximately equal, then each processor can save and process a right angle belt at the same time; ii. If the bandwidth of all right angle belts are not equal, but the number of elements included in each equal distance-total belt group consisted of $belt_i$ and $belt_{p-i+1}$ is approximately equal, then processor P_i could save and process the elements included in two right angle belts (i.e. $belt_i$ and $belt_{p-i+1}$) without exchanging date with other processor.

So the new right angle belts division method could improve the efficiency of the giant matrix transpose operation further.

(3) Lap belt division method

© 2010 ACADEMY PUBLISHER

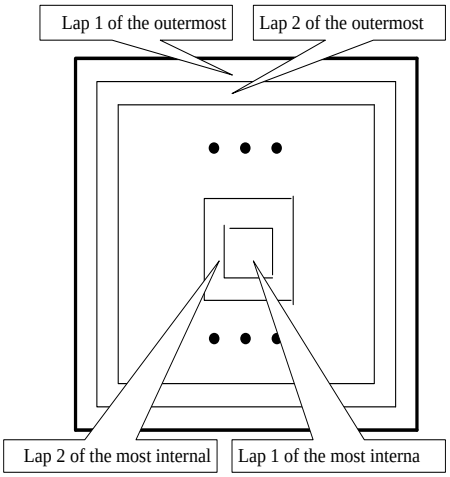


Figure 7. Lap belt division of a matrix

Consider the special and interesting fact: it is the most natural that all elements which are placed on the same lap form the outermost to the most internal of a given matrix should be divided into the same group, shown as Fig. 7.

In general speaking, all the laps of a given matrix could be named as lap_1, lap_2, lap_3,…, lap_m (where $m=[(n+1)/2]\leq p$) form outside to inside.

Lap belt division with equal bandwidth

If the bandwidth of all lap belts is equal, then processor P_i could save and process one equal distance-total lap belts group which is composed of two lap belts: lap $belt_i$ and lap $belt_{p-i+1}$ (for the total of the distances from lap $belt_i$ and lap $belt_{p-i+1}$ to the most internal of the center lap belt (Notice: the center lap belt is just the most internal lap belt) is constant equal to [(n+1)/2]). It is clear that the elements included in lap $belt_i$ and lap $belt_{p-i+1}$ are assigned to a processor for parallel processing. When these elements are exchanged in the matrix transpose operation, only need to carry on exchanging the symmetric elements in each processor's interior, but does not need to exchange the data between various processors. So it can reduce correspondence costs enormously.

Lap belts division with unequal bandwidth that:

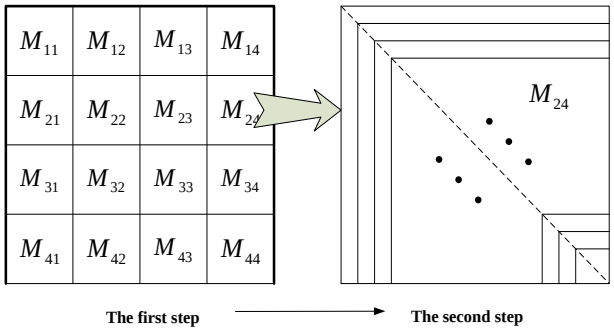


Figure 8. The checkerboard right-angle belt compound division

1) If some lap belts' bandwidth is not equal, but the number of elements included in each lap belt is approximately equal, then each processor can parallel save and process a lap belt at the same time.

2) If the bandwidth of all lap belts are not equal, but the number of elements included in each equal distance-total belt group consisted of lap $belt_i$ and lap $belt_{p-i+1}$ is approximately equal, then processor P_i could save and process the elements included in two lap belts (i.e. lap $belt_i$ and lap $belt_{p-i+1}$) without exchanging date with other processors.

Therefore, the new lap belts division method could improve the efficiency of the matrix transpose operation.

(4) Checkerboard-belt compound division

Under checkerboard division, although correspondence intensity between various processors is high, this division method's merit is: In the division initial stage, the matrix's elements are divided into some blocks, so the matrix division operation is easier to achieve. Speaking of the belt-shaped division, the situation is opposite with it. The belt-shaped division can reduce correspondence pressure between various processors enormously, but the division operation is difficult to realize. Therefore this fact inspires us to compose both checkerboard division and the belt-shaped division into one new matrix division method—checkerboard belt-shaped compound division. This merit of this new type division is: on the one hand, it can reduce correspondence pressure between various processors enormously; on the other hand, it is easy to realize. The steps for realizing this division as follows: Step 1: Carry on the division according to the checkerboard division method to the matrix; Step 2: Carry on the division according to the belt-shaped division to the sub-block (i. e. every smaller checkerboard-block) obtained in Step 1.

The checkerboard belt-shaped compound division process can be shown in Fig. 8.

Ⅳ. CONSTRUCTING PARALELL ALGORITHM

A. Reviewing the Parallel Algorithm Based on Traditional Division Methods

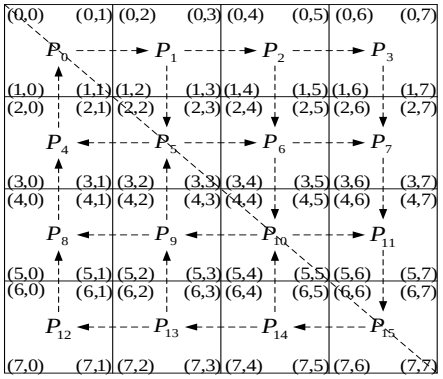
(1) Reviewing the parallel algorithm for matrix transpose operation based on the checkerboard division method

Construct the parallel algorithm for matrix transpose operation based on the checkerboard division method according the following three steps: Step 1: Carry on the iterative division to the matrix. Firstly, matrix is divided into 4 small sub-matrixes; then separately divide those small matrixes into 4^2 smaller sub-matrixes, …, until the original matrix is divided 4^k=P (P is the number of processors) smaller sub-matrixes as correspond sub-blocks; Step 2: Regarding those sub-blocks as element, carry on the transposition operation to them; Step 3: Carry on the transposition in each sub-block's interior. Realize the step 2 and step 3 can be shown as Fig. 9.

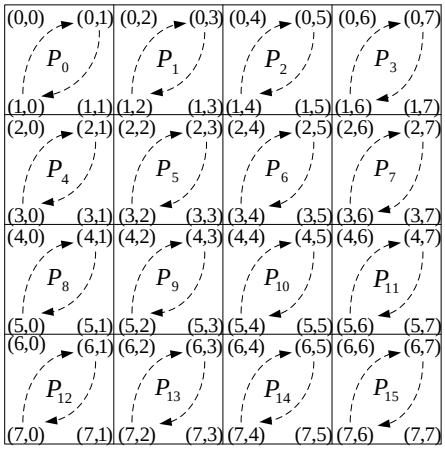
The shortcoming of this parallel algorithm for matrix transpose operation based on the chessboard division method: In the early, because of the relatively small number of matrix the load of communication lines is few and low. But when the computing is going on, while the problems to be computed are more and more complicated, the number of sub-matrix increases rapidly, so that the

© 2010 ACADEMY PUBLISHER

load of communication lines will be growing more rapidly.



A) Sub-block transposition



B) Sub-block interior transposition

Figure 9. The 8×8 matrix transposition based on the checkerboard division

The shortcoming of this parallel algorithm for matrix transpose operation based on the chessboard division method: In the early, because of the relatively small number of matrix the load of communication lines is few and low. But when the computing is going on, while the problems to be computed are more and more complicated, the number of sub-matrix increases rapidly, so that the load of communication lines will be growing more rapidly.

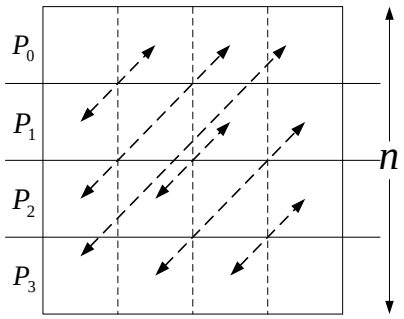


Figure 10. 4 processors, 4×4 matrix transposition

(2) Reviewing the parallel algorithm for matrix transpose operation based on the straight belt-shaped division method

As shown in Fig. 10, regarding a matrix $A_{n\times n}$, using n processors to carry on transpose operation to $A_{n\times n}$, based on lever band division.

At the beginning, the matrix's element (i,0), (i,1), …, (i,n-1) are located in processor P_i, where (i,j) is used to represent element $a_{i,j}$. Then after the transposition, the element (i,0) is located in processor P_0, the element (i,1) is located in processor P_1, and so on. In the transposition process, each processor will transmit the different value to other processors. In fact, it is many to many individual correspondences.

The shortcoming of this parallel algorithm for matrix transpose operation based on the band division method: The matrix transpose algorithm based on the strip division requires each processor to communicate with other processors. That requires high capacity of communication lines between processors. Some processors may sometimes suspend because the communication lines are blocking, so that affect the normal operation of processors, and processors can not fully play to their potential.

B. Constructing Parallel Aalgorithms Based on The New Division Methods

(1) Constructing parallel algorithm based on Right angle symmetrical belt division

Taking the division method shown in Fig.6A as an example, explain how to design a parallel algorithm for the matrix transpose operation based on Right angle symmetrical belt division. Assuming the number of rows of the given matrix is "m", the number of columns of the given matrix is "n" where suppose m=n for indicating conveniently (when m≠n, the processing method is similar to m=n), the number of processors is "P", e=[n/P].

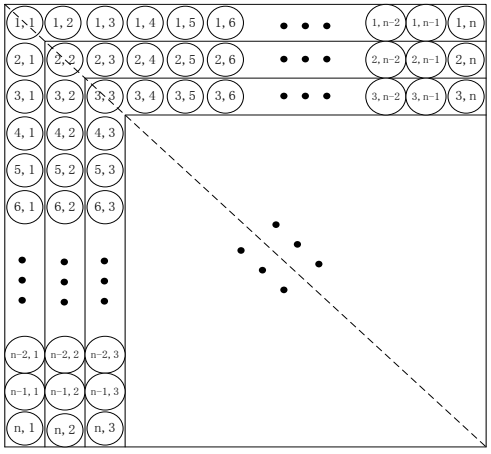


Figure 11. Matrix's dividing result of right angle symmetrical belt division

Carry on the division operation to the matrix according the right angle symmetrical belt division method. In given matrix A, assign every element $a_{i,j}$ of k^{th} to n^{th} in the column "k" and its corresponding element of $k+1^{th}$ to the n^{th} in the row "k" to the belt "k". Matrix's dividing result

© 2010 ACADEMY PUBLISHER

of right angle symmetrical belt division can be shown as Fig.11 (where element $a_{i,j}$ is described simply as "i,j" for convenience, i<j; by analogy to other following figures).

In order to broadcaste belt "k" to workstation$_j$ (where k=0,1, …, p-1) and to balance the workload of the workstation, assign the elements included in the fold belt$_{i\times p+j}$, (where i=0,1, …, e-1) to workstation$_j$ (where j=0,1, …, p-1), and separately put the elements into the same array-shared. Finally put the remaining elements into the workstation "p-1". Carry on the parallel processing in p workstations, each workstation separately carries on the transpose operation to the elements which receives from the main engine. After the parallel operation finished, return the result according to the input order to the main engine, as shown in Fig.12:

The cycle of displacement operation in various workstations: In the matrix division stage, the symmetrical booklet belt division already fully uses the symmetrical characteristic of matrix transpose operation. Therefore, in every workstation, we only need to carry on a number of the cycle of displacement operation to the corresponding arrays. Then complete the transposition operation to the input elements.

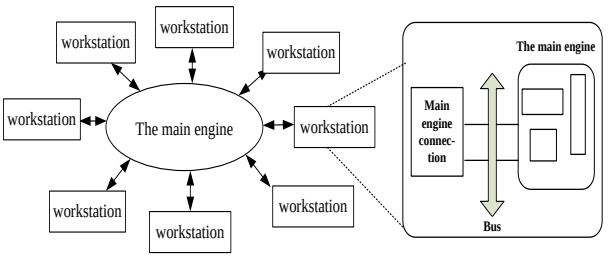


Figure 12. Cluster of workstations (COW)

Below take one of input array (a_{11}, a_{12}, a_{13}, …, a_{1n}, a_{21}, a_{31}, …, a_{n1}) in the first workstation as an example, explain how to realize the cycle of displacement operation in each workstation (shown as Fig. 13 to 14).

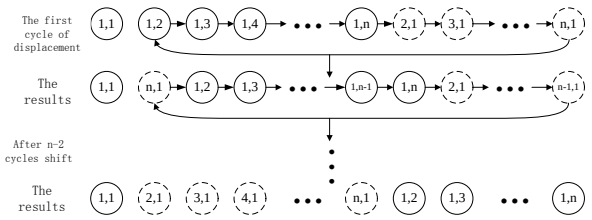


Figure 13. The cycle of displacement operation

The exchange operation process among the matrix's elements is expressed and shown as Fig. 14.

So, the parallel algorithm based on right angle symmetrical belt division can be given as following:

Algorithm Eg01; {Matrix transpose parallel algorithm based on right angle symmetrical belt division}

Input: goal matrix;

Output: transposed matrix;

>>> {Starting}

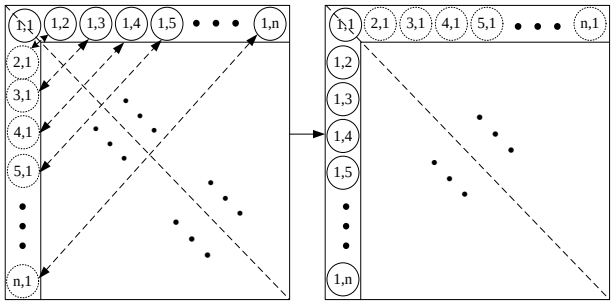


Figure 14. The exchange operation process among the matrix's elements

Step 1: Carry on the division operation to the input matrix according to the right angle symmetrical belt division method which is introduced in the preceding text; {Through this operation, the elements of the input matrix are divided into a series of Symmetrical belt regions}

Step 2: For j=0 to p-1 do {The division result which obtains in Step1 will be broadcasted to p workstations}

\\ {Beginning of the multi-operations' loop-body}

Send the elements in the fold belts$_{i\times p+j}$ (where i=0,1, …, e-1) to workstation p$_j$;

Receive the elements in the fold belts$_{i\times p+j}$ (where i=0,1, …, e-1) from the main engine

// {Ending of the multi-operations' loop-body}

Step 3: For j=0 to p-1 par-do {Carry on the parallel processing in p workstations}

Carry on the the cycle of displacement operation in each workstation;

Step 4: Broadcasts the workstation's processing result to the main engine according to the input order.

!!! {Finished}

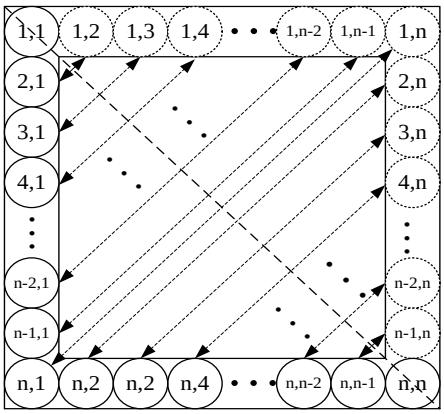


Fig. 15: Transpose operation based on lap (i. e. closed loop) belt-shaped division

Comparing with the right angle fold belt division method, the closed-loop belt-shaped division's improvement is: It can well balance various processors' work load. Along with the matrix scale expansion, this merit is more obvious. But speaking of the design parallel algorithm, algorithmic design process is similar. No longer give unnecessary detail of it here. Only give the Fig. 15 for indicating the basic philosophy of the

© 2010 ACADEMY PUBLISHER

transposition operation under the lap (i. e. closed loop) belt-shaped division.

(2) Constructing parallel algorithm based on checkerboard belt-shaped compound division method

K-levers model of multi-storey nested Cluster of workstations

In the ordinary cluster of workstations parallel computing system, there are only one main engine and the many workstations which are at the same position. Because of ordinary cluster of workstation group's this characteristic, using this parallel computing to process complex question has the following flaw: Can only carry on division operation to the complex question one time, when the question has the multi-storey structure, the ordinary cluster of workstations displays much limitation. Based on this reason, carry on the expansion to the ordinary cluster of workstations and propose one kind of new parallel computing system -- K-levers model of multi-storey nested Cluster of workstations advanced in this article. This new parallel computing system's structure can be shown as Fig.16.

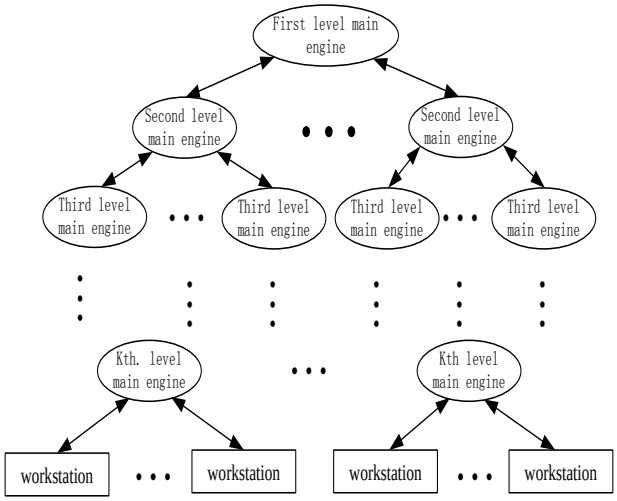


Figure 16. K-levers model of multi-storey nested Cluster of workstations

Constructing parallel algorithm

Constructing parallel algorithm based on the three - levers tree multi-storey nested Cluster of workstations, the process is:

1) Divide the given matrix into some sub-blocks in the first-level main engine according to the checkerboard division method, and broadcast separately various sub-blocks to the second-level main engines;

2) Carry on the belt-shaped division to the sub-blocks which receive from the first-level main engine in each second-level main engine, and broadcast separately the result to the govern workstations;

3) Respectively return the result processed in the workstations to the second-level main engines, and carry on the correspondence between second-level main engines to exchange the date which receive from the govern workstations;

4) Return the exchanging result from the second-level main engines to the first-level main engine. Hence the matrix transpose operation finished.

Involve three correspondences in the above parallel processing's process, respectively is: The first-level main engine $\Longleftrightarrow$ second-level main engines; the second-level main engine $\Longleftrightarrow$ the govern workstations; the second-level main engines $\Longleftrightarrow$ the third-level main engines.

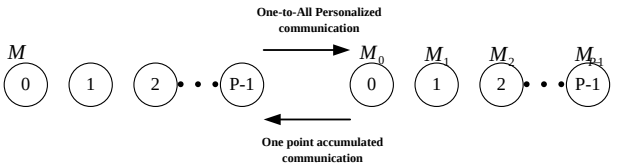


Figure 17. One-to-All Personalized and one point accumulated communication model

It is "the first-level main engine $\Longleftrightarrow$ the second-level main engines; the second-level main engin $\Longleftrightarrow$ ethe govern workstations" that means of communication model is the One-to-All Personalized and One point accumulated communication (shown as Fig .17).

The correspondence of "the second-level main engines $\Longleftrightarrow$ the second-level main engines" should follow the principle: exchanging the symmetrical sub-blocks. Take the 4×4 checkerboard division as an example to explain this exchanging principle. As shown in Fig .18:

Algorithm Eg02; {Matrix transpose parallel algorithm based on checkerboard belt-shaped compound division method}

Input: Goal Matrix;

Output: Transposed matrix;

>>> {Starting}

Step 1: Divide the input matrix into some sub-blocks in the first-level main engine according to the checkerboard division method;

Step 2: For j=0 to p-1 do {Broadcast the division result which obtains in the Step1 to the p second-level main engines}

\\ {Beginning of the multi-operations' loop-body}

Send sub-block j to second-level main engine p_j;

Receive sub-block$_j$ from the first-level main engine

// {Ending of the multi-operations' loop-body}

Step 3: For j=0 to p-1 par-do {Carry on the parallel processing in p second-level main engines}

Call Algorithm eg01 as sub-algorithm; {where Algorithm eg01 is use as sub-algorithm in this Algorithm eg02}

Step4: Respectively return the result processed in the workstations to the second-level main engines;

Step5: Exchange the information between p second-level main engines following the principle: exchanging the symmetrical sub-blocks;

Step6: Return the exchanging result from the second-level main engines to the first-level main engine according to the input order.

!!! {Finished}

© 2010 ACADEMY PUBLISHER

Ⅴ. CONCLUSION

The matrix transpose is in the signal processing a very important research area. Now, on the one hand, the request for processing speed becomes more and more high. Even requests the realization real-time processing; on the other hand, the information becomes more and more complex. Inevitable must draw support from the parallel processing technology regarding complex information's processing. Before construct parallel algorithm for matrix transpose operation, must solve the following question: How to carry on the division to the matrix. In this paper, firstly, make the improvement to the existing matrix division method and introduce the symmetrical booklet belt division, closed-loop division and Checkerboard -- belt-shaped compound division. Then based on these kinds of division method, construct some parallel algorithms for the matrix transpose operation, which realize on the cluster of workstations.

REFERENCES

[1] Jian Fan, Liu Guangping, Zhou Zhiming. Distributed Matrix Transpose on Multicomputer [J].MICROPROCESSORS, 2002,2(5):34-37.

[2] Mong Xiangjie, Zhan Li-lun, Zen Yonghong. Research of Parallel Matrix Transposition Algorithms in Distributed Memory Computing [J].COMPUTER ENGINEERING & SCIENCE, 1999,21(5):67-71.

[3] Zhu Qingyun, Huang Xingming, Zhao Ling. A Fast Computer Algorithm for Matrix Transposing [J].Journal of National University of Defense Technology, 1990,12(3):76-78.

[4] Xie Yingke, Zhang Tao, Hang Chengde. Design and Implementation of Matrix Transposition Unit for RealTime SAR Image Systems [J].JOURNAL OF COMPU TER RESEARCH AND DEVELOPMENT, 2003,40(1):6-11.

[5] Wu Fei. The sparse matrix transpose operation [J].Computer Engineering & Science, 1989(3):85-92.

[6] Xu Li.Matrix transpose algorithm the best choice [J].Journal of Chizhou Teachers College, 2001(03):9-10.

[7] Zhou Qihai, Huang Tao, Li Yan, Wang Honglei. Isomorphic New Parallel Division Methods for Special Large Numerical Matrix Transpose [P]. DCABES 2008 Proceedings,2008(Ⅰ):105-109.

[8] Cheng Guoliang. Parallel computing -- structure programming algorithm [M]. Beijing: Higher Education Press,2002: 109-111.

Zhou Qihai (1947-) is a Full Professor (from 1995), Doctor's (and Master's) tutor and a head of Information Technology Application Research Institute, School of Economic Information Engineering, Southwestern University of Finance and Economics (SWUFE), China. He graduated in 1982 from Lanzhou University, China; has been working in SWUFE since 1982, successively hold posts from teaching assistant (1982-1987), lecturer (1987-1991), vice professor (1991-1995, promoted anomaly in 1991), professor (1995-today, promoted anomaly in 1995); and got the titles of both "Outstanding experts (enjoyed government subsidies) with outstanding contributions of Sichuan province, China" (summa cum laude of Sichuan province government, 1996) and "One hundred academic and managerial leading heads of China informationalization" (summa cum laude about this domain in China, 2006). He has published 46 academic books and over 212 academic papers; and is President of **IITAA** (**I**nternational **I**nformation **T**echnology & **A**pplications **A**ssociation), Chair or Organizing Chair of some important international conferences. His research interests are in algorithm research, computational geometry, isomorphic information processing, economics & management computation, eBusiness, and so on. More (in Chinese) about Prof. Zhou Qihai is shown here:

LI Yan (1983-) is studying in School of Economic Information Engineering, Southwestern University Of Finance and Economics，Chengdu, Sichuan, China. He has published 13 papers. His research areas are in Non-definite decision-making, intelligence information processing.

© 2010 ACADEMY PUBLISHER