Springer

Journal of Mechanical Science and Technology 23 (2009) 3169~3178

Journal of
Mechanical
Science and
Technology

www.springerlink.com/content/1738-494x
DOI 10.1007/s12206-009-1001-4

# Parallel implementation of finite volume based method for isoelectric focusing †

Jaesool Shim[1,*], Prashanta Dutta[2] and Cornelius F. Ivory[3]

[1]*School of Mechanical Engineering Yeungnam University, Kyeongsan, Kyongbuk, Korea, 712-749*
[2]*School of Mechanical and Materials Engineering Washington State University, Pullman, WA, USA 99164-2920*
[3]*School of Chemical Engineering and Bioengineering Washington State University, Pullman, WA, USA 99164-2710*

(Manuscript Received November 26, 2008; Revised June 19, 2009; Accepted September16, 2009)

**Abstract**

A message passing interface (MPI) based parallel simulation algorithm is developed to simulate protein behavior in non-linear isoelectric focusing (IEF). The mathematical model of IEF is formulated based on mass conservation, charge conservation, ionic dissociation-association relations of amphoteric molecules and the electroneutrality condition. First, the concept of parallelism is described for isoelectric focusing, and the isoelectric focusing model is implemented for 96 components: 94 ampholytes and 2 proteins. The parallelisms were implemented for two equations (mass conservation equation and electroneutrality equation). The CPU times are presented according to the increase of the number of processors (1, 2, 4 and 8 nodes). The maximum reduction of CPU time was achieved when four CPUs were employed, regardless of the input components in isoelectric focusing. The speed enhancement was defined for comparison of parallel efficiency. Computational speed was enhanced by maximum of 2.46 times when four CPUs were used with 96 components in isoelectric focusing.

***Keywords***: A message passing interface (MPI); Finite volume method (FVM); Proteins; IEF

## 1. Introduction

Various diseases of the human body are related to protein malfunctions such as protein kinases or phosphoralization [1]. Early research has shown that an acute phase protein mutation in the plasma and cerebrospinal fluid is responsible for various diseases, including Alzheimer's, schizophrenia, and major depression [2]. Hence, for early detection of these diseases, it is very important to separate and preconcentrate the responsible proteins. Generally, 2D-PAGE is used to separate these proteins, where IEF is used as the first separation dimension because of its high resolution capacity. Recently, researchers have used a microfluidic platform to reduce the separation time from days to minutes by using liquid phase ampholyte-based isoelectric focusing (IEF). Some of the other advantages of ampholyte-based microchannel IEF are low reagent consumption, low thermal band dispersion, low sample volume requirement etc. [3]

Using IEF, proteins can be separated in the presence of pH and electric potential gradients in the electrophoretic system [4]. Carrier ampholytes, which are low molecular weight amphoteric molecules, are used to form the pH profile for protein separation. In the design of microchips, numerical simulations of IEF are used to find the optimal design for high performance of protein separation.

The mathematical models of free solution IEF are based on the mass conservation equations of each ionic component, charge conservation and the electroneutrality condition in the system [5]. In IEF, each

† This paper was recommended for publication in revised form by Associate Editor Chang-Wan Kim

*Corresponding author. Tel.: +82 53 810 2465, Fax.: +82 53 810 2465
E-mail address: jshim@ynu.ac.kr

component (both protein and ampholyte) consists of a number of species. For instance, a protein might have ten to a hundred charge states or species depending on the pH of the carrier buffer. In addition, the formation of a pH gradient requires hundreds to thousands of carrier ampholytes. Thus, to simulate IEF, one has to solve hundreds to thousands of coupled nonlinear partial differential equations at heavy computational effort and cost.

The computing power of a personal computer has grown significantly in recent years. Even so, it is still not possible to use a personal computer to simulate non-linear IEF for a real system because of the large number of non-linear partial differential equations involved. For a large matrix operation or system of equations, supercomputers still have to be used at high cost. However, parallel computing can be a cheap solution for large amounts of computation. Generally, there are two ways to perform parallel programming: open multi-processing (OpenMP) and message passing interface (MPI) [6-8]. OpenMP uses an application programming interface (API) that supports multi-platform shared memory multiprocessing. It is easy to program, debug, and run in OpenMP and the process is much like a serial code. OpenMP provides a fork and join execution, which executes a program as a single process, sequentially, until a parallelization for a parallel region is found. In addition, OpenMP supports only loop-level parallelism. On the other hand, MPI-based parallel computation runs on either shared or distributed memory, and it is also flexible and easy to approach. MPI-based parallel computation can provide high speed calculation than OpenMP because users can design and optimize MPI codes specific to a particular problem. One of the main advantages of MPI is that it uses distributed memory computers, which are much cheaper than large shared memory computers. However, in MPI-based parallelization, the calculation speed is dependent on the condition of the Ethernet connection and on the message passing size.

MPI-based parallel computing has been used in many areas, ranging from molecular calculation to fluid flow analysis (non-linear PDE). In 1993, Thangavel et al. [9] developed a parallel algorithm adaptable to supercomputers to construct the molecular distance matrix. They calculated distance polynomials of chemical graphs and showed the effect of the number of processors on computation time. Carvalho et al. [10] implemented the MPI-based parallel simulation algorithm for Monte Carlo simulation of a molecular system and they enhanced the speed by two-times with four processors. In 2005, Borstnik et al. [11] developed a parallel computing program for molecular dynamics (MD) simulation and implemented the split integration symplectic method (SISM). In their approach, the high-frequency vibrational motion corresponding to bonding interactions is treated analytically. Duncan et al. [12] introduced "Parallel Tandem", a fast and accurate search algorithm, to reduce the computational time necessary for the analysis of MS spectra against a protein sequence database. Recently, Huang et al. [13] showed the full power of ab initio quantum mechanics, which represented a full molecule by smaller "kernels" of atoms, to calculate the interaction of long chain molecules of biological and medicinal interest.

In addition to the aforementioned molecular level calculations, MPI-based parallel simulations were used in other computational areas. In 2000, Amodio et al. [14] presented a generalized parallel numerical method for ordinary differential equations as well as partial differential equations. They introduced a parallel block diagonal algorithm to solve a system of PDEs or ODEs for linear systems. Parallel processing was also performed to solve fluid problems. In 2005, Karimian et al. [15] applied a 3D, unstructured, finite volume method to solve incompressible Navier-Stokes equations, and found linear scalability within the available processors.

In this work, we introduce parallel implementation of a finite volume method for isoelectric focusing (IEF). To our knowledge, this is first parallel implementation of a solution method for an IEF problem to study the transient behavior of proteins as well as ampholytes. Parallelism was implemented by dividing the mass conservation equations of amphoteric molecules (proteins and ampholytes) to each sub processor. This parallel approach was demonstrated to be very successful for the IEF simulation, which requires a solution to a large number of mass conservation equations for each amphoteric molecule.

## 2. Ampholyte-based IEF model

The mathematical model of multidimensional IEF is presented in our earlier publications. Simulation of ampholyte-based IEF requires a simultaneous solution for the mass conservation equations for each amphoteric molecule, the charge conservation equa-

tion and the electroneutrality equation. The mass conservation equations for each amphoteric molecule (aka component) can be presented as [5]

$$\frac{\partial C_i}{\partial t} - \nabla \cdot \left[ D_i \nabla C_i + (\langle \mu_i \rangle \nabla \cdot \phi - \vec{U}) C_i \right] = 0 \qquad (1)$$

where $C_i$ , $D_i$ and $\langle \mu_i \rangle$ are the concentration, diffusion coefficient and effective mobility of component i , and $\vec{U}$ is the velocity of bulk flow. Electric potential, $\phi$ , is governed by the charge conservation equation

$$\nabla \cdot \left[ \nabla \cdot (\sigma \phi) + F \left( \sum_{i=1}^{N} D_i \langle z_i \rangle \nabla C_i - \vec{U}_i (\langle z_i \rangle C_i) \right) \right] = 0 \qquad (2)$$

where $\sigma$ is the conductivity ,

$$( \sigma = F \left[ \sum_{i=1}^{N} z_i^2 \omega_i C_i \right] ),$$

$\omega_i$ is electrophoretic mobility, N is the total number of components including a hydronium and hydroxyl. F is the Faraday constant, and $\langle z_i \rangle$ is the effective valence. The concentration of hydronium ions can be found from the electroneutrality condition of the form,

$$\sum_{i=1}^{N-2} \langle z_i \rangle C_i + C_H - \frac{K_w}{C_H} = 0 \qquad (3)$$

where $C_H$ is the concentration of hydronium, $K_w$ the reaction constant of water, and N is all the components, which are proteins as well as ampholytes except hydronium ($C_H$ ) and hydroxyl ($C_{OH}$).

### *2.1 Model assumption*

In this model, Joule heating is neglected since relatively small electric fields are used for separation. Electric field-induced electrokinetic flow is not considered here because in most of our IEF experiments the channel was coated with methylcellulose or other chemicals to suppress electroosmosis. Since all extant models of isoelectric focusing and isotachophoresis treat proteins as point charges, as virtually all treatments of electrophoretic separations do, we also assumed this model as a point mass. However, this assumption was not applied in the papers of Lenhoff's group [16], who took into account surface charge distributions. Simply, surface charge distributions give rise to dipole, quadrapole, octopole, etc., moments that couple with their respective electric field gradients and gradients of those gradients. Ultimately, for the magnitudes of electric fields, and their resulting gradients, commonly used in IEF, ITP and ZE, together with the small size of proteins under consideration, the higher-order moments can be ignored as having a negligible effect, primarily due to the averaging effect of rotational diffusion on multi-pole interactions.

An example where the dipole moment would be important is in the case of cells exposed to a strong electric field gradient. This field of study, known as dielectrophoresis, is currently undergoing resurgence in popularity due to its importance (to cells not proteins) in microscale array structures. Generally, the separation of biological molecules using electrophoresis or electric field is dependent on molecular weight as well as electrophoretic mobility in fixed gels, e.g., PAGE, and in solutions with high concentrations of (linear) polymer, where the frictional resistance of the polymer to the movement of the protein has a strong influence on the protein's effective mobility (and diffusion coefficient). However, a protein's electrophoretic mobility in free solution and low gel concentrations is generally taken as an empirical parameter that accounts for the molecular weight in terms of absolute mobility, which is inversely related to the drag. For this reason, this model considers the effect of molecular weight on the separation as electrophoretic mobility by taking into account absolute mobility as well as a valance [4].

### *2.2 Numerical model for IEF simulation*

A 2D finite volume method (2D FVM) is utilized to solve the transport equations that consider electro migration and diffusion given by Eq. (1), as well as the charge conservation equation defined in Eq. (2). The discretized algebraic equations are derived at each grid point for mass and charge conservation equations. In the discretized algebraic equations, the subscript *i* represents a component, while subscripts denote the grid numbering in the x- and y-direction of the spatial computation domain. The power law scheme is used to calculate the coefficients of the algebraic equations [17]. In the case of the electroneutrality equation, the Newton-Raphson method is used

to obtain the concentration of the hydronium ions. In our simulation, convergence criteria are for mass conservation and for charge conservation and chemical electroneutrality. Numerical results are obtained in a 2D straight channel, and 3000 grid points are used to obtain grid independent results in straight channel cases.

### *2.2.1 Serial code of IEF simulation*

To describe the IEF simulation, the concentrations ($C_i$) of all bio-components in the Eq. (1) are discretized by the finite volume method as follows.

$$\begin{aligned}&(C_i - C_i^o)\frac{\Delta x\Delta y}{\Delta t} + \left[(\langle \mu_i \rangle \nabla \cdot \phi_x) C_i\right]_e \Delta y - \\ &\left[(\langle \mu_i \rangle \nabla \cdot \phi_x) C_i\right]_w \Delta y + \left[(\langle \mu_i \rangle \nabla \cdot \phi_y) C_i\right]_n \Delta x \\ &-\left[(\langle \mu_i \rangle \nabla \cdot \phi_y) C_i\right]_s \Delta x = \left[D_i \frac{\partial C}{\partial x}\right]_e \Delta y - \\ &\left[D_i \frac{\partial C}{\partial x}\right]_w \Delta y + \left[D_i \frac{\partial C}{\partial y}\right]_n \Delta x - \left[D_i \frac{\partial C}{\partial y}\right]_s \Delta x\end{aligned} \tag{4}$$

where, $C_i^0$ is the concentration. In the previous step, the symbols ($e$ and $w$) represent the unit computational domain (j-space) and (n and s) the unit computational domain (k-space). The maximum discretized j and k are $J_{max}$ and $K_{max}$, respectively. In this model, the number ($I_{max}$) of the component ($C_i$) includes all summation of ampholytes, proteins, hydronium and hydroxyl ions. The concentrations ($C_i$) of these components are calculated over a computational domain, (j,k)-space in step 1.

For this reason, the numbers given by Eq. (1) should be solved for all components ($C_i$). However, in step 2, either $C_H$ or ($C_{OH} = \frac{k_w}{C_H}$) in Eq. (3) is calculated over a computational domain, the (j, k)-space, after obtaining the all components ($C_i$). Normally, $C_H$ is preferred for IEF simulation. In step 3, the electric field ($\phi$) in Eq. (2) is solved by the finite volume method in the same fashion as in step 1 over (j, k)-space. The other terms in Eq. (2) can be obtained from steps 1 and 2.

Step 1
*Do i = 1 to $I_{max}$*
*Do j = 1 to $J_{max}$*
*Do k = 1 to $K_{max}$*
*Solve the discretized equation of the concentration matrix based on based on the Tridiagonal matrix algorithm (TDMA) solver*
*Enddo*
*Enddo*
*Enddo*

Step 2
*Do i = 1 to $J_{max}$*
*Do j = 1 to $K_{max}$*
*Solve the hydronium matrix based on the Newton Raphson Method.*
*Enddo*
*Enddo*

Step 3
*Do $_j$ = 1 to $J_{max}$*
*Do $_k$ = 1 to $K_{max}$*
*Solve the discretized equation of the electric potential matrix $\phi_{jk}$ based on the Tridiagonal matrix algorithm (TDMA) solver*
*Enddo*
*Enddo*

The three main equations explain the IEF simulation in the serial code. Before step 1, we initially assume matrix $C_{ijk}$, $(C_H)_{jk}$ and $\phi_{jk}$ for solving Eq. (1). In step 1, three Do-loop commands are used to solve the concentration matrix $C_{ijk}$. In step 2, the concentration matrix $C_{ijk}$ conserved in step 1 is updated into Eq. (3), and the hydronium matrix $(C_H)_{jk}$ is obtained for the computation domain (j, k) using the Newton Raphson method. In step 3, the obtained concentration matrix $C_{ijk}$ from step 1and the hydronium concentration $(C_H)_{jk}$ from step 2 are updated into Eq. (2), and step 3 is executed to solve the electric potential matrix $\phi_{jk}$. Note that conductivity in Fig. 1 should include $(C_H)_{jk}$ and $(C_{OH})_{jk}$, but $(C_{OH})_{jk}$ can be calculated by the water reaction ($K_W = (C_H)_{jk}\ (C_{OH})_{jk}$) relation, where $K_w$ is the equilibrium constant of water. For conservation, all serial steps 1, 2, and 3 are repeated until all matrices $C_{ijk}$, $(C_H)_{jk}$ and $\phi_{jk}$ are conserved simultaneously. After conservation, the next time step ($\Delta t$) is moved. In Fig. 1, the flow chart of the serial code is given for isoelectric focusing, in detail. In the flow chart, the conductivity is calculated after obtaining all concentrations and hydronium based on Eq. (1).

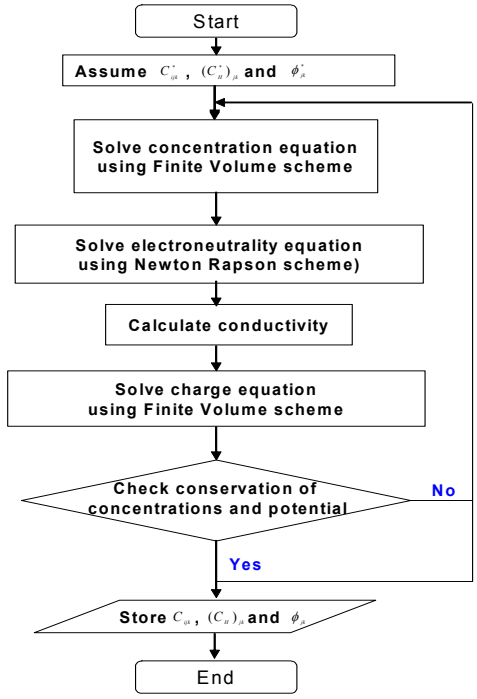


Fig. 1. Flow chart of serial code of isoelectric focusing. Superscript * is the initial guessed values.

### *2.2.2 Parallel code of IEF simulation*

Step 1

*DO i = iam*N+1 to iam*N+N*
*Do j =1 to $J_{max}$*
*Do k =1 to $K_{max}$*
*Solve the discretized equation of the concentration matrix $\boldsymbol{C_{ijk}}$ based on the Tridiagonal matrix algorithm (TDMA) Solver*
*Enddo*
*Enddo*
*Enddo*

Step 2

*Do j = iam*L+1,iam*iam*L+L*
*Do k =1 to $K_{max}$*
*Solve the hydronium matrix $(C_H)_{jk}$ the Newton Raphson Method.*
*Enddo*
*Enddo*

Step 3

*Do j = iam*L+1,iam*iam*L+L*
*Do k =1 to $K_{max}$*
*Calculate the coefficients of discretized equation for solving the electric potential matrix $\phi_{jk}$ in Eq. (2)*
*Enddo*
*Enddo*

Step 4

*Do **j** =1 to Jmax*
*Do k =1 to Kmax*
*Solve the discretized equation of the electric potential matrix based on the Tridiagonal matrix algorithm (TDMA) solver*
*Enddo*
*Enddo*

In the parallel code, N is $I_{max}$/p and L is $J_{max}$/p, and p is the total number of nodes, including master for parallelization. In an IEF simulation, the $I_{max}$ number of PDEs is solved for component conservation, one PDE for charge conservation and one non-linear equation for the electroneutrality condition. The parallelization of the concentration equations is carried out first by uniformly distributing N number of concentrations to each slave as well as the master in step 1 because all component conservation equations are independent of each other. In other words, all of the conservation equations do not couple with each other. If each slave and master take the same time to solve the same number of conservation equations, we can distribute the computational load equally to each node without CPU time loss, where iam is the identification (ID) number of the processor. The iam can be the master or slaves in the parallel code. Note that iam is 0 for the master. In step 2, the parallelism is achieved by dividing the computational domain sections equally for the electroneutrality equation. All of the domain data are equally shared among the slaves from the master except the first domain section, in which each of the slaves and the master can solve the same section of the computational domain. In steps 3, the Do-loop command is executed in parallel to construct the discretized coefficients of the finite volume method for step 4. However, in step 4, we cannot achieve parallelism to solve the electric potential matrix $\phi_{jk}$ because the TDMA solver requires information from the previous domain, and as a result, the electric potential matrix $\phi_{jk}$ is solved serially. In Fig. 2, the flow chart of the parallel code is shown for isoelectric focusing. Messages are communicated

between the concentration equations and the charge equation, and between the charge equation and the electroneutrality equation.

### *2.3 Message communication*

MPI generally requires messages to be passed between the master and slaves or among slaves for communication. In our parallel code, the concentration matrix of proteins and ampholytes ($C_{jk}$) is solved at all nodes, including the master. The master distributes and /or gathers all data information to/from slaves except the first N number of components for Eq. (3) or the first L number of computational domains for Eq. (3). In our program, messages are communicated mainly before step 1 , between step 1 and step 2, and between the message distribution step 5 and step 1. Two examples of message communication (sending and receiving) used in our code are presented below.

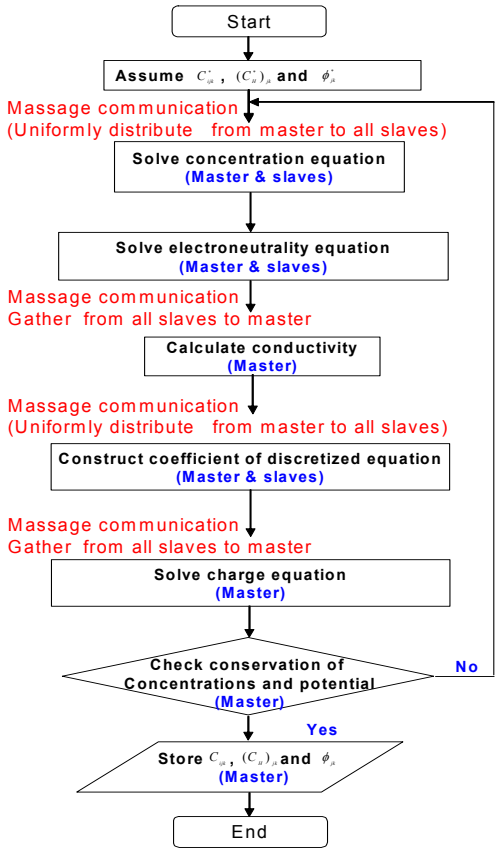


Fig. 2. Flow chart of parallel code of isoelectric focusing. Superscript * is the initial guessed values.

```
IF (iam.NE.master) THEN
 DO i= iam*N+1, iam*N+1
CALL MPI_SEND(C(1,1,I),Jmax*Kmax*N,MPI_DOU-
BLE_PRECISION,master,1,MPI_COMM_WORLD, ierr)
 ENDDO

ELSE IF (Iam.EQ.master) THEN
 DO slave =1, p-1
   DO i= slave*N+1, slave*N+1
CALL MPI_IRECV(C(1,1,i),Jmax*Kmax*N,MPI_DO-
UBLE_PRECISION,slave,1,MPI_COMM_WORLD,
request, ierr)
CALL MPI_WAIT (request,status, ierr)
   ENDDO
 ENDDO
ENDIF
```

where p is the total number of processors (including master and slaves). The concentration matrix $C_{jk}$ is represented as C(i,j,k) in the FORTRAN language. In Figs (3-4), the message communications between master and slaves are represented in detail. In Fig. 3, the communications take place when Eq. (1) is solved in the IEF simulation. The master sends or receives N equally divided concentrations to/from the each slave. In the communications, it is important that each slave sends/receives the same numbers of concentrations so that each slave takes equal CPU time to solve Eq. (1). After the slaves receive N numbers of concentration data, each slave solves N numbers of equations, which have N numbers of computational domains in series. Fig. 4 shows that communications occur in Eq. (3). When the electroneutrality equation is solved, the master sends or receives all concentrations of the L divided computational domains to/from the each slave. Each slave takes care of only its computational domain for solving the hydronium concentration matrix?($(C_N)_{jk}$)

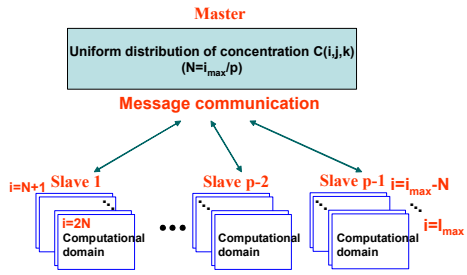


Fig. 3. Message communication between master and slaves for solving the conservation of concentrations equation (Eq. (1)).

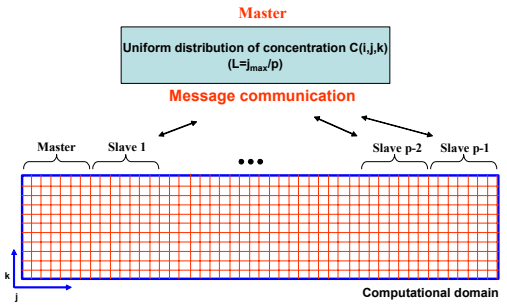


Fig. 4. Message communication between master and slaves for solving the electroneutrality equation (Eq. (3)).

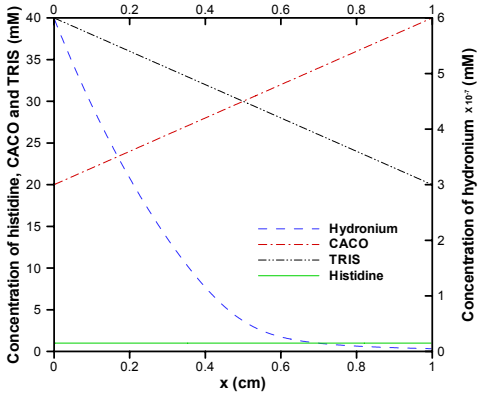


Fig. 5. Initial concentration distributions of CACO, TRIS, histidine, and hydronium for immobilized pH gradient case.

## 3. Results and discussion

### *3.1 Validation of the IEF model*

To verify the 2D model, an immobilized pH gradient IEF is simulated. The input parameters were taken from the literature [18-19]. A pH range of 6.21-8.3 is formed using CACO and TRIS. In this simulation, the mobilities of CACO and TRIS are set to zero, essentially fixing a pH gradient. Histidine is uniformly mixed to an initial concentration and distributed over the entire channel, but the concentration lines of CACO and TRIS intersect each other, as shown in Fig. 5

The amount of histidine is much smaller than that of CACO and TRIS (less than 5%). The electric potential at the anode is varied from case to case, while the electric potential at the cathode is grounded for a 1 cm long and 100 μm wide channel. The evolution of the histidine concentration profile is shown in Fig. 6 for a nominal electric field of 40 V/cm. During isoelectric focusing , the concentration profile of histidine changes from an early asymmetrical shape to a symmetric form. At steady state, the concentration profile becomes nearly Gaussian. In this simulation, steady state is achieved within 60 minutes of the initiation of IEF. These simulation results are in good agreement with findings of the literature [18-19]. In an earlier work, we have our mathematical and numerical model validated with the literature in detail [5]. The objective of this study is to show how the computational time can be reduced using multiple processors using cluster machines and MPI.

### *3.2 Parallel implementation*

IEF simulations were conducted for 22 ampholytes for a pH range of 6 to 9 under an electric field of 300

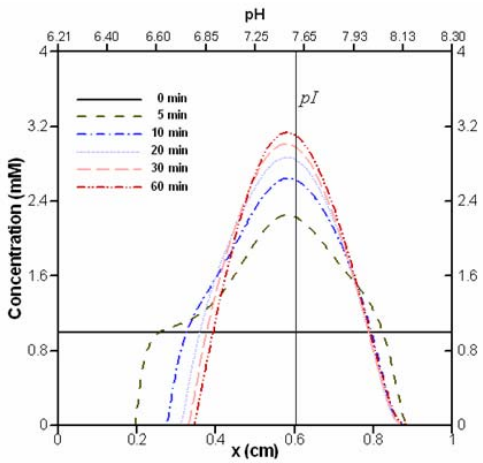


Fig. 6. Concentration profiles of Histidine at different times. The pH of histidine is 7.61.

V/cm in a 1 cm long channel. The parallel code was developed by using the finite volume method in the FORTRAN based MPI. The serial and parallel codes were simulated on the DELL beowulf cluster, which has 16 nodes , each of which has 2 processors.

The main CPU of master and slaves has a 3.4 GHz/2MB cache, Xeon and 8 GB DDR2 400Hz and each front side Bus runs at 800 MHz. For Ethernet with switch, 1000 Mbps is connected. Each processor can communicate by sending and receiving data via a high speed Ethernet connection. The MPI-based parallel code was tested with the various numbers of processors. In addition, 24, 48, 64 and 96 separation components, such as proteins and ampholytes, were simulated in the IEF parallel code, so that the effect of the number of separation components on the computational speed enhancement of our parallel code could

be evaluated. The CPU times required to focus on all proteins and ampholytes are shown in Fig. 7 for different numbers of processors (1, 2, 4 and 8). In addition, computational speed was enhanced, as shown in Fig. 8, for the focusing of the separation components.

In our paper, speed enhancement was defined as how much faster the parallel code on the multiple processors obtained the focusing of the proteins and ampholytes than the parallel code on the single processor. To represent the global speed enhancement, performance of the parallel code was tested. To evaluate the parallel performance, for example, two steps of the three steps were processed in parallel, while the other steps remained in series for the case in which four CPUs were used in the presence of 96 biocomponents for the parallel processing of IEF simulation. The computational times of the parallel performance test were 236, 412 and 375 hours. Meanwhile, the computational time of the serial processing was 425 hours. The CPU time to complete the focusing took 174 hours in parallel processing. In Fig. 7, the CPU times to focusing are 133, 298, 425 and 766 hours with the single processor.

The maximum time reductions were 65, 137, 174 and 274 hours when four CPUs were used for the parallel processing. When eight CPUs were employed, the CPU times (81, 159, 200 and 316 hours) increased for all cases, regardless of the separation components, mainly because of the excessive massage passing between step 1 and step 2, and between step 5 and step 1. In our parallel code, there are three main equations such as mass conservation equations, charge conservation equation and the electroneutrality equation for IEF simulation. If 24 separation components are used, twenty-four mass conservation equations, one charge conservation equation and one electroneutrality equation have to be solved for IEF simulation. Perfect parallelization was obtained for the mass conservation equations in the Eq. (1) and the electroneutrality equation in Eq. (3), but serial processing was still used for the charge conservation equation in Eq. (2) because of the TDMA solver. Thus, the parallelization in our parallel code was well attained up to four CPUs, but if eight CPUs are used, the large amount of massage passing diminished the parallel effect in IEF simulation. For this reason, the CPU time increased for more than eight CPUs.

Fig. 8 shows that as the number of separation components increases, so does the effect of speed enhancement. This can be explained when we separate a

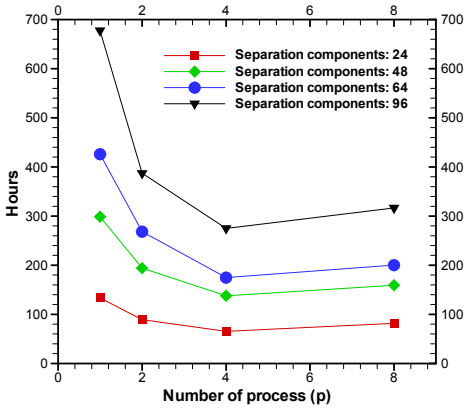


Fig. 7. CPU times required for focusing of all components in ampholyte based IEF simulation according to the number of processors (p). The focus means that all proteins and ampholytes are stable at steady state. The focusing time was 170s at 300 V/cm in a 1 cm channel.

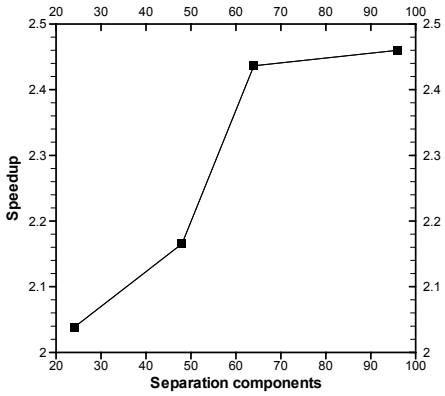


Fig. 8. Computational speed enhancement with increase of the number of separation components. The speed enhancement is calculated based on the ratio of the computational time using 4 CPUs with parallel processing to the time using 1 CPU with serial processing.

large number of separation components, in which case, the number of the mass conservation equations also is increased corresponding to the increase in the number of the separation components, but the numbers of the charge equation and electroneutrality equation remain the same. In other words, the effect of a larger number of separation components on parallel computation is favorably higher because the mass conservation equations can be perfectly parallelized. Our parallel code enhanced speed with an increase of the number of separation components. This result is favorable in IEF simulation because the goal of IEF simulation is to pursue the separation of a

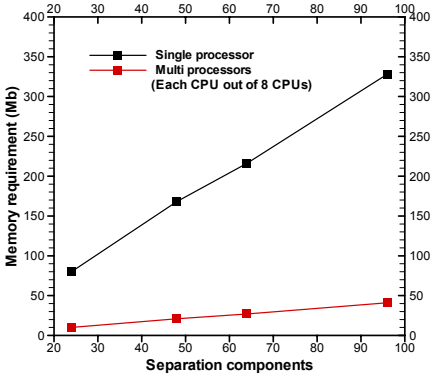


Fig. 9. Memory requirement in both serial and parallel processing.

large number of separation components. In addition to speed enhancement, memory requirement and FLOPs are also good factors to study in the evaluation of parallel processing. Fig. 9 provides the memory requirement for both serial and parallel processing, and shows the advantage of parallel processing. The comparison is carried out for parallel processing with eight CPUs. Memory requirement is linearly proportional to the number of separation components for both serial and parallel processing. Each CPU had the same memory requirement for IEF simulation. Thus, uniform computation load was allowed to each CPU in parallel processing. However, FLOPs were not an effective way to evaluate the efficiency in this application. Our vectorized program was used to solve highly nonlinear, stiff partial different equations. Unlike routine calculations, the number of calculations varies with iteration (increases or decreases). Hence, it is very difficult to estimate the exact number of FLOPs without adding a counter in the program, which will add time to the overall computing, both the counting and sharing process. Moreover, FLOPs are sometimes misleading in the sense that one can delegate some unnecessary floating point calculations to an idle processor to show the better ability of an algorithm. This study was primarily concerned with the overall time reduction in the calculations of nonlinear IEF using parallel processing. Beowulf cluster was found to be very inefficient because of its slow network connection.

## 4. Conclusions

Parallel implementation for ampholyte-based IEF simulation was firstly introduced. An IEF simulation requires the solving of numerous mass equations corresponding to the number of separation components. The parallelism of IEF simulation was achieved by equally dividing the mass equations among the slaves to solve these equations independently and simultaneously. The proposed parallelization to decompose the separation components showed favorable performance as the number of separation components increased in the IEF simulation. Computation time was reduced with the increase of the number of processors up to eight CPUs, and the maximum time reduction was obtained when four CPUs were used, as shown in Fig. 7. The maximum speed enhancement was 2.46 times the computational time of a single processor shown in Fig. 8. If the IEF code is perfectly parallelized for the charge conservation equation in Eq. (2), especially the TDMA solver, the time can be dramatically reduced with the increase of the number of processors in a parallel computer.

## Acknolwledgment

This research was supported by the Yeungnam University research grants in 2008

## References

[1] K. A. E. Stenberg, P. T. Riikonen and M. Vihinen, KinMutBase, *A database of human disease-causing protein kinase mutations*, Nucleic Acids Research 27 (1) (1999) 362-364.

[2] C. R. Merril, M. P. Goldstein, J. E. Myrick, G. J. Creed and P. F. Lemkin, *The protein disease database of human body fluids: I. Rationale for the development of this database*, Applied and Theoretical Electrophoresis 5 (1995) 49-54.

[3] H. Cui, K. Horiuchi, P. Dutta and C. F. Ivory, *Isoelectric focusing in a poly(dimethylsiloxane) microfluidic chip*., Anal. Chem. 77 (2005) 1303-1309.

[4] R. A. Mosher, D. A. Saville and W. Thormann, *The dynamics of electrophoresis*, VCH, Weinheim, (1992).

[5] J. Shim, P. Dutta and C. F. Ivory, *Modeling and simulation of IEF in 2-D microgeometries,* Electrophoresis 28 (4) (2007) 572-586.

[6] The message passing interface (MPI) standard. Available at: http://www-unix.mcs.anl.gov/mpi/

[7] OpenMP architecture review board, OpenMPC and C++ application program interface, version 2.0. Available at: http://www.openmp.org/blog/

[8] R. Aversa, B. D. Martino, M. Rak, S. Venticinque

and U. Villano, *Performance prediction through simulation of a hybrid MPI/OpenMP application,* Parallel Computing 31 (10-12) (2005) 1013-1033.

[9] P. Thangavel and P. Venuvanalingam, *Algorithms for the Computation of. Molecular distance matrix and distance polynomial of chemical graphs on parallel computers, J. Chem. Inf. Comput.* Sci. 33 (1993) 412-414.

[10] A. P. Carvalho, J. A. N. F. Gomes and M. N. D. S. Cordeiro, *Parallel implementation of a Monte Carlo molecular stimulation program,* J. Chem. Inf. Comput. Sci., 40 (3) (2000) 588-592.

[11] U. Borstnik and D. Janezic, *Symplectic molecular dynamics simulations on specially designed parallel computers,* J. Chem. Inf. Model. 45 (8) (2005) 1600-1604.

[12] D. T. Duncan, R. Craig and A. J. Link parallel tandem: *a program for parallel processing of tandem mass spectra using PVM or MPI and X! Tandem,* Journal of Proteome Research 4 (2005) 1842-1847.

[13] L. Huang, L. Massa and J. Karle, *Kernel energy method: The interaction energy of the collagen triple helix, J. Chem.* Theory Comput. 3 (2007) 1337-1341.

[14] P. Amodio, J. R. Cash, G. Roussos, R. W. Wright, G. Fairweather, I. Gladwell, G. L. Kraut and M. Paprzycki, *Almost block diagonal linear systems: sequential and parallel solution techniques, and applications,* Numer. Linear Algebra Appl. 7 (2000) 275-317.

[15] S. A. M. Karimian and A. G. Straatman, *Discretization and parallel performance of an unstructured finite volume Navier-Stokes solver,* International Journal for Numerical Methods in Fluids, 52 (6) (2006) 591-615.

[16] K. S. Chae, A. M. Lenhoff, *Computation of the electrophoretic mobility of proteins,* Biophysical Journal 68 (1995) 1120-1127.

[17] S. V. Patankar, *Numerical heat transfer and fluid Flow,* Hemisphere, New York, (1980).

[18] D. A. Saville and O. A. Palusinski, *Theory of electrophoretic separations Part I: Formulation of a mathematical model,* AIChE J. 32 (1986) 207-214.

[19] O. A. Palusinski, A. Graham, R. A. Mosher, M. Bier, D. A. Saville, Theory of electrophoretic separations. II - Construction of a numerical simulation scheme and its applications, AIChE J. 32 (1986) 215-223.

**Jaesool Shim** received his B.S. in Mechanical Engin-eering from Pusan National, Korea, in 1995. He then received his M.S. from Pohang University of Science and Technology and Ph.D. degrees from WSU in 1997 and 2007, respectively. Dr. Shim is currently a Professor at the School of Mechanical Engineering at Yeungnam University in Gyeongsan, Korea. His research interests include Lab-on-a chip and Bio/Nano technology.

**Prashanta Dutta** received his B.S. in Mechanical Engineering from Bangladesh University of Engineering and Technology, Bangladesh, in 1994. He then received his M.S. from University of South Carolina and Ph.D. degrees from Texas A&M in 1997 and 2001, respectively. Dr. Dutta is currently a Professor at the School of Mechanical Engineering at WSU in Pullman, USA.

**Cornelius F. Ivory** received his B.S. in Chemical Engineering from the University of Notre Dame, USA, in 1974. He then received his M.S. and Ph.D. degrees from Princeton University in 1976 and 1980, respectively. Dr. Ivory is currently a Professor at the Department of Chemical Engineering at WSU in Pullman, USA.