1997

A parallel Algorithm for Determining Orientations of Biological Macro- molecules Imaged by Electron Microscopy

Timothy S. Baker

Ionana M. Boier Martin

Dan C. Marinescu

Report Number:
97-055

A PARALLEL ALGORITHM FOR DETERMINING ORIENTATIONS OF BIOLOGICAL MACROMOLECULES IMAGED BY ELECTRON MICROSCOPY

Timothy S. Baker
Ioana M. Boier Martin
Dan C. Marinescu

CSD-TR #97-055
December 1997

A Parallel Algorithm for Determining Orientations of Biological Macromolecules Imaged by Electron Microscopy

Timothy S. Baker
tsb@bragg.bio.purdue.edu
Biogical Sciences Department
Purdue University
West Lafayette, IN, 47907

Ioana M. Boier Martin
imartin@iusb.edu
Computer Sciences Department
Indiana University
South Bend, IN 46634

Dan C. Marinescu
dcm@cs.purdue.edu
Computer Sciences Department
Purdue University
West Lafayette, IN, 47907

Abstract

A parallel version of the Polar Fourier Transform (PFT) method that facilitates determination of orientations of biological macromolecules imaged with electron microscopy techniques is presented. The algorithm is designed to reduce the computing time of the PFT method by as much as three orders of magnitude by taking advantage of the speed, storage capacity, and I/O bandwidth of high-performance parallel systems and, at the same time, run efficiently on clusters of workstations.

1. Introduction

Structural biology is an area of biology which focuses on determination of the three-dimensional (3D) atomic structure of biological macromolecules. The ultimate goal is to determine where the atoms are located and how they interact during biochemical reactions. In the case of small molecules (i.e., with a few thousand atoms) structure determination has nowadays become a routine process. By contrast, the determination of macromolecular structures such as viruses (i.e., with hundreds of thousands or millions of atoms) remains a lengthy and difficult task. It typically involves very large data volumes which require high-performance computing for their processing and efficient visual representations for their analysis and interpretation.

Nuclear magnetic resonance (NMR), X-ray crystallography, and electron microscopy are the methods of choice for gathering information about the 3D atomic structure of biological macromolecules. NMR methods can be used to obtain 3D models of small proteins, but not to generate detailed information about the arrangement of atoms in large macromolecules. X-ray crystallography is a suitable method for the study of such large structures, provided that crystals that diffract X-rays at high resolution can be produced. This is often a difficult and time-consuming task. The recent successes in determining the structure of the Hepatitis B virus by means of electron microscopy [2], [4] represent an important step away from the need to have ordered specimens with crystalline symmetry and towards the ultimate goal of solving the atomic structures of macromolecules which do not crystallize [12]. They also prove electron microscopy to be a well-suited method for structure determination of very large symmetrical particles.

The problem of determining the 3D structure of a virus particle from electron micrographs (i.e.,

images obtained with the electron microscope) reduces to the mathematical problem of reconstructing the structure of a 3D object given a set of 2D projections of the object in various unknown orientations. The solution is provided by the Projection Theorem [11] and its main steps are illustrated in Figure 1. Individual particle images are extracted from a number of digitized electron micrographs. These constitute the different projected views of the virus particle and their 2D Fourier transforms represent central sections of the 3D Fourier transform of the particle under study.

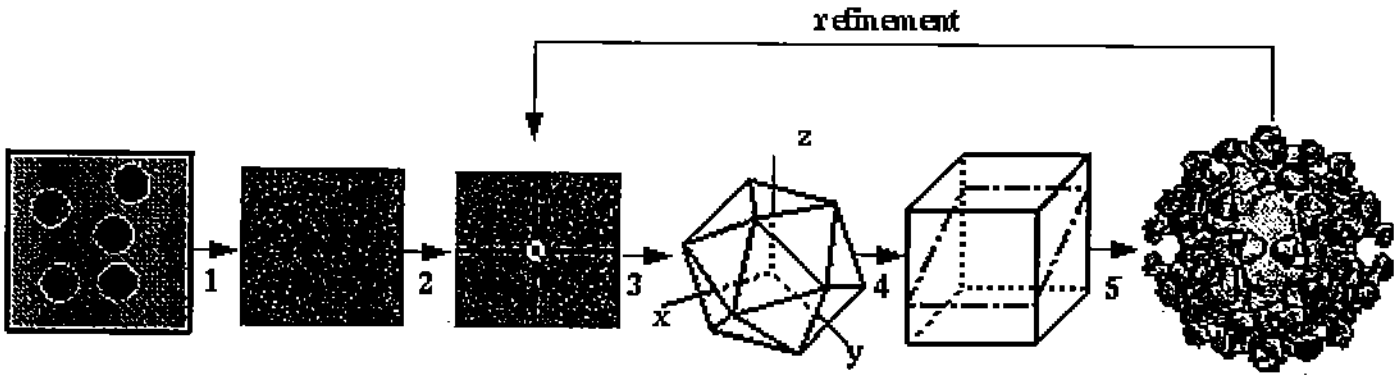

Figure 1. Schematic representation of the reconstruction process from 2D images to the 3D structure of a virus particle. Step 1: extract individual particle images from electron micrograph. Step 2: determine initial location of each particle center. Step 3: determine each particle view orientation. Step 4: fill in 3D Fourier transform. Step 5: compute3D electron density map.

The number of projections to be collected depends on the desired resolution of the final structure and on the particle size. Next, the orientations of the specimen that gave rise to these projections must be determined [7]. Best results are often obtained in the case of highly symmetrical particles such as icosahedral viruses because the high symmetry leads to redundancies in the Fourier transform data and this in turn aids the orientation search process. The 3D Fourier transform of the virus particle is calculated from the 2D Fourier transforms of the projections. The values of the 3D transform must be sampled at the points of a 3D regular grid and this requires interpolation methods [11], [13]. The final step is to compute the electron density function which characterizes the virus structure from the calculated 3D Fourier transform by an inverse Fourier transformation.

In this paper we describe a multi-phase, multi-resolution parallel algorithm for determining the orientations of particle images based on the Polar Fourier Transform (PFT) method [1]. In Section 2 we overview the PFT method, in Section 3 we outline a new algorithm and in Section 4 we present the parallel search algorithm for shared and for distributed memory systems.

2. The PFT Method

The identification of particle orientations is a critical step in determining reliable 3D structures of biological macromolecules from electron micrographs. The proper combination of 2D particle

images to produce a 3D electron density map depends heavily on the accuracy of this step. Several methods have been developed for solving this task. The *Common Lines* method [6] works well with high-contrast images, but may lead to inconsistent results when applied to low-contrast, noisy micrographs. A statistical approach that helps overcome some of the limitations of the Common Lines method is the *Modified Common Lines* method [7]. An alternative is offered by the *Polar Fourier Transform) (PFT)* method [1]. The main steps of this method are shown in Figure 2.

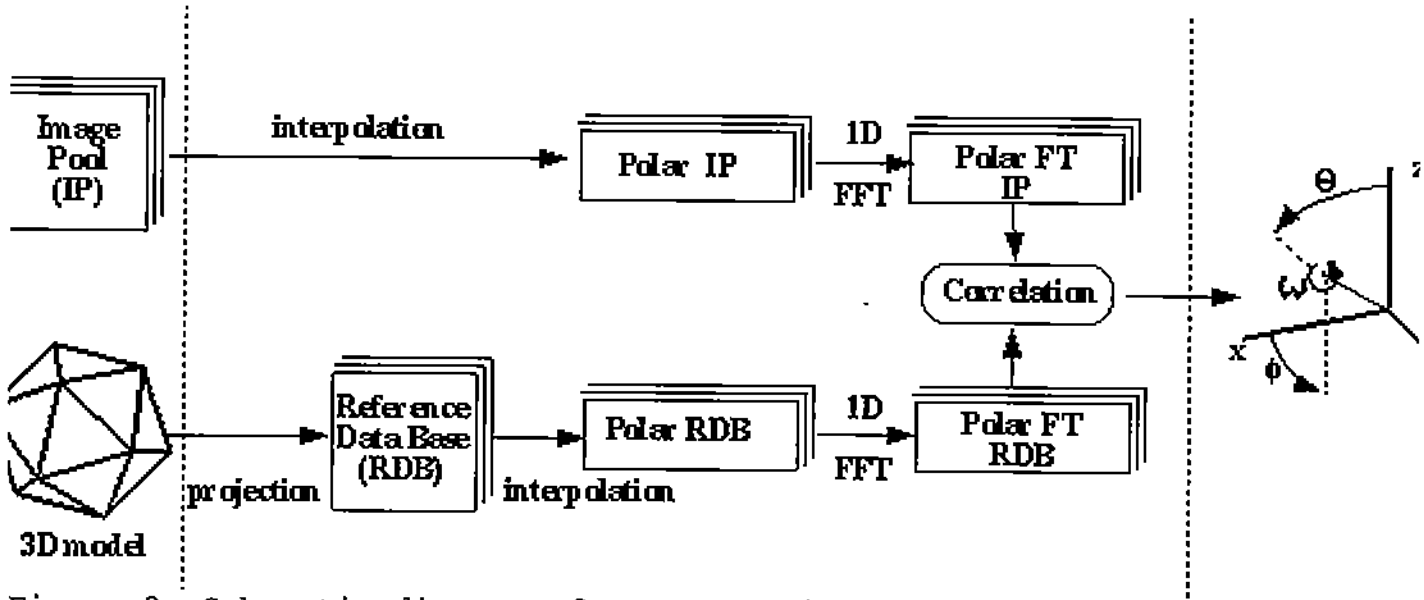

Figure 2. Schematic diagram of one computing cycle of the PFT algorithm. The input consists of an image pool (IP) consisting of the individual 2D views extracted from micrographs and a 3D model. The processing typically begins by computing a series of m projections (bottom row). These form a reference data base (RDB). Then the polar coordinate interpolations, Fourier transformations, and correlations are performed as indicated by the arrows. The output consists of a set of n orientations, one for each image in IP.

In addition to the collection of n 2D particle images whose orientations are to be determined, the method uses as input data a 3D electron density map which serves as a high signal-to-noise model. Such a model may be a computer-generated model, a low-resolution map previously computed, or a map corresponding to a similar, already known, structure. From this model, a reference database consisting of m different views is generated. For instance, for an icosahedral particle, these views cover one half of the asymmetric unit of the structure (e.g., 1/120 of an icosahedron, from θ in the range (69,90) deg and φ in the range (0,32) deg) as shown in Figure 3. Each cartesian reference view (x,y) and each 2D image is interpolated onto a polar grid (r,γ) which subdivides the data in a series of equally-spaced annuli, from $r = 0$ to a radius just outside the particle edge. The advantage of the polar representation is that it is rotation invariant, which greatly reduces the complexity of subsequent computations for determining the orientation parameters. In the next step, each of the reference views and images is Fourier transformed along

the azimuthal direction γ. The result of this operation is a series of one-dimensional Fourier transforms (PFTs) of rotational power spectra, one for each annulus of data. Each of the image PFTs is correlated with all model PFTs to identify which model PFT best matches each image PFT. This process provides an initial estimate of the values of φ and θ for each image. The rotation angle ω and the sign of φ are determined by the rotational correlation (γ direction).

With the orientations known for all n images, a 3D reconstruction can be computed. This reconstruction serves as a new model in the next cycle of refinement of the parameters (θ,ϕ,ω). The refinement is repeated until no significant change occurs in the values of the correlation coefficients.

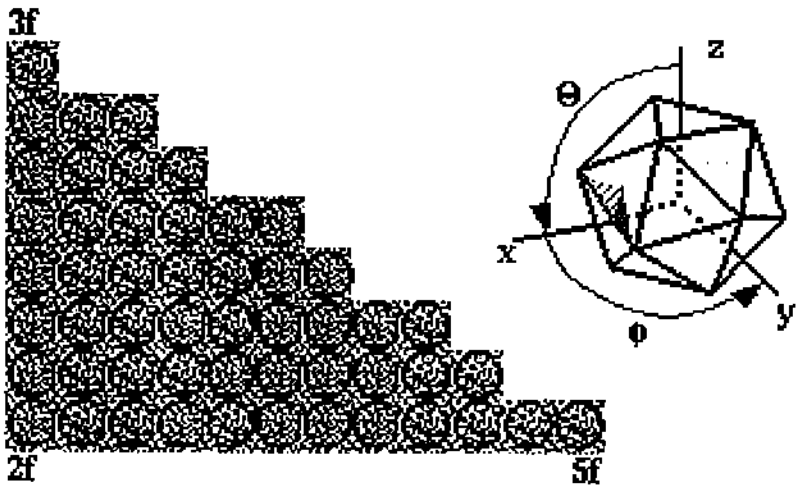

Figure 3 Schematic representation of the generation of the RDB data from model. Fifty-two views are obtained by projecting the model at 3 deg angular increments for orientations covering one-half of the icosahedral asymmetric unit: three-fold view at top left, two-fold view at bottom left and five-fold at bottom right (shaded half triangle shown on the icosahedron). Also shown is an icosahedron in standard orientation: three of its two-fold axes are aligned with the axes of coordinates.Three angles define the orientation of the icosahedron: θ, rotation in the (x,z)plane, from z towards x, φ, rotation in the (x,y) plane, from x towards y, and ω, (not depicted) defines the orientation about the (θ,φ) direction.

A major advantage of this algorithm is that the raw image data is compared to a relatively noise-free 3D model, rather than to other noisy data as it is the case in the Common Lines techniques. In addition, the PFT method makes use of all the available data, whereas in Common Lines only a fraction of the Fourier data is sampled. These features of the PFT technique have let to more consistent and reliable results in the analysis of virus structures.

However, in its sequential implementation, this method, although faster relative to other methods,

is not suitable for real-time high-resolution reconstructions. The determination of the orientations of a couple of thousand particles may take up to several days to compute. Our objective is to reduce this computing time by as much as three orders of magnitude, by taking advantage of the speed, storage capacity, I/O bandwidth, and latency of high-performance parallel systems [5], [8] [9].

3. A New Algorithm for Determining Particle Orientation.

We review the basic idea of the sequential algorithm. The algorithm is iterative, at each iteration we start with a model of the virus particle obtained by 3D reconstruction during the previous iteration. We know the electron density at each grid point of a 3D lattice. Using this information we construct a Reference Data Base, RDB, consisting of m images. Each image is a projection of a virus particles whose orientations we know. The RDB can be computed at different angular resolution, the better the resolution, the larger the size of the database. For example at 3 deg resolution the database consists of 52 images, at 1 deg there are 382 images, and at 0.3 deg there are 3,943 images. We also have an Image Pool, IP, consisting on n images whose orientation we want to determine. These images are obtained by isolating individual virus projections in several micrographs. We compare each image in IP with each image in the RDB and determine a "best fit".

The execution time, T_0, for one iteration of the search process is: $T_0 = I \times n \times m$ with I the time to evaluate the "fitness" of a pair (IP image, RDB image), n the number of images in IP, and m the size of the RDB. The algorithm we propose includes several enhancements that are discussed separately and the speedup (the ratio of the execution time *without* the enhancement and *with* the enhancement) is estimated for each case.

Compression. The time I, to evaluate the fitness of a pair of images and the space to store an image can be reduced if we use compressed images instead of the original images. For example the space needed for one 220 x 512 projection is about 1 MB. We can compress each image e.g by sub-sampling, take every second pixel in each row and each column. In this case the a (220 x 512) pixel image is reduced to an image of size (110 x 256) pixels for a compression factor $C = 4$. The compressed image takes only 256 KB of storage. Clearly, better compression schemes could be used, for example we can use wavelet transformations. The speedup due to compression is: $S_C = C$ since the time to evaluate the fitness of each pair is proportional with the size of the image.

Multi-phase, multi-resolution search. Instead of constructing a high resolution RDB and looking for the best match for each image in the IP, we propose a three phase scheme, as shown in Figure 4. In the first phase we construct a "low resolution RDB", say at 3 deg resolution, with m_1 elements. The search time in this small RDB is $T_1 = I \times n \times m_1$. In the second phase we construct a "medium resolution" RDB, say at 1 deg resolution. We limit the search in this database to a small region of size $r_2 \times m_2$ with $r_2 \ll 1$, e.g. $r_2 = 0.01$ around the area of the best fit during phase one. The search time in the second phase is $T_2 = I \times n \times (r_2 \times m_2)$. For the third phase we generate a "high resolution RDB" say at 0.3 deg. Again, we limit the search to a small region of

size $r_3 \times m_3$ determined using the results of the best fit during phase two. The search time for the third phase is: $T_3 = I \times n \times (r_3 \times m_3)$.

The total search time in this new scheme is: $T = I \times n \times (m_1 + r_2 \times m_2 + r_3 \times m_3)$

The speedup due to the multi resolution search is:

$$S_M = T_0/T = 1/(m_1/m_3 + r_2 \times m_2/m_3 + r_3) \approx 1/r_3.$$

Note that T_0 was computed assuming m_3 images in RDB.

Parallel search. If we have P processors then each one can process a fraction n/P of images in the IP for a speedup $S_P = P$.

All the improvements discussed above are cumulative and could lead to a total speedup of:

$$S = S_C \times S_M \times S_P = \frac{C \times P}{r_3}$$

Assuming that $C = 4$, $P = 20$, and $r_3 = 0.01$, the total speedup of the search process is $S = 8,000$.

4. A parallel search algorithm for shared and distributed memory systems.

The parallel search algorithm requires solutions to two problems: work allocation and data management and distribution. The basic approach to the work allocation problem is straightforward. Assuming that there are P processors have each one generate an equal fraction of the Reference Data Base then process an equal number of particle projections from the Image Pool.

The critical issue is the data management and data distribution. If σ is the size of an image, then the total amount of space needed for RDB and IP is $\mu = \sigma \times (n + m)$. Assuming that $\sigma = 1$ MB, $n = 10,000$, and $m = 4,000$, the total amount of space needed is $\mu = 14$ GB. Clearly this is a very large amount of memory and without a clever scheme, the performance of any implementation will be negatively affected by the I/O speed. We propose *a sliding window approach*. In the followings we discuss only the third phase of the algorithm and the corresponding data management scheme.

The approach discussed above applies to a shared memory system. For a distributed memory system each processor has to compute first a segment, a range of frames in RDB the processor will manage. Then each processor needs to select from the Image Pool those images expected to have a best fit in its own section of the RDB. Then the processor may apply the sliding window approach discussed above. We have a *spatial partitioning* of the RDB and IP corresponding to data distribution to different processors and a *temporal partitioning* to ensure the sliding window

advantage.

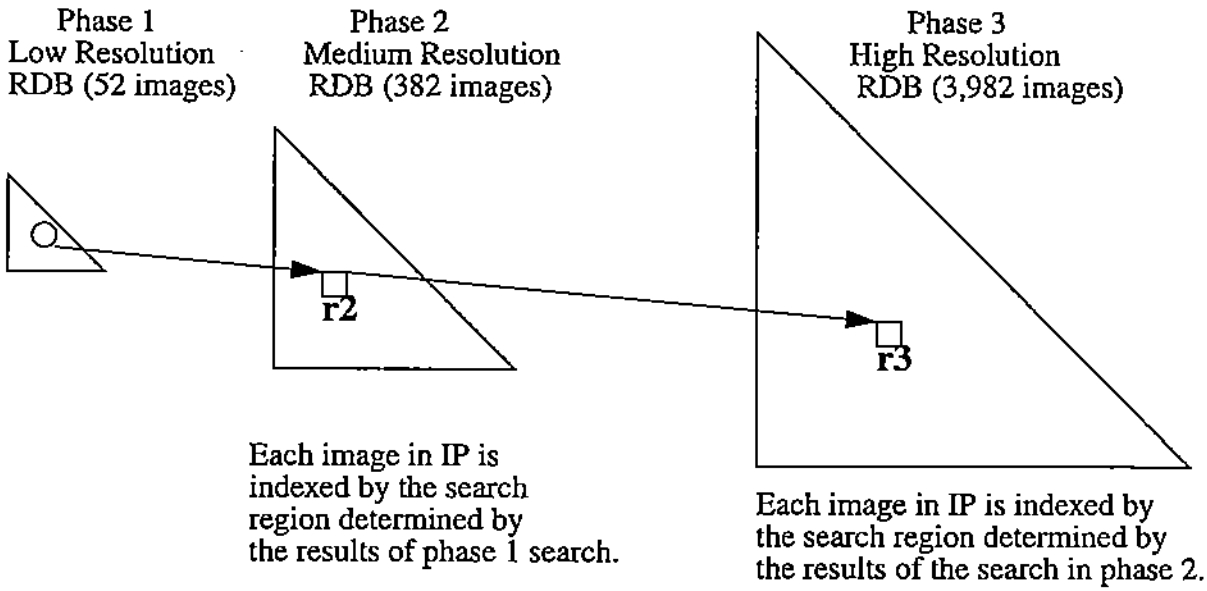

Figure 4. The multi-phase, multi resolution search.

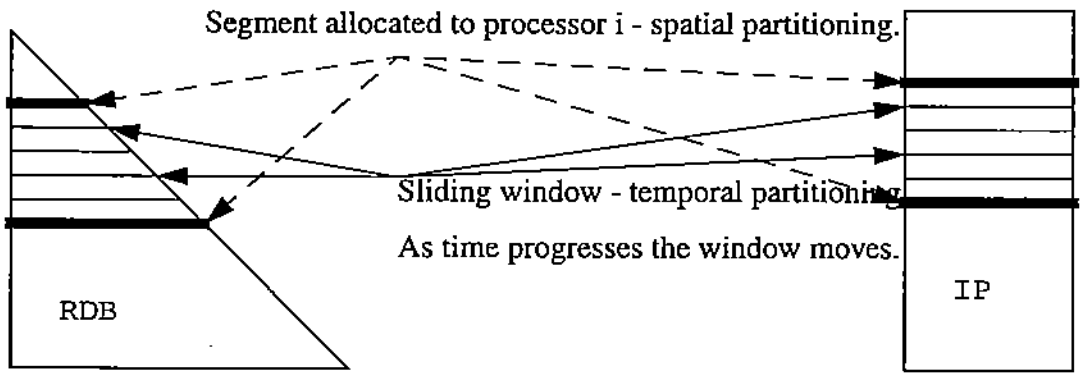

Figure 5. The spatial and the temporal partitioning of the RDB and the IP for the distributed memory systems case. Each processor manages a segment of the RDB and uses a sliding window to process images from IP allocated to it.

Instead of computing the entire high resolution database with m images we compute a window of m' images in the range $((\varphi_{min},\theta_{min}),(\varphi_{max},\theta_{max}))$, we call this RDB', and a corresponding pool of n' images for which we expect to find a best fit in RDB', and call it IP'. As time progresses the sliding window moves, but at any given time, the space needed is $\mu' = \sigma \times (n' + m')$, Figure 5.

The pseudocode for the distributed memory case follows:

```
/*****************************************/
/*        Phase 1 - processor i          */
/*****************************************/
  constructRDBphase1;              /* All processors compute the low resolution RDB*/
  if (compressionDesired) compress RDB;
                                     /* Use the IPpartitioning1 to compute the first
                                       and last image to be processed by processor i */
  calculateIPpartioning1 (myFirstIPimage(i), myLastIPimage(i);
  myFirstRDBimage (i) = 1; myLastRDBimage(i) = m1; /*set serach limits in RDB */
  /* Begin Serach */
  for j=myFirstIPImage(i), myLastIPimage(i) do {
     read IPimage(j);
     if (compressionDesired) compressIPimage(j);
     best(j) = -1;
     bestfit(j) = -1;                  /* The fit function returns a positive value */
     for k= myFirstRDBimage(i), myLastRDBimage(i), do {
       fitValue = fit (RDBimage(k), IPimage(j);
       if (fitness > bestfit(j)) do {
            bestfit(j) =fitness;
            best(j) = k;
       }
     }                                 /* End processing IPimage (j) */
    bestRT(j) = map(best(j), R, Theta);   /* determine the R and Theta of the best fit */
    compute the search region for phase 2, Rmin, Rmax, ThetaMin, ThetaMax;
  }                                    /* End of phase 1 */'
  /* End Search */
  signalEndPhaseOne(i);             /* Processor i signals End of Phase 1 */
  waitForAll;        /* All processors need to finish Phase 1 before going to Phase 2 */
/*****************************************/
/*        Phase 2 - processor i          */
/*****************************************/
   bestRTsort;                  /* Run a distributed parallel sort on the bestRT data and use
                                   its results to compute myFirstIPimage, myLastIPimage,
                                   myFirstRDBimage and myLastRDBimage */
   calculateIPpartitioning2 (myFirstIPimage(i), myLastIPimage(i));
   calculateRDBpartitioning2 (myFirstRDBimage(i), myLastRDBimage(i));
   constructRDBphase2(i);
   search;                          /* Use a modified version of the code in Phase 1 */
   signalEndPhaseTwo(i);            /* Processor i signals End of Phase 2 */
   waitForAll;        /* All processors need to finish Phase 2 before going to Phase 3 */
/*****************************************/
/*        Phase 3 - processor i          */
/*****************************************/
 /* use myFirstIPimage and myLastIPimage from phase 2; all IPimages are in cache */
    ws = WindowSize(myFistRDBimage,myLastRDBimage);
    nsteps = (myLastRDBimage - myFirstRDBimage) / ws;
```

```
for iter = 1 to   do {
  computeIPrange (IPfirst, IPlast);
  computeRDBrange(i,ws,RDBfirst, RDBlast);
  constructRDBphase3(i, RBDfirst, RDBlast);
  search;
}
signalEndPhaseThree(i);          /* Processor i signals End of Phase 3 */
waitForAll;                      /* All processors need to finish Phase 3 */
```

This algorithm is currently implemented for a shared memory system, an SGI Origin 2000 machine, and for a cluster of workstations. The distributed memory version of the algorithm uses MPI for inter-process communication.

Acknowledgments

This research has been partially supported by the National Science Foundation grants BIR-9301210 and MCB-9527131, by the Scalable I/O Initiative, by a grant from the Intel Corporation, a grant from the Purdue Research Foundation, and a Grant-In-Aid of Research from Indiana University.

References

[1] Baker, T.S. and R. H. Cheng, A Model-Based Approach for Determining Orientations of Biological Macromolecules Imaged by Cryoelectron Microscopy, *Journal of Structural Biology*, **116**, 120--130, 1996.

[2] Bottcher, B., S.A. Wynne, and R.A. Crowther, Determination of the Fold of the Core Protein of Hepatitis B Virus by Electron Microscopy, *Nature*, **386**, 88--91, 1997.

[3] R. H. Cheng, R. J. Kuhn, N. H. Olson, M. G. Rossmann, H. C. Choi, T. J. Smith, and T. S. Baker, Nucleocapsid and Glycoprotein Organization in an Enveloped Virus, *Cell*, **80**, 621--630, 1995.

[4] Conway, J.F., N. Cheng, A. Zlotnick, P.T. Wingfield, S.J. Stahl, and A.C. Steven, Visualization of a 4-helix Bundle in the Hepatitis B Virus Capsid by Cryo-electron Microscopy, *Nature*, **386**, 91--94, 1997.

[5] Cornea-Hasegan, M.A., Z. Zhang, Robert E. Lynch, Dan C. Marinescu, A. Hadfield, J.K. Muckelbauer, S. Munshi, Liang Tong, and Michael G. Rossmann, Phase Refinment and Extension by Means of Non-crystallographic Symmetry Averaging using Parallel Computers , *Acta. Cryst*, **D51**, 749--759, 1995.

[6] Crowther, R.A. and L.A. Amos, Harmonic Analysis of Electron Microscope Images with Rotational Symmetry, *Journal of Molecular Biology*, **60**, 123--130, 1971.

[7] Fuller, S.D., S.J. Butcher, R.H. Cheng, and T.S. Baker, Three-Dimensional Reconstruction of Icosahedral Particles -- The Uncommon Line, *Journal of Structural Biology*, **116**, 48--55,

1996.

[8] D.C. Marinescu, J.R. Rice, M.A. Cornea-Hasegan, R.E. Lynch, and M.G. Rossmann, Macromolecular Electron Density Averaging on Distributed Memory MIMD Systems *Concurrency: Practice and Experience*, **5**, 635--657, 1993.

[9] D.C. Marinescu, J.R. Rice, and E.M. Vavalis, Performance of Iterative Methods on Distributed Memory MIMD Systems, *Applied Numerical Mathematics*, **12**, 421--430, 1993.

[10] I. M. Boier Martin and D. C. Marinescu, Exploiting Symmetry in Parallel Computations for Structural Biology, *Lecture Notes in Computer Science*, **124**, Springer Verlag, 255--259, 1996.

[11] M. F. Moody, Image Analysis of Electron Micrographs, *Biophysical Electron Microscopy*, Academic Press, 145--285, 1990.

[12] De Rosier, D.J., Who Needs Crystals Anyway ?, *Nature*, **386**, 26--27, 1997.

[13] Smith, P.R., T.M. Peters, and R.H.T. Bates, Image Reconstruction from Finite Numbers of Projections, *Journal of Physic*, **6**, 319--381, 1973.