INSTITUTE OF PHYSICS PUBLISHING PHYSICS IN MEDICINE AND BIOLOGY

Phys. Med. Biol. **47** (2002) 2785–2795 PII: S0031-9155(02)37839-4

Internet2-based 3D PET image reconstruction using a PC cluster

D W Shattuck[1], **J Rapela**[1], **E Asma**[1], **A Chatzioannou**[2], **J Qi**[3] **and R M Leahy**[1]

[1] Signal and Image Processing Institute, University of Southern California, Los Angeles, CA 90089, USA
[2] Crump Institute for Molecular Imaging, University of California, Los Angeles CA 90095, USA
[3] Center for Functional Imaging, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA

E-mail: leahy@sipi.usc.edu

Received 22 December 2001
Published 17 July 2002
Online at stacks.iop.org/PMB/47/2785

Abstract

We describe an approach to fast iterative reconstruction from fully three-dimensional (3D) PET data using a network of PentiumIII PCs configured as a Beowulf cluster. To facilitate the use of this system, we have developed a browser-based interface using Java. The system compresses PET data on the user's machine, sends these data over a network, and instructs the PC cluster to reconstruct the image. The cluster implements a parallelized version of our preconditioned conjugate gradient method for fully 3D MAP image reconstruction. We report on the speed-up factors using the Beowulf approach and the impacts of communication latencies in the local cluster network and the network connection between the user's machine and our PC cluster.

1. Introduction

Statistically optimal algorithms for PET image reconstruction can produce significant improvements in image quality and resolution. However, the iterative methods used in such approaches can require an hour or more of computation on a single processor computer for fully 3D datasets. Substantial reductions in computation time have been achieved using the ordered subsets expectation maximization (OSEM) algorithm, which can achieve acceptable results in just a few passes through the data (Hudson and Larkin 1994). Further reductions have been achieved by converting fully 3D datasets to two-dimensional (2D) data using rebinning algorithms (e.g. Defrise *et al* 1997) and then using iterative 2D reconstruction rather than fully 3D methods (Comtat *et al* 1998, Kinahan *et al* 1996, Liu *et al* 2001). However, these speed-ups are achieved at a price: the OSEM method never optimizes the likelihood objective function and the results can be dependent on the number of subsets and the number of

iterations that are used; similarly, the rebinning methods, even if exact for true line integrals, are unable to accurately model the true physical response of the scanner. In our work (Qi *et al* 1998b, Mumcuoglu *et al* 1994) we have concentrated on using convergent algorithms to compute maximum *a posteriori* (MAP), or equivalently penalized-ML, solutions to the PET reconstruction problem. The more accurate models that we use with fully 3D datasets have been shown to improve image resolution (Qi *et al* 1998a, 1998b, Chatziioannou *et al* 2000) but inevitably lead to longer computation times. These methods can require an hour or more for 3D datasets, making the method impractical for routine use in clinical and small animal scanners. In this paper we present a distributed computing system and software designed to produce MAP reconstructions of PET data in reasonable time. We also describe our method for decoupling the reconstruction computer from the scanning facility via the Internet, allowing the cluster to serve as a remote computing resource.

Previously, our approach to reducing reconstruction time has been to use rapidly converging methods such as the preconditioned conjugate gradient method (Qi *et al* 1998b). Further reductions were obtained using symmetric multi-processing (SMP) computing. We developed a multi-threaded implementation of our reconstruction method that parallelized the optimization across multiple CPUs. In tests with a 4-processor server, we were able to achieve speed-up factors of approximately 3.4 relative to a single processor. Unfortunately, the number of processors in most entry level or midrange servers is often limited to four, and the cost relative to single or dual processor systems is very high. For this reason, we investigated the use of PC clusters, often referred to as Beowulf clusters (Sterling *et al* 1995). This allowed us to use a large number of low-cost systems to achieve a substantial speed-up relative to a single computer.

Other researchers have also developed distributed computing approaches for PET image reconstruction. Labbé *et al* (1999) developed a library for parallel and distributed PET reconstruction called PARAPET that provides parallel implementations of many functions needed for PET image reconstruction, such as forward and backprojection operators. Vollmar *et al* (2000, 2002) recently presented a cluster approach to reconstruction of images from the high-resolution CTI HRRT brain scanner. In their paper in this special issue they describe implementations using Fourier rebinning and 2D OSEM and fully 3D OSEM in a distributed Windows NT environment.

A Beowulf cluster is simply a network of workstations, typically using Unix or Linux as an operating system. For the purposes of code parallelization, the cluster is usually configured with a head-node that controls the program and a set of worker-nodes that handle processes spawned by the head-node. The difference between the cluster system and a multiple CPU server is that the former does not have shared memory; thus data must be transferred via a local ethernet between processors. This is often the bottleneck in the performance of these clusters and is of particular importance in PET image reconstruction where the datasets and image volumes are large. For this reason, many cluster implementations rely on high-speed interconnections running at 1 Gbit/s or higher. Such components can significantly increase the cost of the nodes on a system. Here we report on our progress in using a combination of multithreading and distributed computing on a Beowulf cluster consisting of nine dual processor 933 MHz Pentium III computers connected via a 100 Mbit/s switched ethernet.

A second goal of our work was to decouple the computer used for reconstruction from that used to acquire data. To do this, we developed a web-browser-based interface to our distributed computing code using Java. Thus data can be processed using the cluster from any Java-enabled computer connected to the Internet. While data transfer times may be unacceptable for standard Internet connections, the availability of higher speed Internet2 connections at many research facilities makes this approach viable. We describe and report

on an experiment we have performed by reconstructing data residing on the PET system computers in the Nuclear Medicine clinic at UCLA using the cluster at the Signal and Image Processing Institute at USC.

2. Methods

2.1. MAP image reconstruction

We use an MAP estimation algorithm to reconstruct PET images (Qi *et al* 1998b). In this approach, the data are modelled as

$$\overline{\boldsymbol{y}} = \mathbf{P}\boldsymbol{x} + \overline{\boldsymbol{r}} + \overline{\boldsymbol{s}} \tag{1}$$

where $\overline{\boldsymbol{y}}$ is the mean of the data, $\boldsymbol{x}$ is the source distribution, $\overline{\boldsymbol{r}}$ is the mean of the randoms, and $\overline{\boldsymbol{s}}$ is the mean of the scattered events. Randoms are estimated using delayed windows, while scatter is estimated using a model-based approach for the ECAT HR+ scanner (Watson 2000) and a Monte Carlo approach for the microPET P4 scanner (Holdsworth *et al* 2002). $\mathbf{P}$ is the system matrix describing the probability that an event is detected, which we factor as

$$\mathbf{P} = \mathbf{P}_{\text{norm}}\mathbf{P}_{\text{blur}}\mathbf{P}_{\text{attn}}\mathbf{P}_{\text{geom}} \tag{2}$$

where $\mathbf{P}_{\text{geom}}$ is the geometric projection matrix describing the probability that a photon pair reaches the front faces of a detector pair in the absence of attenuation and assuming perfect photon pair colinearity. $\mathbf{P}_{\text{blur}}$ models photon pair non-collinearity, inter-crystal scatter and crystal penetration, $\mathbf{P}_{\text{attn}}$ contains attenuation correction factors for each detector pair, and $\mathbf{P}_{\text{norm}}$ is a diagonal matrix containing the normalization factors.

Reconstructions are computed as the maximizer of a posterior probability equal to the sum of the log-likelihood of the data, $\boldsymbol{y}$, conditioned on the image, $\boldsymbol{x}$, and the log-prior, which has the form of a Gibbs energy function

$$\begin{aligned} L(\boldsymbol{x};\boldsymbol{y}) &= \ln p(\boldsymbol{y}|\boldsymbol{x}) + \ln p(\boldsymbol{x}) \\ &= \sum\nolimits_i \{\overline{y}_i + y_i \ln(\overline{y}_i)\} - \sum\nolimits_j \sum_{\substack{k\in\mathcal{N}_j \\ k>j}} \beta_{jk}(x_j - x_k)^2 \end{aligned} \tag{3}$$

where β_{jk} is the smoothing parameter, which may vary spatially to control the resolution properties of the reconstructed image (Qi and Leahy 2000). $\mathcal{N}_j$ is the neighbourhood of voxels about j; in our case, we use a 26 nearest-neighbour model.

As in our previous work on PET image reconstruction, we use a preconditioned conjugate-gradient algorithm for optimization. The specific algorithm we use is the Polak–Ribiere form of the conjugate gradient method. This uses an iterative update rule

$$\begin{aligned} \boldsymbol{x}^{(n+1)} &= \boldsymbol{x}^{(n)} + \alpha^{(n)}\boldsymbol{s}^{(n)} \\ \boldsymbol{s}^{(n)} &= \boldsymbol{d}^{(n)} + \gamma^{(n-1)}\boldsymbol{s}^{(n-1)} \\ \boldsymbol{d}^{(n)} &= \mathbf{C}^{(n)}\boldsymbol{g}^{(n)} \\ \gamma^{(n-1)} &= \frac{(\boldsymbol{g}^{(n)} - \boldsymbol{g}^{(n-1)})'\boldsymbol{d}^{(n)}}{\boldsymbol{g}^{(n-1)'}\boldsymbol{d}^{(n-1)}} \end{aligned} \tag{4}$$

where $\boldsymbol{x}^{(n)}$ is the image at iteration n, $\boldsymbol{s}$ is the direction of the update, $\boldsymbol{g}$ is the gradient of the cost function at $x^{(n)}$, and $\mathbf{C}^{(n)}$ is the diagonal EM-based preconditioning matrix. The gradient of the cost function (3) is the sum of the gradients of the log-prior and log-likelihood terms. The computation cost in the algorithm is dominated by computation of the gradient of the log likelihood which has the form (Qi *et al* 1998a)

$$\frac{\partial}{\partial x_{j'}} \ln p(\boldsymbol{y}|\boldsymbol{x}) = \sum_i p_{ij'} + \sum_i p_{ij'} \frac{y_i}{\sum_j p_{ij} x_j}. \tag{5}$$

In this equation, $\sum_j p_{ij} x_j$ is a forward projection, while $\sum_i p_{ij'} \frac{y_i}{\sum_j p_{ij} x_j}$ is a backprojection of the data normalized by the forward projection of the current image estimate. These terms must be recomputed during each iteration. The remaining term, $\sum_i p_{ij'}$, is computed once and stored.

The PCG algorithm is initialized with $\boldsymbol{s}^{(0)} = \boldsymbol{d}^{(0)}$ and iteratively computes the conjugate directions. It is necessary to check that $\boldsymbol{s}^{(0)}$ is an ascent direction. In the case that $\boldsymbol{s}^{(n)'}\boldsymbol{g}^{(n)} < 0$, $\boldsymbol{s}^{(n)}$ is a descent direction and the algorithm is re-initialized with $\boldsymbol{s}^{(n)} = \boldsymbol{d}^{(n)}$. The step size, $\alpha^{(n)}$, is computed at each iteration using a Newton–Raphson line search to maximize the objective function. We incorporate a positivity constraint by using a bent-line search as we described by (Qi *et al* 1998a).

2.2. Data

Here we report on application of this algorithm, using the PC cluster, to data collected in 3D mode using an ECAT HR+ whole-body scanner (CTI, Knoxville, TN) and the microPET P4 small animal scanner (Concorde Microsystems, Knoxville, TN). The data from the HR+ scanner were a standard 3D dataset rebinned with a span of 9 and a maximum ring difference of 22. There were 239 sinograms each of size 288 (elements) by 144 (angles) elements; these are sent as 16-bit integers, giving a total emission sinogram size of 20 MB. Attenuation correction requires a second sinogram of 32-bit floating point numbers with the same number of elements (40 MB), while normalization is stored in a factored form requiring less than 200 KB. Thus, the total size of the data for a single frame is on the order of 60 MB.

Data from the microPET scanner were rebinned into 703 sinograms each of size 192 (elements) by 168 (angles), for a file size of 45 MB when represented using 16-bit integers. Currently, transmission-based attenuation correction is not available on this scanner so that calculated attenuation correction factors are used; these can be calculated by the cluster. Current practice in the microPET scanner uses a set of normalization sinograms computed directly from a uniform cylinder scan. The resulting normalization factors, represented in a 32-bit floating point, require an additional 90 MB. This produces a set of data on the order of 135 MB in size. The file sizes are important when considering the impact of reconstruction via a browser over an Internet connection. The other files that are required are either small or can be stored on the cluster (such as the forward projection matrix, $\mathbf{P}$). Voxel sizes used in our reconstructions were 2.25 mm $\times$ 2.25 mm $\times$ 2.42 mm for the HR+ scanner and 0.6 mm $\times$ 0.6 mm $\times$ 0.6 mm for microPET, with images of size 128 $\times$ 128 $\times$ 63. The images are thus of size 4 MB when saved as 4-byte real (floating point) values. Transfer times for the reconstructed images back to a remote user are small compared to those for sending the data to the cluster.

2.3. Clusters and code parallelization

2.3.1. System set-up. We built a cluster consisting of one master node and eight worker nodes. Each worker node is a rackmounted dual processor Intel Pentium III 933 MHz system with 512 MB of RAM and 20 GB of disc space. The master node is also a rackmounted dual processor Intel Pentium III 933 MHz system, but has 1 GB of RAM and a 36 GB hard drive. The head node has dual network interface cards (NICs), allowing the cluster to have a private network but still be accessible from the Internet. The head node is connected by a 1 Gbit/s connection to a switch, which distributes this bandwidth to the 100 Mbit/s NICs in the worker nodes. This allows each of the worker nodes to communicate with the head node at full speed. The design of the cluster is shown in figure 1. We configured the system with

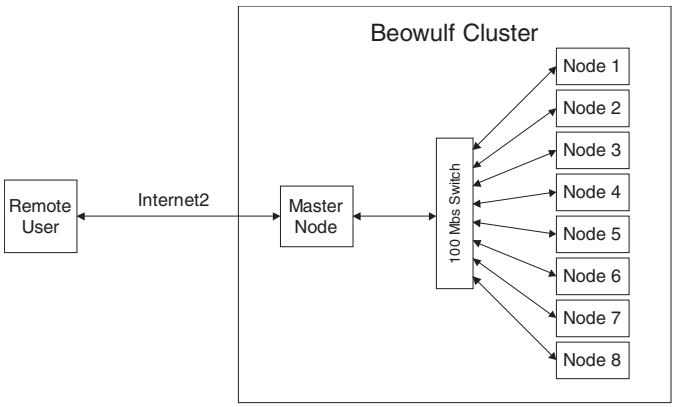

Figure 1. Architecture of the PC cluster.

the Linux operating system (RedHat v. 7.2; Linux kernel version 2.4.10). We also installed the local area multicomputing (LAM) 6.5.6 version of the message passing interface (MPI) protocol onto each node (Burns *et al* 1994). MPI is an open standard for communicating data between computer processes (Gropp *et al* 1999); LAM is an implementation for use on clustered computing systems and provides a programming environment that is portable to other architectures.

2.3.2. Code parallelization. Analysis of our algorithm's performance on a single computer revealed that two operations dominated computation: the backprojection of the sinograms into image space and the forward projection of the image into sinogram space. Computation of the gradient in (5) requires a backprojection and a forward projection during each iteration; a second forward projection may also be required during any iteration in order to maintain the positivity constraint. We distributed the processing of these two key operations across the cluster. Ignoring communication costs, we were able to achieve roughly a factor $0.75N$ speed-up on the forward and backwardprojections using N nodes of the cluster. This number is less than N because the workload is not perfectly balanced across the processors, but still represents a substantial reduction in computing time.

Were we to communicate the sinogram data during the iterations of the algorithm, the high cost of passing these results among the nodes would rapidly consume the gain in performance. Fortunately, we can decompose our problem such that the worker nodes never need to receive or transmit sinogram data once the iterations have begun. Furthermore, each node only needs to receive a subset of the sinogram data during initialization. The forward and backprojection operators are both linear transformations and are represented as a factored system matrix as described above. During backprojection, each element of the image is a function of several elements of the sinograms. Each sinogram is transformed by a block of rows of the composited system matrix to contribute to the reconstructed image. We can thus partition this transformation based on arbitrary sets of sinograms, apply the system matrix separately to these sinograms to obtain their contribution to the reconstructed image, and then sum these partial results to obtain the entire transformation. In our distributed implementation, we assign to each node a range of sinograms for which it is responsible. The node keeps updated versions of the forward projection of the current image estimate into these sinogram planes, and backprojects these into the image space when requested by the head node. The image results are sent back to the head node, where they are combined into a single image.

The forward projection problem can be decomposed into functions producing individual sinograms. However, each operation will still need to access the full image that is being forward projected. Fortunately, the communication cost of transmitting images to each node is relatively small compared to the cost of transmitting sinogram data or performing the reconstruction computation. During forward projection, the head node broadcasts the image to each node; each node is then responsible for producing the sinograms it will use during backprojection. Since the nodes generate any sinogram data they need in addition to their initial data, they do not need to communicate their sinogram results to other nodes of the cluster.

To completely eliminate the need to send full sinogram data back to the head node, we distributed some additional computation across the nodes. The overhead costs of this distributed processing are small compared to the gains from distributing the projection operations. We also use a second layer of parallelization on each node, each of which has two processors. The projection operations are again decomposed based on sinograms with two threads spawned on each node. In this case, we achieve better load balancing since the node can dynamically assign sinograms to the threads as soon as they are idle. The computation of the contribution to the cost function from the prior in (3) is also multi-threaded on the head node. This operation may be distributed to the cluster in future work; however, this would require broadcasting of image vectors and the cost of communication may outweigh the benefit of distributed processing. The projection routines are also used during initialization, so improvements to them reduce start-up costs for the algorithm. Additionally, the geometry matrices used in projection can be hundreds of megabytes in size and are needed on each node. These matrices are used repeatedly for a particular scanner and voxel size, thus we store copies of these files on the local hard drive of each node. This reduces the network burden further.

2.4. Java-browser-based interface

We developed a Java-based interface to the 3D MAP reconstruction program that allows users across the Internet to run reconstructions on our Beowulf PC cluster. The interface consists of two components: a client module and a server module. Figure 2 illustrates the architecture of the interface. The client module was implemented as a Java applet and can run on web browsers equipped with the Java Run-time Enviroment (available for most computers from http://java.sun.com). The user supplies the data files (emission file, normalization file, etc) and parameters (number of bed positions, number of frames, etc) for the MAP reconstruction and submits a reconstruction request. The files may be uploaded to the server in either ECAT Matrix7 format or as raw data files; multiple frames can be reconstructed from a single request.

Because the data sizes used in 3D PET reconstructions are large, we gave the client the capability to compress the data before uploading them to the cluster. The files are compressed using the popular Zip format. Our programs also accept pre-compressed data in gzip format, and in the future they will accept data that have been compressed using our sinogram compression methods (Asma *et al* 2001). The data size we used for HR+ reconstruction was 60 MB, while data size for the Concorde scanner was 135 MB. All HR+ files were transfered in ECAT Matrix7 format, while the Concorde data were transfered as 16-bit integers (emission data) and 32-bit floats (normalization data). The data were transferred over the network and received by a server module on the cluster.

We implemented the server module as a Java servlet that receives the data from the client and performs decompression if necessary. As part of our implementation of the server, we developed a queuing system to handle multiple requests for reconstruction. The servlet will spawn one reconstruction at a time, in the order received. Once a reconstruction has started, text messages describing the reconstruction progress are sent to the appropriate Java client,

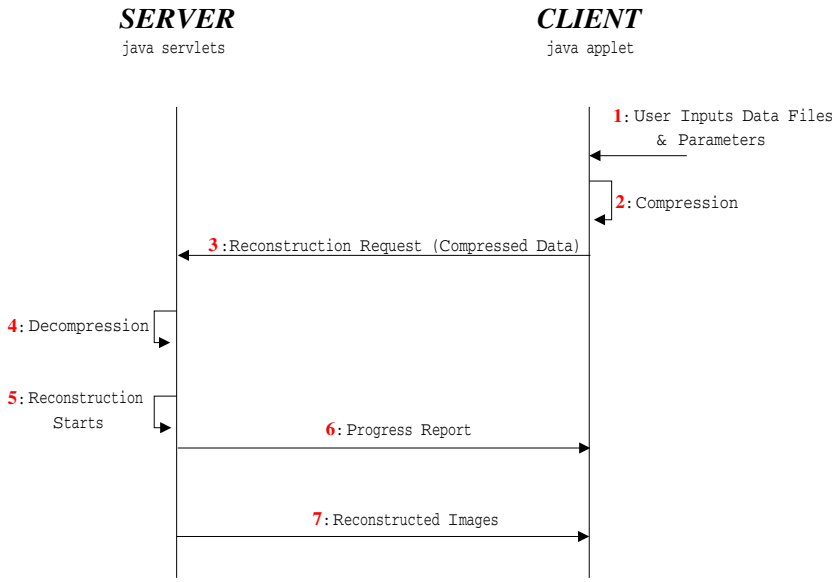

Figure 2. Architecture of the web interface to the 3D PET reconstruction program. The user supplies the data files and parameters to the java applet. If requested, the applet compresses the data files and submits a reconstruction request to the server. The server decompresses the data files if necessary, starts the reconstruction and sends the reconstruction progress to the applet, which displays it in the client browser. When the reconstruction finishes the server sends an e-mail telling the user that the data are ready for download.

which will display reconstruction progress in the specific user's browser. The server will also e-mail the user once the reconstruction has finished, providing the user with a private location from which files may be downloaded. The user may leave the website once the reconstruction request has been sent.

2.5. Network connections

Internet2 is a research and development consortium that has established high-speed connections among hundreds of US universities and companies. Internet2 is a subset of the standard Internet, where two machines that are both on Internet2 are capable of connecting at higher speeds. To evaluate performance over Internet2, the PC-cluster server was connected through a 100 Mbit/s network to the University of Southern California backbone to Internet2. Client computers containing the data were connected via gigabit connections to two sites on the UCLA computer network and one site on the same local area network (LAN) as the cluster. The link between USC and UCLA is part of the California Research and Education Network-2 (CalREN-2). CalREN-2 is a high-performance advanced-services network with a minimum communication bandwidth of 622 Mbit/s; USC and UCLA are part of an Optical Carrer level 48 (OC-48) ring with a bandwidth of 2.488 Gbit/s.

3. Results and discussion

3.1. Computational efficiency

We performed reconstructions on our cluster using different numbers of nodes to assess the benefit of using distributed processing. Figure 3 shows the performance gains achieved for

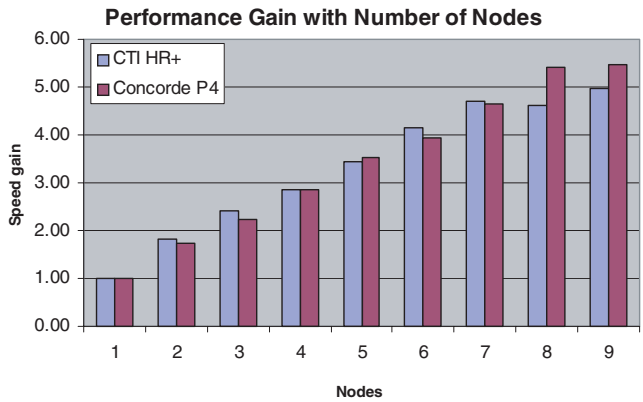

Figure 3. Speed increase in MAP PET reconstruction for different sized clusters.

reconstructions of both HR+ and P4 data; all reconstructions used 30 iterations. We observed the gains to be similar in both the initialization and main loop of the program. The chart shows that we achieved increases in processing speed greater than $N/2$, where N is the number of nodes in the cluster. The HR+ reconstruction was performed on nine nodes five times faster than the same reconstruction performed on a single node. The cluster version of the code computed a reconstruction in 19 min and 14 s, as compared to 94 min on a single node; this represents a performance gain of 5. The reconstruction of the Concorde data took 21 min and 08 s on nine nodes, compared to 1 h and 55 min on a single node. The better peformance gain of 5.5 was likely due to the larger size of the reconstruction problem.

Figure 4 shows the time spent by the head node in key components of a reconstruction iteration for the HR+ data as computed with different sized clusters. Forward and backprojection clearly dominate the computation time when the algorithm is performed on a single node. When two nodes are used, these times drop significantly. Part of this reduction is due to load imbalance, as the processing of the sinograms assigned to the head node requires less time than that required for the sinograms assigned to the other nodes. If the next computation to be done does not depend on the current results from the nodes, then the head node can proceed to its next computing task. This explains the better-than-N speed-up achieved in the forward and backprojection as the number of nodes goes from 1 to 2, while the overall performance remains slightly less than N. As the number of nodes increases, the times for these forward and backprojection are greatly reduced. When the ninth node is added to the cluster, the line search requires almost as much time as the forward and backprojection. This figure indicates that to achieve further gains by adding more nodes we must either distribute additional processing or perform better load balancing. With hand-tuned load balancing, where planes of sinogram data were assigned to nodes based on their known computation times, we were able to achieve a peformance gain of 6.5 with nine nodes. We will investigate an automatic load-balancing scheme in future work.

3.2. Remote computing and compression

We reconstructed images remotely using the Java client/server on a local area network at USC, on a fast connection between USC and UCLA, and on a slower connection between UCLA and USC. The results are summarized in table 1. Additional overhead for data transfers between the USC and UCLA sites over Internet2 was minimal. Transfer of the combined transmission, attenuation correction and normalization files in uncompressed Matrix7 format (60 MB) took

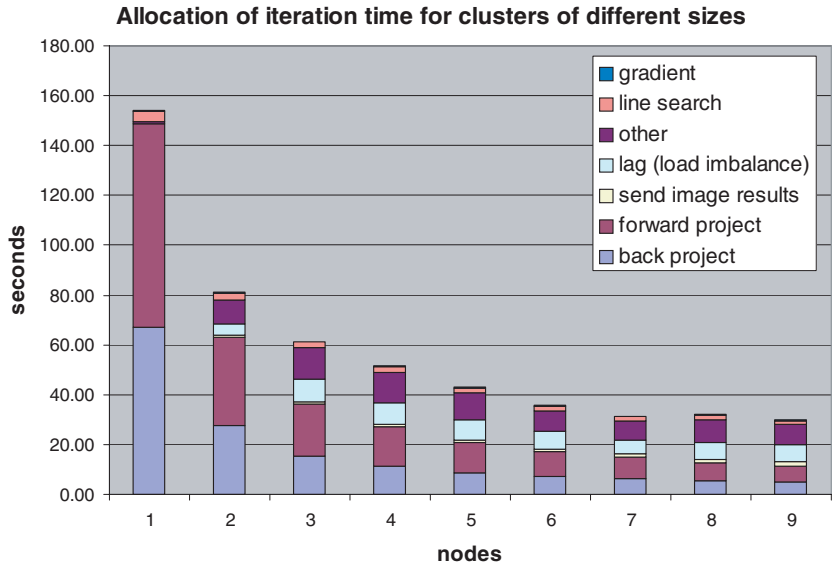

Figure 4. Usage of computing time in a single iteration of reconstruction. Forward projection lagtime is the difference between when the head node finishes its portion of the forward projection and when all nodes are finished; some of this time may be used by the head node to perform additional computation.

(This figure is in colour only in the electronic version)

Table 1. Transfer and reconstruction times for 30 iterations on the PC cluster. UCLA-I2 is a computer connected directly to the Internet2 via gigabit switching. UCLA-S is a standard connection using 100 megabit switching. USC-LAN is a computer at USC connected to the same local area network as the cluster. Note that transfer rates are highly dependent on network traffic. Compression times shown are for an Intel 933 MHz PIII Xeon client.

	ECAT HR+		Concorde P4	
Reconstruction time	18:44		21:08	
Compression type	None	Zip	None	Zip
Transfer data size	60 MB	41 MB	136 MB	36 MB
Compression time	n/a	0:42	n/a	0:39
Decompression time	n/a	0:06	n/a	0:10
Data transfer times				
USC-LAN	0:08	0:05	0:19	0:04
UCLA-S	0:27	0:18	1:05	0:15
UCLA-I2	0:10	0:05	0:20	0:05

10 s on the direct Internet2 connection, while transfer of the P4 transmission and normalization files (135 MB) took 20 s. These were comparable to the transfer rates achieved on the local area network at USC. These times are acceptable compared to reconstruction times of 18 min and 44 s and 21 min and 08 s respectively. We expect to reach higher transfer rates once certain routing issues are resolved; recent tests performed between USC in Los Angeles, CA, and a USC satellite campus in Washington, DC, have demonstrated that transfer speeds of 600 Mbit/s are currently achievable using Internet2. We can typically achieve 30–75% compression using the compression applet, which will reduce the transfer time accordingly.

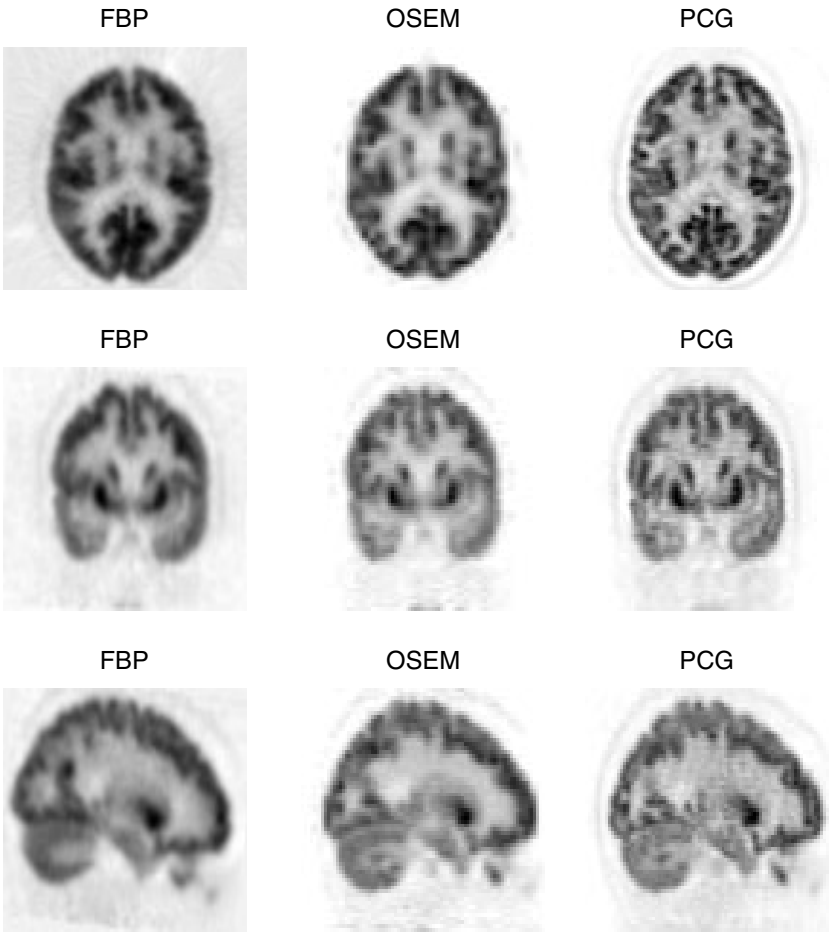

Figure 5. Reconstructions for a fully 3D FDG brain scan: ramp-filtered 3D backprojection, Fourier rebinned with 2D OSEM reconstruction (40 iterations, 8 subsets), and PCG MAP reconstruction using our cluster code (30 iterations).

However, the time taken to compress the files is approximately 40 s using a 933 MHz Intel Pentium III Xeon computer; an additional 5–10 s were required to decompress the files once they were on the cluster. This cost often exceeds the transfer time required for the uncompressed files. For systems with slower Internet connections, the trade-off between compression and transfer times will be different and use of compression may be appropriate.

An example of the advantage in image quality in using this approach is shown in figure 5. Reconstructions for a fully 3D FDG brain scan are shown using rampfiltered 3D backprojection, Fourier rebinned 2D OSEM reconstruction (40 iterations, 8 subsets), and MAP reconstruction (30 iterations) using our cluster code. Improvements in resolution resulting from the accurate system modelling used in our approach are clear in the improved definition of cortical and subcortical structures.

This preliminary study demonstrates the feasibility of using remote PC clusters for image reconstruction for PET sites with access to fast networks. Furthermore, the cluster presents a relatively low-cost approach to achieving practical reconstruction times using 3D iterative PET reconstruction.

Acknowledgments

This work was supported by grant R01-EB00363 from the National Insitute of Biomedical Imaging and Bioengineering.

References

Asma E, Shattuck D W and Leahy R M 2001 Lossless compression of list-mode 3D PET data *Proc. IEEE Nuclear Science Symposium and Medical Imaging Conference (San Diego, CA)*

Burns G, Daoud R and Vaigl J 1994 LAM: an open cluster environment for MPI *Proc. Supercomputing Symposium '94 (University of Toronto)* ed J W Ross pp 379–86

Chatziioannou A, Moore A, Annala A, Nguyen K, Leahy R and Cherry S 2000 Comparison of 3D maximum a posteriori and filtered backprojection algorithms for high resolution animal imaging with microPET *IEEE Trans. Med. Imaging* **19** 507–12

Comtat C, Kinahan P E, Defrise M, Michel C and Townsend D W 1998 Fast reconstruction of 3-D PET data with accurate statistical modeling *IEEE Trans. Nucl. Sci.* **45** 1083–9

Defrise M, Kinahan P E, Townsend D W, Michel C, Sibomana M and Newport D 1997 Exact and approximate rebinning algorithms for 3-D PET data *IEEE Trans. Med. Imaging* **16** 145–58

Gropp W, Lusk E and Skjellum A 1999 *Using MPI* 2nd ed (Cambridge, MA: MIT Press)

Holdsworth C H, Levin C S, Janecek M, Dahlbom M and Hoffman J 2002 Performance analysis of an improved 3D PET Monte Carlo simulation and scatter correction *IEEE Trans. Nucl. Sci.* **48** 83–9

Hudson H M and Larkin R S 1994 Accelerated image reconstruction using ordered subsets of projection data *IEEE Trans. Med. Imaging* **13** 601–9

Kinahan P, Michel C, Defrise M, Townsend D, Sibomana M, Lonneux M, Newport D and Luketich J 1996 Fast iterative image reconstruction of 3D PET data *Proc. IEEE Nuclear Science Symposium and Medical Imaging* pp 1918–22

Labbé C, Zaidi H, Morel C and Thielemans K 1999 An object-oriented library incorporating efficient projection/backprojection operators for volume reconstruction *Int. Meeting on Fully Three-Dimensional Image Reconstruction in Radiology and Nuclear Medicine (Egmond aan Zee)* ed F Beekman *et al* pp 137–40

Liu X, Comtat C, Michel C, Kinahan P, Defrise M and Townsend D 2001 Comparison of 3-D reconstruction with 3D-OSEM and with FORE+OSEM for PET *IEEE Trans. Med. Imaging* **20** 804–14

Mumcuoglu E U, Leahy R M, Cherry S R and Zhou Z 1994 Fast gradient-based methods for Bayesian reconstruction of transmission and emission PET images *IEEE Trans. Med. Imaging* **13** 687–701

Qi J and Leahy R 2000 Resolution and noise properties of MAP reconstructions in fully 3D PET *IEEE Trans. Med. Imaging* **19** 493–506

Qi J, Leahy R, Hsu C, Farquhar T and Cherry S 1998a Fully 3D Bayesian image reconstruction for the ECAT EXACT HR+ *IEEE Trans. Nucl. Sci.* **45** 1096–103

Qi J, Leahy R M, Cherry S R, Chatziioannou A and Farquhar T H 1998b High resolution 3D Bayesian image reconstruction using the microPET small-animal scanner *Phys. Med. Biol.* **43** 1001–3

Sterling T, Savarese D, Becker D, Fryxell B and Olson K 1995 Communication overhead for Space Science Applications on the Beowulf Parallel Workstation *Proc. 4th IEEE Symposium on High Performance Distributed Computing (HPDC) (August 1995)* pp 23–30

Vollmar S, Lercher M, Knöss C, Michel C, Wienhard K and Heiss W D 2000 BeeHive: cluster reconstruction of 3-D PET data in a Windows NT network using FORE *Proc. IEEE Med. Imaging Conf. (Lyon)*

Vollmar S, Michel C, Treffert J T, Newport D, Knöss C, Wienhard K and Heiss W D 2002 *HeinzelCluster*: accelerated reconstruction for FORE and OSEM3D *Phys. Med. Biol.* **47** 2651–8

Watson C C 2000 New, faster, image-based scatter correction for 3D PET *IEEE Trans. Nucl. Sci.* **47** 1587–94