New Techniques for Parallel Simulation of High-Temperature Superconductors

Lori Freitag
MCS Division
Argonne National Lab.
Argonne, IL 60439

Mark Jones
Computer Science Department
University of Tennessee
Knoxville, TN 37996

Paul Plassmann
MCS Division
Argonne National Lab.
Argonne, IL 60439

Abstract

In this paper we discuss several new techniques used for the simulation of high-temperature superconductors on parallel computers. We introduce an innovative methodology to study the effects of temperature fluctuations on the vortex lattice configuration of these materials. We have found that the use of uniform orthogonal meshes results in several limitations. To address these limitations, we consider nonorthogonal meshes and describe a new discrete formulation that solves the difficult problem of maintaining gauge invariance on nonorthogonal meshes. With this discretization, adaptive refinement strategies are used to concentrate grid points where error contributions are large (in this case, near vortex cores). We describe the algorithm used for the parallel implementation of this refinement strategy, and we present computational results obtained on the Intel DELTA.

1 Introduction

High-temperature superconductors have the potential to be used in a wide variety of industrial applications including generators and motors, energy storage, and magnetically levitating trains. One of the most interesting properties of these superconductors is that they allow normal and superconducting regions to coexist in the same sample. Discrete packets of external magnetic flux penetrate the sample, causing penetration regions to have the properties of a normal conductor. The rest of the superconducting sample is protected from these normal regions through the formation of vortices of superconducting electrons. It is well known that the vortices arrange themselves in a hexagonal lattice pattern to minimize the free energy in the sample when the temperature is below a critical temperature, T_c. Using the standard discretization scheme on an orthogonal mesh, we have successfully modeled vortex lattice structure in three-dimensional superconducting materials for temperatures $T < T_c$. On the Intel DELTA, this approach obtained sustained computational rates of 4.26 gigaflops with 512 processors [6].

A problem of more current interest to physicists is the study of the effect of temperature fluctuations, $T > T_c$, on the vortex lattice configuration and structure. We have developed a new methodology for studying these effects that uses derivative information already calculated in the minimization of the free energy. We have been successful in isolating the characteristic energies of the sample, but found that the use of uniform, orthogonal meshes results in several limitations, including restrictive problem sizes and symmetry constraints.

We may circumvent these problems by using adaptive mesh refinement on nonorthogonal meshes. To take advantage of the computing power and large memory capacity of state-of-the-art distributed memory architectures, we have developed the algorithms and software necessary to perform adaptive refinement on parallel computers [4]. We refine elements of the mesh according to proximity to the vortex core. In this way, grid points are concentrated where the solution changes rapidly and are relatively sparsely placed in areas where the solution is essentially constant. Thus fewer total mesh points are used without compromising accuracy in the modeling of vortex core structure.

Although this approach is efficient in its use of mesh points, the resulting discretized problem is both unstructured and dynamic. In order to achieve good load balancing and performance on parallel computers, the dynamic mesh must be partitioned (an assignment of unknowns and elements to processors) after each modification. We have developed a new geometric partitioning algorithm that strives to minimize both latency and transmission communication costs on distributed memory architectures.

An additional consideration when using nonorthogonal meshes for this problem is that standard approximation techniques fail to maintain a discrete form of the gauge invariance found in the continuous model. Therefore, we have developed a new discretization scheme that maintains gauge invariance on nonorthogonal meshes. We use asymptotic analysis to show that the free energy obtained using this approach converges to the same value obtained when using standard discretization techniques on orthogonal meshes.

The remainder of the paper is organized as follows. First we describe the equations used to model superconducting materials and present the methodology used to incorporate the effects of temperature into the model. We then give the new discrete formulation of the problem appropriate for use on nonorthogonal meshes. This discretization is used in conjunction with adaptive mesh refinement to reduce the total number of grid points required to model the vortex lattice structure. We describe the parallel implementation of the adaptive mesh refinement and partitioning algorithms. Finally, we present computational results that show the efficiency of the adaptive mesh algorithms within the framework of the superconductivity problem on the Intel DELTA.

2 High-temperature superconductors

We use the highly successful Ginzburg-Landau equations to model the vortex configurations in high-temperature superconductors. An effective approach to solving these equations numerically is a damped Newton's method. This method requires the computation of derivative information, which we use to incorporate the effects of temperature into the model.

2.1 Numerical model

To study the internal structures and lattice configurations of the vortices for temperatures $T < T_c$, we minimize the nondimensionalized Ginzburg-Landau free energy functional. The total free energy over the volume Ω is given by

$$F(\mathbf{u}) = F(\psi, \mathbf{A}) = \tag{1}$$

$$F_{\text{cond}}(\psi) + F_{\text{kin}}(\psi, \mathbf{A}) + F_{\text{fld}}(\mathbf{A}). \tag{2}$$

These three terms are generally known as the condensation, kinetic, and field energy terms and are given by the formulae

$$F_{\text{cond}} = \int_\Omega -|\psi|^2 + \frac{1}{2}|\psi|^4 d\Omega, \tag{3}$$

$$F_{\text{kin}} = \int_\Omega |(\nabla + i\mathbf{A})\,\psi|^2 d\Omega, \tag{4}$$

$$F_{\text{fld}} = \int_\Omega \kappa^2 |\nabla \times \mathbf{A}|^2 d\Omega. \tag{5}$$

The variable ψ is the complex-valued order parameter and $\mathbf{A}$ is the vector potential. The physical quantities of interest are $|\psi|^2$, the local density of superconducting electron pairs, and $\mathbf{B} = \nabla \times \mathbf{A}$, the magnetic field induced by the motion of the electron pairs through the sample. The parameter κ gives the ratio of the the characteristic length over which $|\psi|^2$ varies (the correlation length, ξ), to the characteristic length over which $\mathbf{B}$ varies (the penetration depth, λ).

An important property of this free energy functional is that its value is unchanged by a gauge transformation. That is, given $(\psi, \mathbf{A})$ and any scalar function χ, we find that the pair $(\psi', \mathbf{A}')$ given by

$$\psi' = \psi e^{i\chi} \qquad \mathbf{A}' = \mathbf{A} - \nabla\chi$$

leaves the free energy invariant. Thus, there are an infinite number of solutions to the optimization problem, which complicates the computation of a minimizer.

We have found that an effective approach to computing a minimizer $\mathbf{u}^* = (\psi^*, \mathbf{A}^*)$ of the discretized free energy functional is a damped Newton's method. Each step of Newton's method requires computation of the gradient vector, ∇F, and Hessian matrix, $\nabla^2 F$. These terms are used to compute a correction term, $\mathbf{s}_i$, in the following manner:

$$(\nabla^2 F + \gamma_i I)\mathbf{s}_i = -\nabla F$$

$$\mathbf{u}_{i+1} = \mathbf{u}_i + \alpha_i \mathbf{s}_i,$$

where α_i is computed by a line search. As a result of the gauge symmetries of the problem, the Hessian is highly singular at the solution (approximately one-fourth of the eigenvalues are zero in two dimensions), and we include the damping term, γ_i, to improve the convergence of the method as described in Garner et al. [3]. The computational kernel of this technique is the solution of the damped Newton system—a large, sparse linear system of equations. We do not explicitly invert this system but use an iterative solver to obtain an approximate (inexact) solution to the Newton system. The `BlockSolve` package developed at Argonne National Laboratory by Mark Jones and Paul Plassmann [5] is used for this purpose.

2.2 Fluctuation calculations

Given that one can compute minimizers of the free energy functional using a Newton iteration, we note

that it is possible to incorporate the effects of temperature into the model. Temperature fluctuations around the critical temperature, T_c, of a superconducting material cause the vortices to move and vibrate around the equilibrium hexagonal vortex configuration. If the temperature is high enough, the hexagonal structure is lost as the vortex lattice melts and vortices move freely in the superconducting sample. In this case, the value of the free energy functional is now dependent on $\mathbf{u}$ and T and is incorporated in the following way

$$\mathcal{F}(\mathbf{u}, T) = -Tln(Z), \quad (6)$$

$$Z = \int \mathcal{D}[\mathbf{u}(r)]e^{-\frac{F(\mathbf{u})}{T}}, \quad (7)$$

where the partition function, Z, is the integral over all possible vortex configurations in the sample multiplied by the probability that each configuration will occur. This probability is a function of $F(\mathbf{u})$ and T, where $F(\mathbf{u})$ is the Ginzburg-Landau functional evaluated at the perturbed configuration $\mathbf{u}$. It is important to note that the negative sign in the exponential term shows that the higher the free energy value of the configuration, the less likely it is to occur. That is, the most likely configurations will be those that result from small perturbations around the equilibrium lattice. Hence, in this paper we consider small perturbations of T around T_c.

Traditionally the partition function, Z, is found numerically by using Monte Carlo techniques, which require millions of evaluations of F. For the problem sizes considered here, each evaluation of F requires several minutes, so that the use of Monte Carlo techniques results in a computationally intractable problem. Instead, we use the fact that the value of the free energy functional for small perturbations, $\mathbf{p}_i$, is given by the Taylor expansion

$$F(\mathbf{u}^* + \mathbf{p}_i) = F(\mathbf{u}^*) + \nabla F \mathbf{p}_i + \mathbf{p}_i^T \nabla^2 F \mathbf{p}_i + \cdots, \quad (8)$$

where $F(\mathbf{u}^*)$ is the minimum free energy of (2)-(5) at $T \leq T_c$, the linear term is zero since $\mathbf{u}^*$ is the optimal configuration, and $\nabla^2 F$ is the Hessian of the energy functional evaluated at $\mathbf{u}^*$. If we choose $\mathbf{p}_i$ to be the orthonormal eigenvectors of the Hessian, $\mathbf{v}_i$, the expansion in (8) reduces to a sum of the associated eigenvalues λ_i

$$F(\mathbf{u}^* + \epsilon \mathbf{v}_i) = F(\mathbf{u}^*) + \epsilon^2 \lambda_i + \mathcal{O}(\epsilon^3) + \cdots. \quad (9)$$

Thus, the calculation of the free energy value for small perturbations is reduced to spectral analysis of the Hessian which must be performed only once. Although this is still a computationally intensive operation, it is tractable for the current high-performance computing architectures.

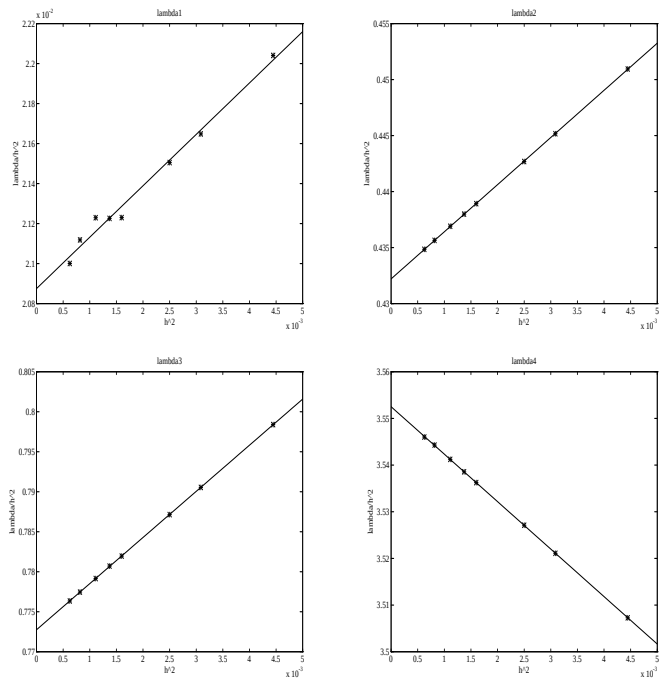

Figure 1: Asymptotic results for the smallest four nonzero eigenvalues of a single vortex superconductivity sample

We start with an orthogonal mesh and use the standard finite difference discretization of the free energy functional (2)–(5) (see, for example, [3]). To find the characteristic energies that most affect the partition function evaluation for a sample containing two vortices, we use asymptotic analysis. The results for the smallest four nonzero eigenvalues are shown in Figure 1. These are the smallest nonzero eigenvalues of the Hessian associated with each problem size. We see that the eigenvalues asymptotically approach values independent of h. In particular, the four smallest nonzero eigenvalues of this problem are

$$\begin{aligned} \lambda_1 &= .020857111, \\ \lambda_2 &= .432188615, \\ \lambda_3 &= .772728681, \\ \lambda_4 &= 3.55252043. \end{aligned}$$

Note that the eigenvalues increase several orders of magnitude rapidly. Thus, only a few of the smallest eigenvalues contribute significantly to the approximation of the partition function.

To eliminate the error associated with the discrete nature of the numerical approximation, we must plot and parameterize the phonon dispersion curves. These curves give the infinite wavelength response to temperature fluctuations and are critical to finding an accurate representation of the partition function, Z. However, to obtain even four points on the phonon dispersion curves requires solution of a sample containing 256 vortices. Problems of this size using orthogonal meshes require billions of grid points to adequately resolve vortex structure and remain computationally intractable.

3 Adaptive mesh refinement

To address the limitations of using an orthogonal mesh, we use adaptive mesh refinement to concentrate grid points around vortex cores. In this way, the total number of grid points is reduced. We use nonorthogonal meshes and introduce a new discrete formulation which maintains a discrete form of gauge invariance on these meshes. With this discretization, adaptive refinement strategies are used to concentrate grid points where error contributions are large (in this case, near vortex cores). We describe the algorithm used for the parallel implementation of this refinement strategy, and we present computational results obtained on the Intel DELTA.

3.1 Nonorthogonal discrete formulation

We would like to significantly reduce the total number of grid points required to obtain an accurate representation of the lattice configuration. This reduction would in turn allow us to study the problem sizes of interest in fluctuation calculations. To do this, we concentrate mesh grid points near the vortex core (where the solution changes rapidly) and use relatively few grid points far from the vortex cores.

On orthogonal meshes, nonuniform grid point spacing requires that constraints be placed at nonconforming nodes (nodes that lie on the edge of an element, but are not a corner vertex). These constraints unnecessarily increase the work required to solve the system and complicate implementation. Therefore, we choose to use nonorthogonal, simplicial meshes. The question now arises as to whether standard discretization techniques maintain the important property of discrete gauge invariance on these nonorthogonal meshes. We introduce the following definition which gives the requirements for a discretization to be considered gauge invariant.

Definition 1 *Let $\bar{\psi}$ and $\bar{\mathbf{A}}$ be the discrete representations of ψ and $\mathbf{A}$. In addition, let $\bar{F}$ be a discrete formulation of the free energy functional, and let $\bar{\chi}$ be any scalars defined on the same grid points as $\bar{\psi}$. Consider the transformation $\bar{\psi}' = \bar{\psi}e^{i\bar{\chi}}$. We define discrete gauge invariance to be a property of the discretization scheme that allows for a corresponding transformation of $\bar{\mathbf{A}}$ to some $\bar{\mathbf{A}}'$ such that $\bar{F}(\bar{\psi}', \bar{\mathbf{A}}') = \bar{F}(\bar{\psi}, \bar{\mathbf{A}})$.*

We found that the standard finite-difference and finite-element approximation techniques do not maintain gauge invariance on nonorthogonal meshes as defined above (see [2]). Therefore, we have developed a new discrete formulation for use on nonorthogonal meshes that ensures that a discrete version of gauge invariance is preserved. The unknowns of the new formulation are a, b, and the phase, θ, where

$$\theta = \int \mathbf{A} \cdot \mathbf{T} ds$$

is associated with the links between the grid points and $\mathbf{T}$ is the unit tangent vector such that $\mathbf{n} \times \mathbf{T}$ always points into the domain. Let the vertices of a typical triangle in the mesh be v_1, v_2, and v_3 and the links opposite each vertex be $\mathbf{l}_1$, $\mathbf{l}_2$, and $\mathbf{l}_3$. Let the area of the triangle be $A_\triangle$ and the area of the entire domain be A_Ω. We give the discretization of each of the three terms in Equation (2) that ensures that the conditions of Definition 1 are satisfied.

The condensation energy density in Equation (3) is evaluated at the vertices of the triangle and averaged to obtain a value for the entire element:

$$\bar{F}_{\text{cond}} = \frac{A_\triangle}{3A_\Omega} \sum_{i=1}^{3} \left[-(a(v_i)^2 + b(v_i)^2) + \frac{1}{2}(a(v_i)^2 + b(v_i)^2)^2 \right]. \quad (10)$$

For the field energy density (5), we use Green's theorem to obtain

$$F_{\text{fld}} = \int_\triangle \kappa^2 |\nabla \times \mathbf{A}|^2 d\triangle = \oint_{\partial\triangle} \kappa^2 |\mathbf{A} \cdot \mathbf{T}|^2 ds.$$

The discrete representation of this equation using the field variable θ is

$$\bar{F}_{\text{fld}} = \frac{1}{A_\triangle A_\Omega} \left(\kappa \sum_{i=1}^{3} \theta(\mathbf{l}_i) \right)^2. \quad (11)$$

The kinetic energy density (4) is calculated at each vertex in the triangle, and the average of these values is used to represent the energy of the element.

To maintain gauge invariance, we use a link variable formulation similar to that used in the standard finite-difference discretization. First, we rotate the element so that all values of the order parameter are in the same gauge basis. That is,

$$\hat{\psi}(v_2) = \psi(v_2)e^{-i\theta(\mathbf{l}_3)}, \quad \hat{\psi}(v_3) = \psi(v_3)e^{-i(\theta(\mathbf{l}_3)+\theta(\mathbf{l}_1))},$$

$$\hat{\psi}(v_1) = \psi(v_1)e^{-i(\theta(\mathbf{l}_3)+\theta(\mathbf{l}_1)+(\mathbf{l}_2))}.$$

In this case the gauge invariant differences along each link are given by

$$K_1 = \hat{\psi}(v_3) - \hat{\psi}(v_2), \qquad K_2 = \hat{\psi}(v_1) - \hat{\psi}(v_3),$$

$$K_3 = \hat{\psi}(v_2) - \psi(v_1).$$

The contribution to the kinetic term at the vertex v_1 is then

$$K(v_1) = K_2^2\mathbf{l}_3^2 + K_3^2\mathbf{l}_2^2 + 2\mathbf{l}_2 \cdot \mathbf{l}_3 K_2 K_3^*.$$

The cross term $2\mathbf{l}_2 \cdot \mathbf{l}_3 K_2 K_3^*$ is not required for orthogonal meshes but is incorporated here to adjust for the nonorthogonality of the triangle sides. Similar terms for the other two vertices are summed to give the kinetic energy for the element,

$$\bar{F}_{\text{kin}} = \frac{1}{12\ A_\triangle A_\Omega} \sum_{i=1}^{3} K(v_i). \tag{12}$$

To validate the new formulation, we used asymptotic analysis on a sample containing two vortices with $\kappa = 5$ to show the validity of the new discrete model. The results obtained on uniform, orthogonal meshes using finite differences were compared with those obtained on uniform triangular meshes using the new discrete formulation. As the element area $\mathcal{O}(h^2)$ approaches zero, we see in Figure 2 that condensation, kinetic, and total free-energy terms converge linearly to the same result. The methods approach the solution from different directions: finite differences from below for the condensation and total free-energy terms, the new formulation from above and vice versa for the kinetic term. The kinetic energy is highest around the vortex core, which, for the new element technique, may be located anywhere in the element, not just at grid points, as is the case in finite differences. In this case the gradient term in F_{kin} is not well approximated, and the kinetic energy term is always underestimated.

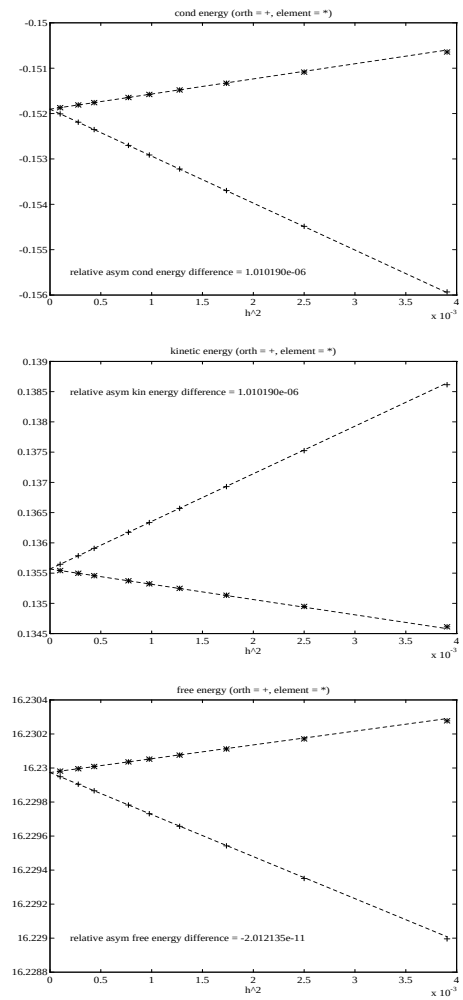

Figure 2: Asymptotic results for the condensation, kinetic terms and final free-energy value

3.2 Parallel implementation

With the new discrete formulation, we may use adaptive mesh refinement on simplicial meshes to concentrate grid points where error contributions are large. In this case, the adaptive mesh tracks vortex development and movement. In any simulation using adaptive mesh refinement techniques, maintaining mesh quality is a primary concern. As the minimum angle of the mesh decreases, the condition number of the linear system grows, making solution more difficult. As the maximum final angle of the mesh approaches π, the interpolation error in the approximate solution increases.

One refinement technique that guarantees mesh quality is the bisection algorithm of Rivara [8]. In

this algorithm, a triangle marked for refinement is divided by connecting the midpoint of the longest side to the opposite vertex. This approach creates a nonconforming point in a neighboring triangle, and the refinement is propagated until all nonconforming points are removed from the mesh (see Figure 3 for an illustration). The algorithm guarantees that mesh angles are bounded away from 0 and π if the angles in the initial mesh are. In particular, the smallest angle of the $k-th$ mesh, θ^k_{min}, is greater than or equal to one-half of the smallest angle in the initial mesh θ^0_{min} [9]. Using this result, we may also choose to bisect a triangle selectively across a smaller side if the resulting angles are greater than $\frac{1}{2}\theta^0_{min}$ and still maintain mesh quality.

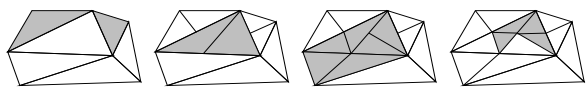

Figure 3: Propagation within the bisection algorithm

To implement the bisection algorithm on medium-grain parallel architectures such as the Intel DELTA or IBM SP1, we partition the vertices of the initial mesh across the processors, as illustrated in Figure 4. Partition boundaries are indicated by dashed lines, and the vertices and triangles owned by the center processor are indicated by the black dots and shaded triangles. In addition, each processor stores the nearest neighbor information, indicated by clear dots and triangles in the figure. Note that any given triangle is owned by only one processor even though the vertices associated with that triangle may be partitioned across processor boundaries. A processor owns a triangle if it owns two or more of the vertices of the triangle. If all of the vertices of a triangle are owned by different processors, triangle ownership is determined by a tie-breaking procedure. Any vertex or triangle created in the refinement procedure is owned by the processor that created it.

Synchronization in the parallel refinement algorithm must be managed so that there is a unique global list of vertices and element neighbor information is correct across the processors. To ensure that these conditions are satisfied, we refine independent sets of triangles in parallel. We define the independent sets in the context of the dual graph of the mesh, where the dual graph is defined to be $D = (T, F)$, where T is the set of triangles in the mesh and F is the set of links that connect two triangles if they share a common edge. We say that a triangle, t_i, is in the independent set, I_k, if for every neighboring triangle

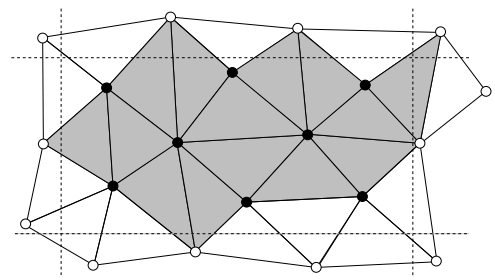

Figure 4: Partitioning of vertices and triangles across processors

$t_j \in D$

1. t_j is not marked for refinement,
2. t_j is owned by the same processor as t_i, and
3. $\rho(t_j) < \rho(t_i)$,

where $\rho(t)$ is a random number assigned to the triangle at its creation. We note that finding I_k requires no communication, since each processor stores the triangle neighbor information.

Using independent sets, we now describe an algorithm that avoids the synchronization problems mentioned above and has a provably good run time (for a complete description of this algorithm see [4]).

$k = 0$
Based on local error estimates, let Q_0 be the set of triangles initially marked for refinement
While $Q_k \neq$ **do**
  Choose $I_k \in D$ from Q_k
  Simultaneously refine the triangles in I_k
  Distribute updated element information
  $k = k + 1$
  Q_k is the set of new nonconforming triangles
  $Q_k = Q_k \cup (Q_{k-1} - I_{k-1})$
Endwhile

The only communication required in this algorithm is the distribution of updated element information to the processors and the global reduction required to check whether Q_k is empty.

As grid points are dynamically added and deleted in the mesh, we must determine good partitionings of the mesh for distributed-memory architectures. We define a good partition to be one in which the grid points are evenly distributed to the processors in such a way that interprocessor communication costs are minimized. To reduce communication costs, we must minimize the number of processor neighbors in the partition and the number links crossing the partition boundary. For uniform meshes, a good partitioning

of grid points may be determined *a priori* by the geometric domain. For unstructured adaptive meshes, however, the partitioning cannot be predetermined because it changes with each new refinement of the mesh.

Several interesting techniques may be used to determine the partitioning of an unstructured mesh. Spectral methods (see [7], for example) have the advantage of global access to information about the graph to find good separators at the cost of eigenvalue/eigenvector computation. Although the eigenvectors generally do not need to be found to much accuracy, spectral methods fail to utilize the geometric information known about the vertices of the mesh.

Geometric information is used in bisection partitioning algorithms such as the orthogonal recursive bisection (ORB) algorithm [1]. This algorithm makes an initial cut to divide the grid points in half. Orthogonal cuts are then made recursively in the new subdomains until the grid points are evenly distributed among the processors. Although this algorithm obtains good load balancing and is inexpensive to compute, it ignores the communication minimization problem. Long, thin partitions may be created that have a high ratio of links crossing the partition boundaries to total number of links in the partition.

To address this problem, we have developed a modification of ORB which we call the unbalanced recursive bisection (URB) algorithm. Instead of dividing the unknowns in half, we choose the cut that minimizes partition aspect ratio and divides the unknowns into $\frac{nk}{P}$ and $\frac{n(P-k)}{P}$ sized groups, where n is the total number of unknowns, P is the number of processors, and $k \in \{1, 2, ..., P-1\}$. This algorithm leads to an even distribution of grid points with more balanced partition aspect ratios. This tends to minimize the communication costs in two ways. First, and most important, partitions with good aspect ratios (close to one) tend to have fewer partition neighbors and hence fewer messages to send. Second, the percentage of mesh links crossing the partition boundary to the total number of links in the nearly square partitions is small compared with the long, thin partitions generated by the ORB algorithm. Thus, the ratio of computation to communication is increased compared with the ORB algorithm, while execution time to compute the partition is significantly less than for spectral techniques.

4 Computational results

To adaptively refine the mesh around vortex core singularities, we used the following refinement rule: a triangle, t_j, is refined if

$$\min\left(|\psi(v_i)|^2 \cdot A_\triangle\right) < \epsilon_T,$$

where the ϵ_T is a user-defined tolerance. The new values of a and b are obtained by linear interpolation at the new grid point. The new phase, θ_{new}, is chosen so that the magnetic flux density in the two new triangles is equal to the original flux density.

We now demonstrate the efficiency and scalability of the refinement and partitioning algorithms within the framework of the superconductivity problem. Because the current implementation of the adaptive refinement algorithms allows data to be associated with vertices only, we have used a preliminary formulation of the finite element given previously. However, the computational results presented here would be quantitatively the same for the revised element. The results of four typical runs are shown in the table below where P gives the number of processors and E indicates the number of triangular elements in the final solution mesh. The number of vortices in each sample are 32, 48, 64, and 72 for 16, 32, 64, and 128 processors, respectively. Thus, the problem size increases in proportion to the number of processors used. The amount of time required for refinement and partitioning is given as a percentage of total solution time. These operations require less than one percent of the execution time in all cases, and the solution of the linear systems dominates the cost of the calculation.

P	E	Percent Refine Time	Percent Partition Time	Percent Solution Time
16	30484	.229	.193	69.5
32	48416	.091	.117	86.3
64	111660	.087	.167	88.8
128	196494	.181	.452	86.0

Statistics on the partitions generated by the new geometric partitioning algorithm, URB, are given in the following table. The average aspect ratio for the partitions is less than two in all cases, and the maximum aspect ratio is less than 3.6. These result in a partition quotient graph whose average degree is between five and six, which corresponds to an average of five to six messages sent per processor to transfer nearest neighbor information. Finally, to estimate the amount of data that must be transferred between processors, we consider the percentage of edges that cross

partition boundaries to the total number of edges in the partition. This number is less than 15 percent in all cases.

P	Avg. Graph Degree	Max. Graph Degree	Avg. Aspect Ratio	Max. Aspect Ratio	Percent Cross Edges
16	5.31	7.00	1.47	2.88	6.72
32	5.40	8.00	1.89	3.55	8.32
64	5.64	8.00	1.34	2.49	10.0
128	5.71	9.00	1.81	3.55	13.7

The final triangular mesh of a 32 vortex problem is shown in Figure 5. Vortex cores are indicated by the location of the heavily refined areas of the mesh. This problem was run on 64 processors of the Intel DELTA, and partitions are indicated by the numbered boxes. The partitions tend to split the vortex cores to evenly distribute grid points and are nearly square in most cases. As we refine the mesh around vortex cores, the fact that the kinetic term is approaching the asymptotic result from below causes the vortex to drift toward regions containing larger mesh elements. We are currently working to eliminate this problem and are temporarily using Gaussian well pinning sites to fix vortex position.

Acknowledgments

This work was supported by the Office of Scientific Computing, U.S. Department of Energy, under Contract W-31-109-Eng-38.

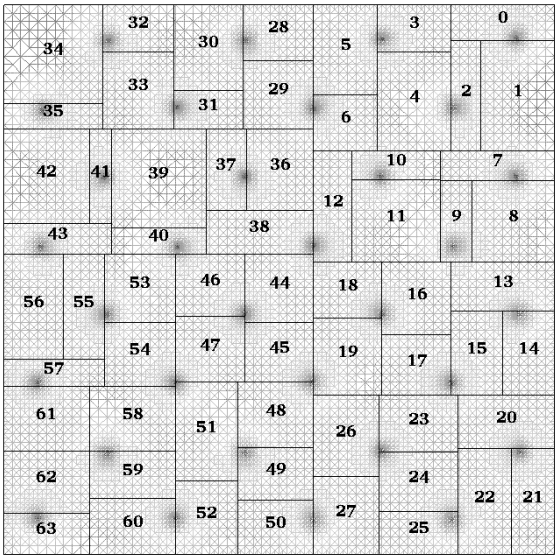

Figure 5: Results of a sample containing 32 vortices pinned to a square lattice configuration on 64 processors of the Intel DELTA

References

[1] M. Berger and S. Bokhari. A partitioning strategy for nonuniform problems on multiprocessors. *IEEE Transactions on Computers*, C-36(5):570–580, 1987.

[2] Lori A. Freitag, Mark T. Jones, and Paul E. Plassmann. New advances in the modeling of high-temperature superconductors. In *1994 International Simulation Conference, "Grand Challenges in Computer Simulation"*, La Jolla, California, April 11–15, 1994.

[3] J. Garner, M. Spanbauer, R. Benedek, K. Strandburg, S. Wright, and P. Plassmann. Critical fields of Josephson-coupled superconducting multilayers. *Physical Review B*, 45:7973–7983, April 1992.

[4] Mark T. Jones and Paul E. Plassmann. Parallel algorithms for the adaptive refinement and partitioning of unstructured meshes. In *Scalable High Performance Computing Conference*, Knoxville,Tennessee, May 1994.

[5] Mark T. Jones and Paul E. Plassmann. BlockSolve v1.0: Scalable library software for the parallel solution of sparse linear systems. ANL Report ANL-92/46, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, Ill., 1992.

[6] Mark T. Jones and Paul E. Plassmann. Computation of equilibrium vortex structures for type-II superconductors. *The International Journal of Supercomputer Applications*, 7(2), 1993.

[7] Alex Pothen, Horst Simon, and Kang-Pu Liou. Partitioning sparse matrices with eigenvectors of graphs. *SIAM Journal on Matrix Analysis*, 11:430–452, 1990.

[8] M. Rivara. Mesh refinement processes based on the generalized bisection of simplices. *SIAM Journal on Numerical Analysis*, 21:604–613, 1984.

[9] I. Rosenberg and F. Stenger. A lower bound on the angles of triangles constructed by bisecting the longest side. *Mathematics of Computation*, 29:390–395, 1975.