A parallel domain decomposition method for parabolic partial differential equations

M. S. Ingber¹, C. C. Schmidt², J. A. Tanski², J. Phillips²
¹Department of Mechanical Engineering, University of New Mexico, U.S.A.
²Engineering Science and Applications Division, Los Alamos National Laboratory, Los Alamos, New Mexico, U.S.A.

Abstract

A parallel domain decomposition method is developed for the solution of three-dimensional parabolic partial differential equations. Within each subdomain, time is discretized using the generalized trapezoidal rule. The resulting modified Helmholtz equation is solved using the particular solution boundary element method which is a variant of the dual reciprocity method. Interfacial conditions between subdomains are satisfied using a Schwarz Neumann-Neumann iteration scheme. Outside of the first time step where zero initial flux is assumed on all interfacial boundaries, the initial estimates for the interfacial flux is given from the converged solutions from the previous step. This significantly reduces the number of iterations required to meet the convergence criterion.

1 Introduction

The boundary element method (BEM) has become an attractive alternative technique to domain methods such as the finite difference and finite element methods for solving partial differential equations. The method has proven to be particularly conducive for a wide class of linear elliptic boundary value problems because of the inherent reduction in the dimensionality of the problem. In fact, in many cases, the BEM requires only the discretization of the boundary of the domain. This can be particularly advantageous in a design setting where the creation of a discretized
representation of the solid model often represents the major portion of the overall effort. Unfortunately, this advantage is generally lost for nonelliptic problems. Nevertheless, the BEM has been successfully applied to a variety of parabolic and hyperbolic partial differential equations.

In this research, a BEM based on discretization in time [3] is used to solve parabolic differential equations. The advantage of the current approach is that the boundary element analysis can be reduced to essentially solving a sequence of Poisson problems which can be effectively done using a variety of dual reciprocity or particular solution methods [7, 4].

Boundary element methods using discretization in time to solve parabolic differential equations can become memory and CPU intensive for large problems. One method of reducing the CPU requirements is to employ a domain decomposition method. Domain decomposition for boundary element methods have been studied by Kamiya et al. [5], Davies and Mushtaq [1], and Mai-Duy et al. [6]. However, all of these previous BEM investigations have dealt with stationary elliptic problems. In the current research, a parallel domain decomposition method based on the Schwarz Neumann-Neumann iteration to update the interfacial boundary conditions is used. The accuracy and parallel efficiency of the domain decomposition BEM is demonstrated through a benchmark problem.

2 Numerical Formulation

The model parabolic equation is given by heat conduction with constant conductivity $k$, density $\rho$, and specific heat $c_p$ in a three-dimensional domain $\Omega$ bounded by the surface $\Gamma$. The governing equation is given by

$$\rho c_p \frac{\partial u}{\partial t} + g = k \nabla^2 u$$

where $u$ is the temperature and $g$ represents internal heat generation. The boundary conditions for this problem can be any combination of the following types:

**Prescribed temperature**: $u = \bar{u}$

**Prescribed flux**: $q = -\bar{q}$

**Convection**: $q = -h(u - u_{\infty})$ (2)

where $q$ is the heat flux defined by $q = k \frac{\partial u}{\partial n}$, $n$ is the outward unit normal vector, $h$ is the heat transfer coefficient, and $u_{\infty}$ is the ambient temperature. The initial condition is given by

$$u(\bar{x}, 0) = f(\bar{x})$$

(3)

The method of discretization in time is employed whereby the spatial variables are discretized using the boundary element method and time is
discretized using the finite difference method. In particular, the generalized trapezoidal method (θ-method) is used to approximate the time derivative as

\[
\frac{1}{\Delta t} \left( \frac{u^n - u^{n-1}}{\alpha} \right) = \theta(\nabla^2 u^n - g^n / k) + (1 - \theta)(\nabla^2 u^{n-1} - g^{n-1} / k) \tag{4}
\]

where the diffusivity, \( \alpha \), is defined by \( \alpha = k / \rho c_p \), \( \Delta t \) is the time step, \( u^n = u(\vec{x}, n\Delta t) \), and \( g^n = g(\vec{x}, n\Delta t) \). Defining

\[ v^n = u^n - \frac{\theta - 1}{\theta} u^{n-1} \tag{5} \]

the above equation can be rearranged as

\[
\nabla^2 v^n = \frac{v^n}{\theta \alpha \Delta t} - \frac{1}{\theta^2 \alpha \Delta t} u^{n-1} + \frac{(1 - \theta)}{\theta k} g^{n-1} + g^n / k \tag{6}
\]

In the current BEM approach, the right-hand side of (6) can be considered as a "generalized" forcing function. Following standard formulation procedures for the BEM, \( v^n \) may be represented in terms of a boundary integral as follows:

\[
\eta(\vec{x}) v^n(\vec{x}) = \int_{\Gamma} \left[ G'(\vec{x}, \vec{\xi}) v^n(\vec{\xi}) - G(\vec{x}, \vec{\xi}) v^n(\vec{\xi}) \right] d\Gamma(\vec{\xi})
+ \int_{\Omega} \left[ \frac{v^n(\vec{\xi})}{\theta \alpha \Delta t} - \frac{1}{\theta^2 \alpha \Delta t} u^{n-1}(\vec{\xi}) + \frac{(1 - \theta)}{\theta k} g^{n-1}(\vec{\xi}) + g^n(\vec{\xi}) / k \right] G(\vec{x}, \vec{\xi}) d\Omega(\vec{\xi}) \tag{7}
\]

where \( G(\vec{x}, \vec{\xi}) \) is the Green's function given by \( G(\vec{x}, \vec{\xi}) = \frac{1}{|\vec{x} - \vec{\xi}|} \), the prime denotes the derivative in the direction of the outward normal to the boundary at the point \( \vec{\xi} \), and the coefficient \( \eta(\vec{x}) \) can be determined from the integral

\[
\eta(\vec{x}) = \int_{\Gamma} G'(\vec{x}, \vec{\xi}) d\Gamma(\vec{\xi}) \tag{8}
\]

A particular solution method is employed to eliminate the domain integral in Eq. 7. The particular solution method is complicated by the fact that \( v^n(\vec{\xi}) \) appears in the domain integral. For brevity, the details of the particular solution BEM are omitted.

In the parallel domain decomposition method, the three-dimensional domain \( \Omega \) is subdivided into multiple subdomains. At each time step, neither the potential \( u \) nor flux \( q \) is known along the interfaces of the subdomains. In general, there are three iterative approaches for determining the interfacial potential and flux [5]. In Uzawa's method, the potential is assumed on all interfacial boundaries and the potential is iteratively updated, in part, by considering the resulting mismatch in flux across the interfaces. In the Schwarz Dirichlet-Neumann method, the potential is assumed on one side of an interface while the flux is assumed on the other side
of the interface. Here, the interfacial potential is updated by considering the resulting mismatch in interfacial potential. In the Schwarz Neumann-Neumann method, the flux is assumed on all interfacial boundaries and the flux is again updated, in part, by considering the mismatch in interfacial potential.

The Schwarz Neumann-Neumann method is chosen in the current research for a technical reason. The boundary element method implementation uses double nodes along subdomain edges and triple nodes at corners [8]. In order to avoid having to perform off-functional node collocation, Dirichlet conditions cannot be specified at two or more coincident nodes along an interfacial edge or corner. In order to avoid this situation, the Schwarz Neumann-Neumann method is chosen since all assumed interfacial boundary conditions specify an assumed value for the flux.

Consider an interface between subdomain \( i \) and subdomain \( j \). At the initial time step, all interfacial fluxes are assumed to be zero. That is,

\[
q_i^{0,0} = 0 \quad q_j^{0,0} = 0
\]

where the subscript indicates the iteration number, the first superscript indicates the domain number, and the second superscript indicates the time step number. After the first time step, the initial interfacial fluxes are given by the converged values from the previous time step. That is,

\[
q_i^{0,n} = q_i^{0,n-1} \quad q_j^{0,n} = q_j^{0,n-1}
\]

where \( q_i^{0,n-1} \) represents the converged value of the flux on the \( i \)-subdomain side of the interface at the \( n-1 \)-time step.

The assumed interfacial conditions are modified using

\[
q_m^{i,n} = q_m^{i,n-1} + \beta(u_m^{i,n} - u_m^{j,n})
\]

\[
q_m^{j,n} = -q_m^{i,n}
\]

where \( \beta \) is a positive relaxation parameter. The chosen value of \( \beta \) will affect the convergence rate as discussed in the next section.

The parallel implementation of the algorithm is based on the Single Program, Multiple Data (SPMD) paradigm with Message Passing Interface (MPI) for parallel communication. Each subdomain is assigned to a processor. Although theoretically, any processor could accommodate more than one subdomain, in the current implementation, only one subdomain is allowed per processor.

Each boundary element analysis performed for each subdomain on separate processors is independent and does not require any communications. After solving for the unknown boundary data and interior unknowns, an MPI-allgather of these unknowns is performed. The allgather is a global MPI operation in which all unknowns are collected on all processors. This is a potentially time consuming communications operation but was required
since one-sided communications such as "put"s and "get"s as contained in
the MPI2 specifications are not generally available. The one-sided commu-
nications would obviate the need for the global allgather.

With all boundary and interior data now available on all processors,
a convergence criterion can be set. The convergence criterion used in this
research is given by
\[ \max | u_m^{k,n} - u_m^{l,n} | < \epsilon \]  \hspace{1cm} (13)
where the maximum is taken over all interfacial nodes and \( \epsilon \) is a prescribed
tolerance. If the criterion is met, the converged values of the flux are used
as the initial estimates for the next time step. If not, the interfacial fluxes
are updated for the next iteration according to (11) and (12). The only
other communication required outside of the allgather discussed above is an
MPIAllreduce at the end of each iteration step. Each interface is considered
either a master or slave surface. The convergence criterion is checked on
all master surfaces, and the processor will set the convergence flag to 1
if all its master surfaces satisfy the convergence criterion or 0 otherwise.
The MPIAllreduce takes the minimum of all convergence flags from each
processor and stores the result in the convergence flag variable on each
processor. At the end of the MPIAllreduce, all processors have either a 1
or a 0 stored in the convergence flag variable, and hence, each processor will
either go to the next iteration step or the next time step.

3 Benchmark Demonstration Problem

The governing equation for the benchmark problem is given by
\[ \frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} + g(x, y, z, t) \]
to be solved in the domain \( 0 \leq x, y, z \leq 1 \). The forcing function is given by
\[ g(x, y, z, t) = 3 \sin x \sin y \sin z(3 \sin t + \cos t) \]
The initial and boundary conditions are appropriate to the solution
\[ u = 1 + 3 \sin x \sin y \sin z \sin t \]
The dependent variable \( u \) is prescribed on the face \( x = 1 \) and the flux \( q \) is
prescribed on all other faces. This problem is of interest since the boundary
conditions and the forcing function are time dependent.

The unit cube was subdivided into either 8, 27, or 64 cubic subdomains
of equal size. Within each subdomain, the mesh included 294 boundary
element nodes, 54 boundary elements, and 125 centers of the radial basis
functions used to determine the approximate particular solution. A value of
\( \theta = 0.5 \) is chosen for the time discretization (Crank-Nicolson scheme) since
this results in a second-order accurate truncation error [2].
Results at the interior point \( x = 0.75, \ y = 0.75, \ z = 0.75 \) for the 8- and 27-subdomain discretizations are shown in Fig. 1. Both discretizations provide excellent results with the largest pointwise errors on the order of 0.5%.

![Graph showing comparison between 8 subdomain BEM, 27 subdomain BEM, and analytic results.](image)

Figure 1: Comparison at the point \( x = y = z = 0.75 \) between the 8 subdomain BEM, 27 subdomain BEM, and analytic results.

The convergence characteristics of the Schwarz Neumann-Neumann iteration method are shown in Figs. 2 to 4. In Fig. 2, the minimum and maximum number of iterations to satisfy the convergence criterion (13) is shown for \( \epsilon = 1.0E-4 \) and \( \epsilon = 1.0E-6 \). These results were generated using the 8 subdomain discretization, and the minimum and maximum number of iterations were taken over the first 40 time steps. It is interesting to observe that overrelaxation of the iteration, that is choosing \( \beta \approx 4.0 \), minimizes the overall iteration count. This was unexpected since Kamiya et al. [5] had chosen \( \beta = 0.2 \) for their 2D stationary problem. In fact, choosing \( \beta \) between 2 and 4 provides relatively fast convergence. However, for \( \beta \) greater than approximately 4.5, the Schwarz Neumann-Neumann method ceases to converge for the benchmark problem.

As discussed in the previous section, for the initial time step, all inter-
facial fluxes are initially assumed to be zero. After the first time step, the initial interfacial flux at the new time step is chosen to be the converged interfacial flux from the previous time step. With better initial estimates, the iteration count at a given time step can be reduced. This reduction in the number of iterations is shown in Fig. 3, again for the 8 subdomain case and $\epsilon = 1E - 6$.

A plot of the maximum residual for the 8-, 27- and 64-subdomain cases as a function of the iteration number at the initial time step is shown in Fig. 4 for $\beta = 4.0$. It is seen in the figure that the iteration count increases sublinearly with increasing number of subdomains. Also as seen in the figure, the reduction in maximum residual monotonically decreases with each iteration step.

![Figure 2: Minimum and maximum number of iterations for the subdomain method to converge as a function of $\beta$ using 8 subdomains.](image)

The three discretizations used in the benchmark problem provide an excellent means of measuring the scaled parallel efficiency of the BEM subdomain method since each subdomain contains the exact same number of nodes and elements. Therefore, each processor should be using approximately the same amount of memory and performing approximately the
Figure 3: The number of iterations for convergence as a function of the time step using 8 subdomains and $\beta = 4.0$.

The same number of operations. The total CPU time and the CPU time per iteration for the 8-, 27-, and 64-subdomain cases is shown in Table 1. Again, as previously seen for the initial time step (Fig. 4), the total number of iterations over 1000 time steps is seen to grow sublinearly in the table. However, the CPU time per iteration is essentially constant which shows an almost perfect scaled parallel efficiency. Essentially, this indicates that the communication costs are insignificant in comparison to the computational costs.

<table>
<thead>
<tr>
<th>Number of Subdomains</th>
<th>Total Number of Iterations</th>
<th>Total CPU Time</th>
<th>CPU Seconds per Iteration</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>31246</td>
<td>1475</td>
<td>0.0472</td>
</tr>
<tr>
<td>27</td>
<td>44057</td>
<td>1765</td>
<td>0.0401</td>
</tr>
<tr>
<td>64</td>
<td>56054</td>
<td>2346</td>
<td>0.0419</td>
</tr>
</tbody>
</table>

Table 1: CPU times using $\beta = 3.0$ and $\epsilon = 1.0E - 6$. 
Figure 4: The maximum residual as a function of the iteration number for the initial time step for 3 different discretizations and $\beta = 4.0$.

4 Conclusions

A parallel domain decomposition boundary element method has been developed for parabolic differential equations. Fluxes are assumed on all interfaces at the beginning of each time step. At the first time step, the fluxes are assumed to be zero, while at all subsequent time steps, the fluxes are initially assumed to be given by the converged values from the previous time step. The Schwarz Neumann-Neumann iteration is used to determine the interfacial fluxes and potentials.

Numerical experiment shows that overrelaxation of the iteration scheme, that is, choosing $\beta \approx 4.0$ yields the fastest convergence rate. The mismatch in calculated potential along the interfaces decreases monotonically. Further, the iteration count is seen to go down with time indicating the efficiency of starting from the previous time step’s converged results for the interfacial flux.

The parallel strategy is based on assigning each subdomain a unique processor. Only interfacial data needs to be communicated across proces-
provides a very effective parallel strategy. For moderately sized problems, the communication time is small compared to CPU time. Hence, the scaled parallel efficiency is nearly perfect. Further, the number of iterations per time step is seen to grow very slowly with increasing number of subdomains. Therefore, domain decomposition provides a very effective parallel strategy.

5 Acknowledgment

This work was supported by Los Alamos National Laboratory for the U.S. Department of Energy under contract DE-AC04-94AL85000.

References


