TACC Technical Report TR-11-02

The Spike factorization as domain decomposition method; equivalent and variant approaches

Victor Eijkhout*, Robert van de Geijn†

July 12, 2011

* Texas Advanced Computing Center, The University of Texas at Austin
† Computer Science Department, The University of Texas at Austin

Abstract

In this paper we present the Spike algorithm of Sameh and Polizzi in the context of domain decomposition methods. We present several variants that differ in their treatment of the separators, showing that one of these is equivalent to the Spike algorithm.

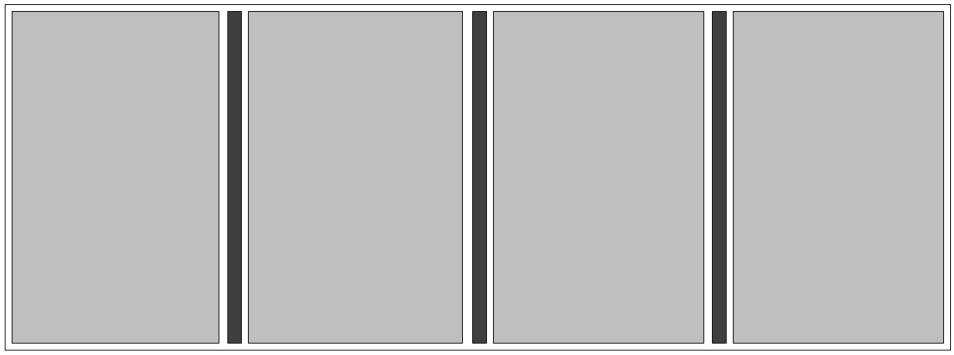

Figure 1: A one-dimensionally partitioned domain with 4 subdomains and 3 separators

1 Introduction

The parallel solution of linear systems has a long history, spanning both direct and iterative methods. While direct methods exist that have great generality, here we consider a subcase of practical importance, that of banded matrices. We note that many PDE problems naturally give rise to banded systems, given a large enough bandwidth.

For any banded matrix, we can impose a block structure such that the matrix is block tridiagonal. This structure gives each processor a contiguous block row of the matrix; we assume that the number of processors is low enough that the the part owned by any processor comprises one or more of the blocks that define the block tridiagional structure.

In this paper we present a number of variants on the Spike factorization of Polizzi and Sameh [7], but going back to Sameh and Kuck [8]. Instead of the customary algebraic presentation we view this algorithm as a domain decomposition method, where each processor corresponds to a subdomain, and the problem variables are divided in interior regions and separators. We will do a cost analysis for the case where the algorithm is applied to a finite element type matrix. Note that our analysis is only in terms of flop counting; in practice the merits of the Spike algorithm and other banded solvers are determined to a large degree by memory access patterns and other considerations related to computer architecture. The Spike algorithm is then seen to be slightly more expensive than a regular domain decomposition.

Throughout this paper, we will discuss the 1D domain decomposition model problem, pictured in figure1. This leads to a matrix of the form:

$$
\begin{array}{|c|c|c|c|c|c|}
\hline
* & & & & & \cdots \\
 & * & * & * & & \\
\hline
 & * & & & & \\
 & & * & * & & \cdots \\
\hline
 & & & * & * & \\
\hline
 & & & * & & \ddots \\
\hline
 & & & & \ddots & \ddots \\
\hline
\end{array}
\qquad (1)
$$

where the large blocks correspond to subdomains and the small ones to separators.

However, this is only for ease of analysis; in practice the only requirement for applicability of our ideas is that the matrix is partitioned with an alternating sequence of separators and subdomain interiors.

We will say that this matrix has block dimension N, where each block comprises a subdomain and a separator. For a model cost analysis, we assume that each subdomain interior consists of m lines of size n each, and that the matrix has a typical sparsity pattern based on some finite difference or finite element scheme. Thus, in the natural ordering, the matrix has dimension $N \times (m+1) \times n \approx Nmn$ and halfbandwith n. The cost of a sequential factorization is then $Nmn \cdot n^2$ muladds and the cost of solving a system with the resulting LU factorization $Nmn \cdot n$ muladds.

2 Single separator case

We will now describe the factorization of the matrix of equation 1. The first steps of the factorization are in parallel over the subdomains; we will illustrate the factorization by considering a subdomain with the two surrounding separators.

$$
A = \begin{array}{|c|c|c|}
\hline
A_{i-1i-1} & A_{i-1i} & \\
\hline
A_{ii-1} & A_{ii} & A_{ii+1} \\
\hline
 & A_{i+1i} & A_{i+1i+1} \\
\hline
\end{array}, \qquad A_{ii-1} = \begin{pmatrix} * \\ 0 \\ \vdots \\ 0 \end{pmatrix}, \quad A_{ii+1} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ * \end{pmatrix}
$$

The matrix blocks are of dimensions

$$\begin{cases} A_{ii} & \text{size } mn, \text{ halfbandwidth } n \\ A_{ii-1}, A_{ii+1} & \text{size } n, \text{ halfbandwidth } O(1) \end{cases}$$

We make an LU factorization of the large subdomain diagonal blocks.

$$L^{-1}A = \begin{array}{|c|c|c|}\hline 1 & & \\ \hline & L_{ii} & \\ \hline & & 1 \\ \hline \end{array}^{-1} \quad A = \begin{array}{|c|c|c|}\hline A_{i-1i-1} & A_{i-1i} & \\ \hline L_{ii}^{-1}A_{ii-1} & U_{ii} & L_{ii}^{-1}A_{ii+1} \\ \hline & A_{i+1i} & A_{i+1i+1} \\ \hline \end{array}$$

Next the U factor:

$$\begin{array}{|c|c|c|}\hline 1 & & \\ \hline & U_{ii} & \\ \hline & & 1 \\ \hline \end{array}^{-1} \quad (L^{-1}A) = \begin{array}{|c|c|c|}\hline A_{i-1i-1} & A_{i-1i} & \\ \hline A_{ii}^{-1}A_{ii-1} & I & A_{ii}^{-1}A_{ii+1} \\ \hline & A_{i+1i} & A_{i+1i+1} \\ \hline \end{array}$$

Next we left-multiply in parallel by a matrix T to eliminate the connection between the subdomain interior and the separators:

$$\begin{array}{|c|c|c|}\hline I & A_{i-1i} & \\ \hline & I & \\ \hline & A_{i+1i} & I \\ \hline \end{array}^{-1} \quad U^{-1}(L^{-1}A) = \begin{array}{|c|c|c|}\hline \begin{array}{c} A_{i-1i-1} \\ -A_{i-1i}A_{ii}^{-1}A_{ii-1} \end{array} & \emptyset & -A_{i-1i}A_{ii}^{-1}A_{ii+1} \\ \hline -A_{ii}^{-1}A_{ii-1} & I & -A_{ii}^{-1}A_{ii+1} \\ \hline -A_{i+1i}A_{ii}^{-1}A_{i-1} & \emptyset & \begin{array}{c} A_{i+1i+1} \\ -A_{i+1i}A_{ii}^{-1}A_{ii+1} \end{array} \\ \hline \end{array} \tag{2}$$

noting that the cost of forming the various products is limited to $O(n^3)$ since A_{i-1i}, A_{i+1i} are of size $n \times mn$, but have only one nonzero $n \times n$ block.

We now have a factorization $A = LUTS$, where S is called a 'Spike' matrix after the dense columns flanking the large identity blocks. Note that the spikes need not be stored explicitly: we can multiply by $A_{ii}^{-1}A_{ii-1}$ by solving a linear system with A_{ii} and multiplying by A_{ii-1}.

The cost analysis of this factorization is as follows:

- The A_{ii} blocks that describe the subdomain interior are of size mn with halfbandwidth n, so factoring them takes mn^2 muladds, ignoring lower order terms: each elimination step adds a row to the n rows below it.
  Solving a system with the matrix A_{ii} takes $2mn^2$ operations: in both the forward and backward solve each element of the factors is touched once, and the factors have size mn and bandwidth n. Note that the factors are dense inside the band, unlike the original matrix.
- We have a choice whether to store the blocks $A_{ii}^{-1}A_{ii\pm1}$ explicitly or to leave them implicitly defined. They are of size $mn \times n$ and fully dense, so storing them explicitly would double the storage of a dense banded matrix, or exceed the storage of a sparse band matrix A_{ii} of the interior by a factor $O(n)$.
- Forming $A_{ii}^{-1}A_{ii\pm1}$ involves solving n linear systems, for a total cost of $2mn^3$
- Multiplying a vector by an explicitly stored block $A_{ii}^{-1}A_{ii\pm1}$ takes mn^2 operations; doing this with an implicitly formed block takes $O(n)$ operations for the multiplication by $A_{ii\pm1}$, and $2mn^2$ operations for the solution with A_{ii}. This is on the same order as the explicit multiplication.
- We note that because of the sparsity pattern of A_{ii+1} only the backward sweep of A_{ii} is needed, halving the cost of applying the spike block $A_{ii}^{-1}A_{ii+1}$.

The preliminary conclusion of this analysis is that the flop count for factoring the subdomain interiors equals (up to lower order terms) that of factoring the matrix sequentially, and performing a system solve on the subdomains has the cost of solving a system sequentially. Clearly, dealing with the Spike matrix S is parallel overhead.

Next we factor the matrix S. This is no longer parallel over the subdomains, so we use two subdomains with separators to illustrate the inductive process.

$$S = \begin{array}{|c|c|c|c|c|} \hline S_{i-1i-1} & & S_{i-1i} & & \\ \hline S_{ii-1} & I & S_{ii+1} & & \\ \hline S_{i+1i-1} & & S_{i+1i+1} & & S_{i+1i+3} \\ \hline & & S_{i+2i+1} & I & S_{i+2i+3} \\ \hline & & S_{i+3i+1} & & S_{i+3i+3} \\ \hline \end{array}$$

We sweep the first column with a lower triangular matrix L_{i-1}:

$$\begin{array}{|c|c|c|c|c|}\hline I & & & & \\ \hline S_{ii-1}S_{i-1i-1}^{-1} & I & & & \\ \hline S_{i+1i-1}S_{i-1i-1}^{-1} & & I & & \\ \hline & & & I & \\ \hline & & & & I \\ \hline \end{array}^{-1} \cdot S = \begin{array}{|c|c|c|c|c|}\hline S_{i-1i-1} & & S_{i-1i} & & \\ \hline \emptyset & I & \tilde{S}_{ii+1} & & \\ \hline \emptyset & & \tilde{S}_{i+1i+1} & & S_{i+1i+3} \\ \hline & & S_{i+2i+1} & I & S_{i+2i+3} \\ \hline & & S_{i+3i+1} & & S_{i+3i+3} \\ \hline \end{array}$$

where we note that elements are updated:

$$\tilde{S}_{ii+1} = S_{ii+1} - S_{ii-1}S_{i-1i-1}^{-1}S_{i-1i+1}, \quad \tilde{S}_{i+1i+1} = S_{i+1i+1} - S_{i+1i-1}S_{i-1i-1}^{-1}S_{i-1i+1},$$

For this we need to factor the dense blocks S_{i-1i-1}, S_{i+1i+1}.

We are left with a matrix that is (block) upper triangular on the first subdomain, and has the same structure on the next subdomain as what we started out with. This is enough to continue the inductive process. In the end this leaves us with

$$S = \Pi_i L_i \Pi_i U_i.$$

The diagonal blocks $S_{i\pm1,i\pm1}$ are of size $n \times n$ and dense so to solve a system with them they have to be factored. The blocks $S_{i\pm1,i\mp1}$ are of the same size and dense.

Factorization cost analysis The operation count for the factorization is as follows.

- The separator block S_{i-1i-1} is dense of size n, so there is a cost of $1/3\,n^3$ muladds in factoring it.
- A further cubic cost of $2n^3$ comes from the update $\tilde{S}_{i+1i+1} = S_{i+1i+1} - S_{i+1i-1}S_{i-1i-1}^{-1}S_{i-1i+1}$.
- The update $\tilde{S}_{ii+1} = S_{ii+1} - S_{ii-1}S_{i-1i-1}^{-1}S_{i-1i+1}$ takes $2mn^3$ muladds.

Taking this together, the factorization of the Spike matrix takes the same $2Nmn^3$ operation count as factoring the interiors, which was the same as doing the sequential factorization, making a parallel factorization roughly twice as expensive as the sequential method.

In parallel, the dominant cost of forming $\tilde{S}_{ii+1}$ is not on the path of sequential dependencies, so it can be done in parallel, or by any inactive processors. The remaining cost is then forming the sequence of updated diagonal blocks $\tilde{S}_{i+1i+1}$, which adds a sequential time of $2Nn^3$, which is $O(m)$ lower than the cost of factoring the interiors.

Solution cost analysis The solution of a system $Ax = y$ involves the parallel subdomain solves with the A_{ii} blocks, and a sequential solve with the lower and upper factors of S. The important observation is that these carry sequential dependencies only between the separator blocks; the subdomain interiors depend on them, but carry no further dependencies. Hence, their solution can be happen after solving the separators system, or interweaved with it.

In terms of operations counts, the cost is dominated by subdomain solves that occur both in the block diagonal L and U factors, and in multiplying with the S_{ii-1} and S_{ii+1} spike blocks. Note that solving a spike system $Sx = y$ involves solving with the subdomain interiors twice, once in the forward and once in the backward solve, since $S_{ii-1} = -A_{ii}^{-1}A_{ii-1}$ and $S_{ii+1} = -A_{ii}^{-1}A_{ii+1}$. As argued above, the right spike only needs the backward solve.

Comparison to domain decomposition methods Traditionally, parallel factorizations are a variant of LU applied to a partitioned and permuted domain. This partitioning can be based on a multi-colouring (see [3, 4]), or on a division in subdomains and separators, see for instance [1, 2, 5]. This latter approach leads to a factorization *LSU* with a similar analysis as we just saw: a parallel solution of the subdomain interiors, and a sequential system that couples the separators. In the traditional case there is a solve on the interior during both the parallel forward and backward sweep, giving a total cost of $2 \cdot (|L| + |U|)$, where $|\cdot|$ indicates the cost of applying a matrix. In the Spike factorization there is a parallel interior solve plus an interior solve in multiplying by the spike matrices $A_{ii\pm1}$, where we note that A_{ii+1} only requires the backward solve. The Spike factorization carries a total cost of $3 \cdot |U| + 2 \cdot |L|$, making it more expensive by a factor of 5/4.

Yet another banded solver based on single separators can be found in [6].

3 Double separator case

One problem with domain decomposition methods using separators is the matter of distributing the separators. Since they are located between two subdomains, their processing does not trivially belong on either. We will now consider a method that allows for simpler work assignment, since it splits each separator in two; see figure 2.

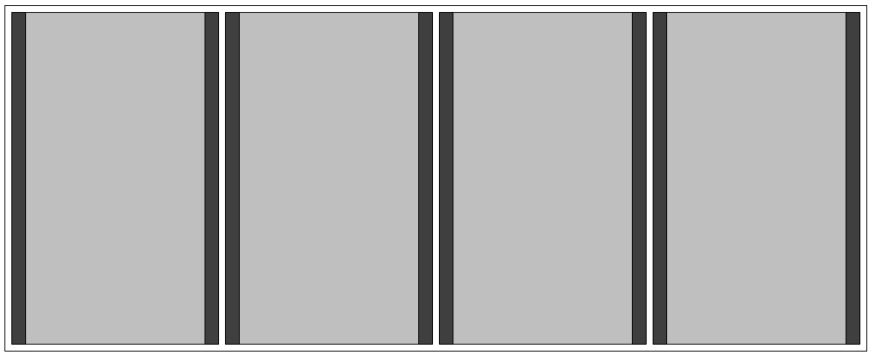

Figure 2: One dimensional partitioning of a domain, with separators divided over the processors

The resulting matrix has a structure

$$
\begin{array}{|c|c|c|c|c|c|c|}
\hline
* & * & & & & & \\
\hline
* & & & & & & \\
 & * & & & & & \\
 & & * & & & & \\
\hline
 & & * & * & & & \\
\hline
 & & * & * & * & & \\
\hline
 & & & * & & & \\
 & & & & * & & \\
 & & & & & * & \\
\hline
 & & & & & * & \ddots \\
\hline
 & & & & & \ddots & \ddots \\
\hline
\end{array}
\qquad (3)
$$

where the large blocks correspond to subdomain interiors and the small ones to separators, and the heavy lines indicate the boundary between processors.

To factor this we consider a subdomain with its separators and the connections to the previous and next subdomain

$$
A = \begin{array}{|c|c|c|c|c|}
\hline
A_{i-1i-2} & A_{i-1i-1} & A_{i-1i} & & \\
\hline
 & A_{ii-1} & A_{ii} & A_{ii+1} & \\
\hline
 & & A_{i+1i} & A_{i+1i+1} & A_{i+1i+2} \\
\hline
\end{array}
$$

First we eliminate the subdomain interior. Applying the forward sweep gives:

$$L^{-1}A = \begin{array}{|c|c|c|}\hline 1 & & \\ \hline & L_{ii} & \\ \hline & & 1 \\ \hline \end{array}^{-1} \quad A = \begin{array}{|c|c|c|c|c|}\hline A_{i-1i-2} & A_{i-1i-1} & A_{i-1i} & & \\ \hline & L_{ii}^{-1}A_{ii-1} & U_{ii} & L_{ii}^{-1}A_{ii+1} & \\ \hline & & A_{i+1i} & A_{i+1i+1} & A_{i+1i+2} \\ \hline \end{array}$$

and after the backward sweep:

$$U^{-1}(L^{-1}A) = \begin{array}{|c|c|c|}\hline 1 & & \\ \hline & U_{ii} & \\ \hline & & 1 \\ \hline \end{array}^{-1} \quad (L^{-1}A) = \begin{array}{|c|c|c|c|c|}\hline A_{i-1i-2} & A_{i-1i-1} & A_{i-1i} & & \\ \hline & A_{ii}^{-1}A_{ii-1} & I & A_{ii}^{-1}A_{ii+1} & \\ \hline & & A_{i+1i} & A_{i+1i+1} & A_{i+1i+2} \\ \hline \end{array}$$

Next we left-multiply by a matrix T to eliminate the connection between the subdomain interior and the separators:

$$S = T^{-1}U^{-1}L^{-1}A = \begin{array}{|c|c|c|}\hline I & A_{i-1i} & \\ \hline & I & \\ \hline & A_{i+1i} & I \\ \hline \end{array}^{-1} \quad U^{-1}(L^{-1}A) = \begin{array}{|c|c|c|c|c|}\hline S_{i-1i-2} & S_{i-1i-1} & \emptyset & S_{i-1i+1} & \\ \hline & S_{ii-1} & I & S_{ii+1} & \\ \hline & S_{i+1i-1} & \emptyset & S_{i+1i+1} & S_{i+1i+2} \\ \hline \end{array} \tag{4}$$

where $S_{i-1i-2} = A_{i-1i-2}$, $S_{i+1i+2} = A_{i+1i+2}$ and the following matrices will be explicitly formed:

$$\begin{array}{l} S_{i-1i-1} = A_{i-1i-1} - A_{i-1i}A_{ii}^{-1}A_{ii-1}, \quad S_{i+1i+1} = A_{i+1i+1} - A_{i+1i}A_{ii}^{-1}A_{ii+1} \\ S_{i+1i-1} = -A_{i+1i}A_{ii}^{-1}A_{i-1}, \quad S_{i-1i+1} = -A_{i-1i}A_{ii}^{-1}A_{ii+1} \end{array};$$

the blocks

$$S_{ii-1} = A_{ii}^{-1}A_{ii-1}, \quad A_{ii}^{-1}A_{ii+1}$$

are not explicitly formed, and their application involves a subdomain solve with A_{ii}.

Together we now have a factorization

$$A = LUTS$$

where the first three factors can be processed in parallel, both during the factorization and the system solution. It remains to analyze S. We see that the interior of the subdomain depends on the two separators, but not the other way around: in effect we now have a linear system where only the separators, two per subdomain, are mutually coupled. We can now proceed in two different ways.

3.1 Direct factorization

To make an inductive analysis of the factorization of S we consider, for purposes of illustration, two full subdomains with separators.

S_{i-1i-1}		S_{i-1i}			
S_{ii-1}	I	S_{ii+1}			
S_{i+1i-1}		S_{i+1i+1}	S_{i+1i+2}		
		S_{i+2i+1}	S_{i+2i+2}		S_{i+2i+4}
			S_{i+3i+2}	I	S_{i+3i+4}
			S_{i+4i+3}		S_{i+4i+4}

We make an LU factorization of this matrix. The first step of forward sweep is applying a matrix $L_i^{(1)}$ to sweep the first column:

$$L_i^{(1)^{-1}} S = \begin{array}{|c|c|c|} \hline I & & \\ \hline S_{ii-1}S_{i-1i-1}^{-1} & I & \\ \hline S_{i+1i-1}S_{i-1i-1}^{-1} & & I \\ \hline \end{array} \qquad \begin{array}{|c|} \hline \\ \\ \\ \hline I \\ \hline \end{array}$$

$$S = \begin{array}{|c|c|c|c|c|c|} \hline S_{i-1i-1} & & S_{i-1i} & & & \\ \hline \emptyset & I & \tilde{S}_{ii+1} & & & \\ \hline \emptyset & & \tilde{S}_{i+1i+1} & S_{i+1i+2} & & \\ \hline & & S_{i+2i+1} & S_{i+2i+2} & & S_{i+2i+4} \\ \hline & & & S_{i+3i+2} & I & S_{i+3i+4} \\ \hline & & & S_{i+4i+3} & & S_{i+4i+4} \\ \hline \end{array}$$

where

$$\tilde{S}_{ii+1} = S_{ii+1} - S_{ii-1}S_{i-1i-1}^{-1}S_{i-1i+1}, \quad \tilde{S}_{i+1i+1} = S_{i+1i+1} - S_{i+1i-1}S_{i-1i-1}^{-1}S_{i-1i+1}$$

Secondly, we apply a matrix $L_i^{(2)}$ to communicate with the left separator of the next subdomain:

$$
\begin{array}{|c|c|c|c|c|c|}
\hline
I & & & & & \\ \hline
 & I & & & & \\ \hline
 & & I & & & \\ \hline
 & & \tilde{S}_{i+2i+1}\tilde{S}^{-1}_{i+1i+1} & I & & \\ \hline
 & & & & I & \\ \hline
 & & & & & I \\ \hline
\end{array}^{-1}
L^{(1)^{-1}}S=
\begin{array}{|c|c|c|c|c|c|}
\hline
S_{i-1i-1} & & S_{i-1i} & & & \\ \hline
\emptyset & I & \tilde{S}_{ii+1} & & & \\ \hline
\emptyset & & \tilde{S}_{i+1i+1} & S_{i+1i+2} & & \\ \hline
 & & \emptyset & \tilde{S}_{i+2i+2} & & S_{i+2i+4} \\ \hline
 & & & S_{i+3i+2} & I & S_{i+3i+4} \\ \hline
 & & & S_{i+4i+3} & & S_{i+4i+4} \\ \hline
\end{array}
$$

where

$$\tilde{S}_{i+2i+2} = S_{i+2i+2} - \tilde{S}_{i+2i+1}\tilde{S}^{-1}_{i+1i+1}S_{i+1i+2}.$$

The second subdomain now has the same connections as the first had, so we can continue inductively.

Parallel solve time We see that the spike matrix can be factored as a product of $L^{(1)}, L^{(2)}, U$ matrices per subdomain. Solving a system with any of these takes solving a system on a separator. The interiors are not on the critical path, so the parallel solve consists of solving the separator system sequentially, and then the interiors in parallel.

Comparison to single separators This factorization behaves much like the single-separator case in section 2. The only difference is the introduction of the $L^{(2)}$ matrix connecting the right separator of one domain and the left separator of the next. This mostly adds one solution with an $n \times n$ dense matrix per subdomain to the sequential time.

3.2 Full elimination of the subdomain

There is a second way of dealing with the spike matrix, which we show using only a single subdomain:

$$S = \begin{array}{|c|c|c|c|c|} \hline S_{i-1i-2} & S_{i-1i-1} & & S_{i-1i+1} & \\ \hline & S_{ii-1} & I & S_{ii+1} & \\ \hline & S_{i+1i-1} & & S_{i+1i+1} & S_{i+1i+2} \\ \hline \end{array}$$

We first sweep the left separator by applying a lower triangular matrix $L^{(s)}$:

$$L^{(s)^{-1}}S = \begin{array}{|c|c|c|} \hline S_{i-1i-1} & \emptyset & \\ \hline S_{ii-1} & I & \\ \hline S_{i+1i-1} & \emptyset & I \\ \hline \end{array}^{-1} \quad S = \begin{array}{|c|c|c|c|c|} \hline S_{i-1i-2} & I & & S_{i-1i+1} & \\ \hline \tilde{S}_{ii-2} & \emptyset & I & S_{ii+1} & \\ \hline \tilde{S}_{i+1i-2} & \emptyset & & S_{i+1i+1} & S_{i+1i+2} \\ \hline \end{array}$$

Next we sweep the second separator with a matrix $U^{(s)}$:

$$U^{(s)^{-1}}L^{(s)^{-1}}S = \begin{array}{|c|c|c|} \hline I & & S_{i-1i+1} \\ \hline & I & S_{ii+1} \\ \hline & & S_{i+1i+1} \\ \hline \end{array}^{-1} \quad L^{(s)^{-1}}S = \begin{array}{|c|c|c|c|c|} \hline S_{i-1i-2} & I & & \emptyset & \tilde{S}_{i-1i+2} \\ \hline \tilde{S}_{ii-2} & \emptyset & I & \emptyset & \tilde{S}_{ii+1} \\ \hline \tilde{S}_{i+1i-2} & \emptyset & & I & S_{i+1i+2} \\ \hline \end{array}$$

The resulting factorization

$$A = LUL^{(s)}U^{(s)}Z$$

gives the matrix Z appearing in the original Spike algorithm, and is fully parallel in the parts just considered: any sequential component is entirely in the Z matrix.

We show the factorization of the Z matrix by considering two subsequent subdomains.

The factorization is then

$$
\begin{array}{|c|c|c|c|c|c|}
\hline
I & & & S_{i-1i+2} & & \\ \hline
 & I & & S_{ii+2} & & \\ \hline
 & & I & S_{i+1i+2} & & \\ \hline
 & & S_{i+2i+1} & I & & \\ \hline
 & & S_{i+3i+1} & & I & \\ \hline
 & & S_{i+4i+1} & & & I \\ \hline
\end{array}
=
$$

$$
\begin{array}{|c|c|c|c|c|c|}
\hline
I & & & & & \\ \hline
 & I & & & & \\ \hline
 & & I & & & \\ \hline
 & & S_{i+2i+1} & I & & \\ \hline
 & & S_{i+3i+1} & & I & \\ \hline
 & & S_{i+4i+1} & & & I \\ \hline
\end{array}
\begin{array}{|c|c|c|c|c|c|}
\hline
I & & & & & \\ \hline
 & I & & & & \\ \hline
 & & I & & & \\ \hline
 & & & I - S_{i+2i+1}S_{i+1i+2} & & \\ \hline
 & & & -S_{i+3i+1}S_{i+1i+2} & I & \\ \hline
 & & & -S_{i+4i+1}S_{i+1i+2} & & I \\ \hline
\end{array}
\cdot
$$

$$
\begin{array}{|c|c|c|c|c|c|}
\hline
I & & & S_{i-1i+2} & & \\ \hline
 & I & & S_{ii+2} & & \\ \hline
 & & I & S_{i+1i+2} & & \\ \hline
 & & & I & & \\ \hline
 & & & & I & \\ \hline
 & & & & & I \\ \hline
\end{array}
$$

As before, we observe that only the separators are mutually dependent; the interiors are dependent on the separators but not the other way around.

3.3 Original derivation of the Spike algorithm

Instead of going through the S matrix, we can also derive Z directly. As before, we eliminate the subdomain interiors, and their connections with the separators; see the derivation of equation (2). Thus we start with

A_{i-1i-2}	A_{i-1i-1}		A_{i-1i+1}	
	A_{ii-1}	I	A_{ii+1}	
	A_{i+1i-1}		A_{i+1i+1}	A_{i+1i+2}

Note that these are no longer the original matrix blocks. We continue factoring by sweeping the column of the first separator: $A^{(1)} = LA^{(2)}$

I		
$A_{ii-1}A_{i-1i-1}^{-1}$	I	
$A_{i+1i-1}A_{i-1i-1}^{-1}$		I

A_{i-1i-2}	A_{i-1i-1}		A_{i-1i+1}	
A_{ii-2}	$\emptyset$	I	A_{ii+1}	
A_{i+1i-2}	$\emptyset$		A_{i+1i+1}	A_{i+1i+2}

where

$$\begin{cases} A_{ii-2} = -A_{ii-1}A_{i-1i-1}^{-1}A_{i-1i-2} \\ A_{i+1i-2} = -A_{i+1i-1}A_{i-1i-1}^{-1}A_{i-1i-2} \\ A_{ii+1} \leftarrow A_{ii+1} - A_{ii-1}A_{i-1i-1}^{-1}A_{i-1i+1} \\ A_{ii+1} \leftarrow A_{ii+1} - A_{ii-1}A_{i-1i-1}^{-1}A_{i-1i+1} \end{cases}$$

Now we sweep the upper part of the column of the second separator: $A^{(2)} = UA^{(3)}$

I		$A_{i-1i+1}A_{i+1i+1}^{-1}$
	I	$A_{ii+1}A_{i+1i+1}^{-1}$
		I

A_{i-1i-2}	A_{i-1i-1}			A_{i-1i+2}
A_{ii-2}		I		A_{ii+2}
A_{i+1i-2}			A_{i+1i+1}	A_{i+1i+2}

where now blocks in the right spike are newly formed, and ones in the left spike get updated. After normalizing the diagonal blocks on the separators, we now have an identity block for the whole subdomain, and the traditional spikes flanking it, which is the Z matrix of the previous subsection.

4 Discussion

We have given three one-sided factorization algorithms, of which one is a computational variant of the Spike algorithm. The factorizations are presented using the interior/separator division of the subdomains that is commonly associated with domain decomposition methods. These methods, applied to sparse finite element type matrices, are seen to be slightly more expensive than traditional domain decomposition methods. It should be noted, however, that the actual performance of methods depends on memory access patterns and other matters not considered in this paper.

Acknowledgments

This work was sponsored by NSF through awards CCF 0917096 and OCI-0850750. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation (NSF).

References

[1] P. Bjørstad and O. Widlund. Iterative methods for the solution of elliptic problems on regions partitioned in to substructures. *SIAM J. Numer. Anal.*, 23:1097–1120, 1986.

[2] M. Dryja. A capacitance method for elliptic problems on regions partitioned into substructures. *Numer. Math.*, 39:51–64, 1982.

[3] M.T. Jones and P.E. Plassmann. A parallel graph coloring heuristic. *SIAM J. Sci. Stat. Comput.*, 14, 1993.

[4] M. Luby. A simple parallel algorithm for the maximal independent set problem. *SIAM Journal on Computing*, 4, 1986.

[5] Gérard Meurant. Domain decomposition methods for partial differential equations on parallel computers. *Int. J. Supercomputing Appls.*, 2:5–12, 1988.

[6] Maxim Naumov and Ahmed H. Sameh. A tearing-based hybrid parallel banded linear system solver. *Journal of Computational and Applied Mathematics*, 226(2):306–318, 2009. Special Issue: Large scale scientific computations.

[7] E. Polizzi and A.H. Sameh. A parallel hybrid banded system solver: the SPIKE algorithm. *Parallel Computing*, 32:177–194, 2006.

[8] A. H. Sameh and D. J. Kuck. On stable parallel linear system solvers. *J. Assoc. Comput. Mach.*, 25(1):81–91, Jan. 1978.