Performace Prediction Model for Spike Algorithm on Large Number of Processors

Murat Manguoglu *
mmanguog@cs.purdue.edu

Ananth Grama*
ayg@cs.purdue.edu

April 29, 2008

*Department of Computer Science, Purdue University

1 Introduction

2 Algorithm Description and Cost Analysis

2.1 Algorithm Description

Given a diagonally dominant linear system

$$Ax = f \tag{1}$$

We describe a version of truncated spike algorithm. To illustrate the algorithm we assume 4 partitions as follows

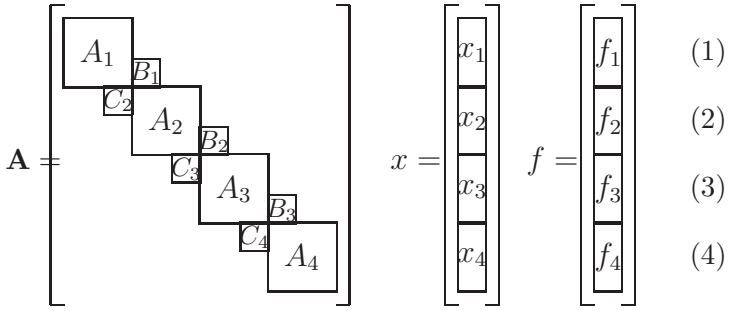

Figure 1: Illustration of the partitioning of the linear system

1. Compute the LU (via DDBTRF)
   - $L_j U_j \leftarrow A_j$ for $j = 1, 2, 3, 4$
2. Compute spikes (via DTBTRS)
   - Solve for V_j: $L_j U_j V_j = \begin{bmatrix} 0 & \cdots & 0 & B_j^T \end{bmatrix}^T$ for $j = 1, 2, 3$
   - Solve for W_j: $L_j U_j W_j = \begin{bmatrix} C_j^T & 0 & \cdots & 0 \end{bmatrix}^T$ for $j = 2, 3, 4$.
3. Communicate spike tips $W_j^{(t)}$ (processor j sends to processor j-1 , for $j = 2, 3, 4$). $W_j^{(t)}$ is the top $k \times k$ part of W_j.
4. Factorize the reduced system (via DGETRF)

$$\begin{pmatrix} I & V_j^{(b)} \\ W_j^{(t)} & I \end{pmatrix} \tag{2}$$

   for $j = 2, 3, 4$.

5. Modify the right hand side by solving (via DTBTRS) : $L_j U_j g_j = f_j$ $(j = 1, 2, 3, 4)$

6. Communicate the modified righ hand side tips $g_j^{(t)}$ (processor j sends to processor j-1 for $j = 2, 3, 4$).

7. Solve the reduced system

$$\begin{pmatrix} I & V_j^{(b)} \\ W_{j+1}^{(t)} & I \end{pmatrix} \begin{pmatrix} x_j^{(b)} \\ x_{j+1}^{(t)} \end{pmatrix} = \begin{pmatrix} g_j^{(b)} \\ g_{j+1}^{(t)} \end{pmatrix} \tag{3}$$

   for $j = 2, 3, 4$.

8. Communicate the reduced system solution $g_{j+1}^{(t)}$ (processor j sends to processor j+1 for $j = 1, 2, 3$) .

9. Retrieve x_j $(j = 1, 2, 3, 4)$ (via DGEMV) $x_j = f_j - V_j x_{j+1}^{(t)} - W_j x_{j-1}^{(b)}$ $(V_4 = 0$ and $W_1 = 0)$

2.2 Cost of the algorithm

Here we assume, $k = kl = ku$, i.e. upper and lover bandwidths of the matrix are equal to k. The dimension of the matrix is N. The dimension of the partitioned blocks are $n = N/p$.

We model the each stage of the algorithm described in the previous section. Stages $1, 2, 4, 5, 7, 9$ are computations and memory references, while $3, 6, 8$ are communications and memory references. In Table 1, we depict a cost model for each of these stages.

Table 1: Truncated Spike Algorithm Cost Model

Stage	Description	Cost
1	Factorize the Diagonal Blocks	$\alpha_1 nk^2 + \beta_1 nk$
2	Compute Spikes	$\alpha_2 nk^2 + \beta_2 nk$
3	Communicate Tips of Spikes	$\alpha_3 k^2(p-1) + \beta_3 k^2 + \gamma_3$
4	Factorize The Reduced System	$\alpha_4 k^3 + \beta_4 k^2$
5	Modify the Right Hand Side	$\alpha_5 nk$
6	Communicate Tips of MRHS	$\alpha_6 k(p-1) + \beta_6 k + \gamma_6$
7	Solve the Reduced System	$\alpha_7 k^2$
8	Communicate Solution of the Reduced System	$\alpha_8 k(p-1) + \beta_8 k + \gamma_8$
9	Retrieve the Solution	$\alpha_9 nk + \beta_9 n$

Both Stage 1 and Stage 2 cost $O(nk^2)$ computation and $O(nk)$ memory references. In Stage 4, the cost is $O(k^3)$ computation an $O(k^2)$ memory references. In Stage 5, we have $O(nk)$ computation and $O(nk)$ memory references. Similary in Stage 7 we have $O(k^2)$ computation and $O(k^2)$ memory references. Finally, in Stage 9 there are $O(nk)$ computation for matrix vector product and $O(n)$ computation for vector addition. The memory references for the last stage are $O(nk)$ and $O(n)$.

In Stages $3, 6, 8$ we model the cost by $O(data)$ for the communication of data, $O(data \times (p-1))$ for the saturation of the network, and a constant for MPI communication the overhead. Where *data* is the amount of data to be communicated between neighboring processors.

3 Training

For training we used a digonally dominant toeplitz system of dimension $5,000,000$ with 4.0 on the main diagonal -1.0 on the first off diagonals and -0.01 on all other off diagonals with a right hand side vector of all ones.

First platform we trained the model is Ranger Sun Constellation Linux Cluster at TACC. Ranger has $3,936$ nodes and each node has 16 cores (of AMD Barcelona Processor) . Interconnection network between nodes is Infiniband.

Training is done using $16, 32, 64$ processors ($1, 2, 4$ nodes respectively) for bandwidths $k = 15, 25, 35$. We timed the individual stages of the algorithm via wall clock time on each processor independently and use the maximum among each MPI process as the time consumed in that stage.

Using a least squares approximation we find the following parameters $\alpha_i, \beta_i, \gamma_i$ for $i = 1, .., 9$.

Table 2: Cost Model Parameters

i	α_i	β_i	γ_i
1	8.62×10^{-10}	2.61×10^{-8}	-
2	3.96×10^{-8}	3.66×10^{-7}	-
3	9.07×10^{-14}	1.21×10^{-6}	9.48×10^{-4}
4	1.65×10^{-29}	7.43×10^{-6}	-
5	2.85×10^{-8}	-	-
6	4.31×10^{-9}	4.10×10^{-7}	3.59×10^{-5}
7	1.19×10^{-7}	-	-
8	2.57×10^{-7}	3.39×10^{-31}	6.42×10^{-4}
9	8.86×10^{-8}	9.51×10^{-9}	-

4 Verification

In order to verify the model we used two different systems one with the same toeplitz system used in the training and the other one is another toeplitz system with dimension $10,000,000$. We verify the model for 128, 246, 512, and $1,024$ processors for these two systems.

Table 3: Model Verification

N	k	p	Observed	Model	Error
$5,000,000$	35	128	2.78	2.64	0.13
$5,000,000$	25	128	1.55	1.49	0.06
$5,000,000$	15	128	0.70	0.66	0.04
$5,000,000$	35	256	1.49	1.33	0.16
$5,000,000$	25	256	0.79	0.75	0.04
$5,000,000$	15	256	0.35	0.33	0.02
$5,000,000$	35	512	0.67	0.67	0.00
$5,000,000$	25	512	0.38	0.38	0.00
$5,000,000$	15	512	0.20	0.17	0.03
$5,000,000$	35	$1,024$	0.37	0.35	0.02
$5,000,000$	25	$1,024$	0.21	0.20	0.01
$5,000,000$	15	$1,024$	0.10	0.09	0.01
$10,000,000$	35	128	5.36	5.27	0.09
$10,000,000$	25	128	3.03	2.98	0.06
$10,000,000$	15	128	1.36	1.31	0.05
$10,000,000$	35	256	2.78	2.64	0.13
$10,000,000$	25	256	1.55	1.49	0.06
$10,000,000$	15	256	0.68	0.66	0.02
$10,000,000$	35	512	1.51	1.33	0.18
$10,000,000$	25	512	0.79	0.75	0.04
$10,000,000$	15	512	0.35	0.33	0.02
$10,000,000$	35	$1,024$	0.69	0.68	0.01
$10,000,000$	25	$1,024$	0.45	0.38	0.07
$10,000,000$	15	$1,024$	0.19	0.17	0.02

5 Conclusions

6 Acknowledgements

We would like to thank you Professor Zhiyuan Li for running tests on TACC Ranger platform.