OPTIMAL LOAD BALANCING TECHNIQUES FOR BLOCK-CYCLIC DECOMPOSITIONS FOR MATRIX FACTORIZATION

Peter Strazdins

Australian National University

e-mail: peter@cs.anu.edu.au

December 14, 1998

1 Overview

- what is the best advanced technique for parallel dense linear algebra libraries?
  - 1990–6: storage blocking developed as main technique; used in ScaLAPACK
  - 1994–8: algorithmic blocking developed as an enhancement
  - 1997–8: conception of new load balancing methods (PLAPACK, ANU)
- overview of the Fujitsu AP+
- background concepts: the block-cyclic matrix distribution; block-partitioned LU decomposition & its ||ization (via storage blocking methods)
- panel scattering: overall idea & important properties
  - implementation issues for block-cyclic matrices
  - performance in LU, LLT and QR factorizations
    - comparisons with storage and algorithmic blocking on the Fujitsu AP+
    - effect of block size and performance analysis of the three methods
- conclusions

2 Overview of the Fujitsu AP+

- the AP+ is a SuperSPARC-based distributed memory multiprocessor
  - cells have a single 16 KB write-through data cache
    - $\Rightarrow$ a blocking factor of $\omega \approx 32$ optimal

    and a 64-entry TLB
  - local matrix multiply speed is $\gamma_3^{-1} = 33$ MFLOPS
  - communication startup cost is $\alpha = 12\mu s$, bandwidth is $\beta^{-1} = 13$ MBs
- cells are connected in a torus network using wormhole routing
  - innovation: row/column broadcasts have the cost of a normal message
    - in a row b/c, the router in each intermediate cell both forwards and collects the message
    - very useful for linear algebra applications!
  - global broadcasts have 2x the latency (requires a synchronization for their safe operation)

3 Background: the Block-cyclic Matrix Distribution

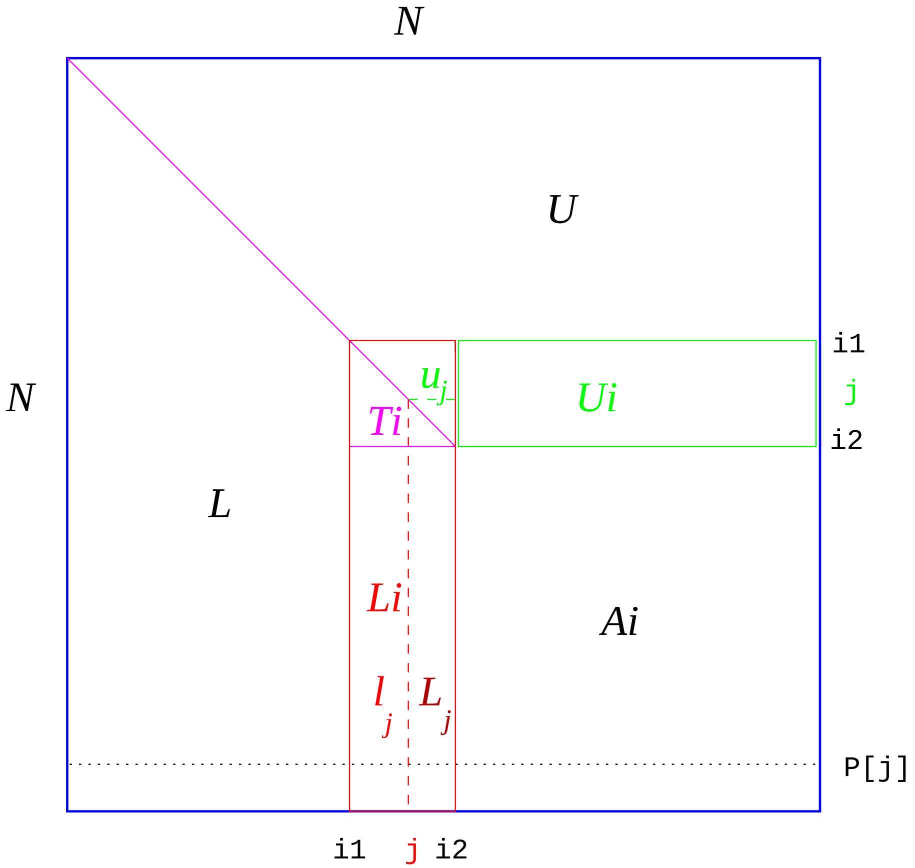
- ‘standard’ for most parallel dense linear algebra applications
- ie. divide the global matrix A into $r \times s$ blocks on a $P \times Q$ grid;
  if 0th block is on cell (p_0, q_0), block (i, j) of A is on cell $((i+p_0)\%P, (j+q_0)\%Q)$
- eg. if $p_0 = q_0 = 0, r = 3, s = 2$ on a 2×3 grid, a 10×10 matrix A is distributed:

a_{00}	a_{01}	a_{06}	a_{07}	a_{02}	a_{03}	a_{08}	a_{09}	a_{04}	a_{05}
a_{10}	a_{11}	a_{16}	a_{17}	a_{12}	a_{13}	a_{18}	a_{19}	a_{14}	a_{15}
a_{20}	a_{21}	a_{26}	a_{27}	a_{22}	a_{23}	a_{28}	a_{29}	a_{24}	a_{25}
a_{60}	a_{61}	a_{66}	a_{67}	a_{62}	a_{63}	a_{68}	a_{69}	a_{64}	a_{65}
a_{70}	a_{71}	a_{76}	a_{77}	a_{72}	a_{73}	a_{78}	a_{79}	a_{74}	a_{75}
a_{80}	a_{81}	a_{86}	a_{87}	a_{82}	a_{83}	a_{88}	a_{89}	a_{84}	a_{85}
a_{30}	a_{31}	a_{36}	a_{37}	a_{32}	a_{33}	a_{38}	a_{39}	a_{34}	a_{35}
a_{40}	a_{41}	a_{46}	a_{47}	a_{42}	a_{43}	a_{48}	a_{49}	a_{44}	a_{45}
a_{50}	a_{51}	a_{56}	a_{57}	a_{52}	a_{53}	a_{58}	a_{59}	a_{54}	a_{55}
a_{90}	a_{91}	a_{96}	a_{97}	a_{92}	a_{93}	a_{98}	a_{99}	a_{94}	a_{95}

- PBMD (Physically Based Matrix Distribution) is a special case, with $s = Pr$ (PLAPACK)
  - $\surd$ enables special algorithms
  - $\times$ s is large

4 Background: Blocked LU Factorization

- uses partial pivoting on an $N \times N$ matrix A (on step i, $i_1 = i\omega, i_2 = i_1 + \omega$)



1. factor L^i:
   `for` j $= i_1..i_2$
   find $P[j]$ s.t. $|L^i{}_{P[j],j}| \geq |L^i{}_{j:M,j}|$
   $L^i{}_{j,:} \leftrightarrow L^i{}_{P[j],:}$
   $l_j \leftarrow l_j / L^i_{j,j}$
   $L_j \leftarrow L_j - l_j u_j$
2. do swaps for $P[i_1..i_2]$ on A (outside L^i)
3. row broadcast of T^i, L^i
4. $U^i \leftarrow (T^i)^{-1} U^i$
5. column broadcast of U^i
6. $A^i \leftarrow A^i - L^i U^i$

- LLT (Cholesky) factorization similar except no U^i panel and A^i is symmetric (lower triangular form)

5 Background: Parallel Factorization Methods

- LU has $(3\lg_2 P + 2)N\alpha$ commun. startup cost in forming L^i; (also $4N$ others)
  - cf. $2/3N^3\gamma_3$ FLOP cost, $(2/Q + \frac{\lg_2 P}{2}/Q + 1/P)N^2\beta$ commun. volume cost
- storage blocking: $\omega = r = s$:
  - ie. one processor column (row) holds L^i (U^i)
  - × suffers from load imbalance: (nb. typically $1.5 \leq \gamma_{2.5}/\gamma_3, \gamma_2/\gamma_{2.5} \leq 2$)
    - formation of L^i (largest cost): $N^2 s/P\gamma_2$; also of U^i: $N^2 r/Q\gamma_{2.5}$
    - also due to block size in $A^i \leftarrow A^i - L^i U^i$: $N^2(r/Q + s/P)\gamma_3$
- algorithmic blocking: uses optimal ω, $r = s \approx 1$
  - $\surd$ greatly reduces these imbalances
  - × introduces $4N$ extra commun. startups; local panel width is small ($\approx \omega/Q$)
- LLT has lower startup costs $(4N/r\alpha)$; LLT & QR have higher block size load imbalances
- but can we get perfect load balance, keeping the local panel width at ω?

6 Panel Scattering and its Implementation

- idea: when a subset of cells forms data which must then be broadcast to all others, re-distribute the data first (enables a faster broadcast!)
  - for LU etc., re-distribute the $M \times \omega$ panel L^i onto a logical $PQ \times 1$ grid
  - eg. $M = 25r, \omega = 2s$ on a 3×4 grid; leading block at cell $(0, 0)$

$l_{0,0}$ $l_{3,0}$ $\ldots$ $l_{24,0}$	$l_{0,1}$ $l_{3,1}$ $\ldots$ $l_{24,1}$		
$l_{1,0}$ $l_{4,0}$ $\ldots$ $l_{22,0}$	$l_{1,1}$ $l_{4,1}$ $\ldots$ $l_{22,1}$		
$l_{2,0}$ $l_{5,0}$ $\ldots$ $l_{23,0}$	$l_{2,1}$ $l_{5,1}$ $\ldots$ $l_{23,1}$		

$\Rightarrow$

$l_{0,0}$ $l_{0,1}$ $l_{12,0}$ $l_{12,1}$ $l_{24,0}$ $l_{24,1}$	$l_{3,0}$ $l_{3,1}$ $l_{15,0}$ $l_{15,1}$	$l_{6,0}$ $l_{6,1}$ $l_{18,0}$ $l_{18,1}$	$l_{9,0}$ $l_{9,1}$ $l_{21,0}$ $l_{21,1}$
$l_{1,0}$ $l_{1,1}$ $l_{13,0}$ $l_{13,1}$	$l_{4,0}$ $l_{4,1}$ $l_{16,0}$ $l_{16,1}$	$l_{7,0}$ $l_{7,1}$ $l_{19,0}$ $l_{19,1}$	$l_{10,0}$ $l_{10,1}$ $l_{22,0}$ $l_{22,1}$
$l_{2,0}$ $l_{2,1}$ $l_{14,0}$ $l_{14,1}$	$l_{5,0}$ $l_{5,1}$ $l_{17,0}$ $l_{17,1}$	$l_{8,0}$ $l_{8,1}$ $l_{20,0}$ $l_{20,1}$	$l_{11,0}$ $l_{11,1}$ $l_{23,0}$ $l_{23,1}$

- Properties:

1. the scatter can be performed by P independent row-wise (block-cyclic) matrix transpositions over each $1 \times Q$ sub-grid
2. (after formation,) the scattered panel can be replicated across the rows of the $P \times Q$ grid by P independent (block-cyclic) transposed-spread operations over each $1 \times Q$ sub-grid (same cost as an ordinary row 'spread')

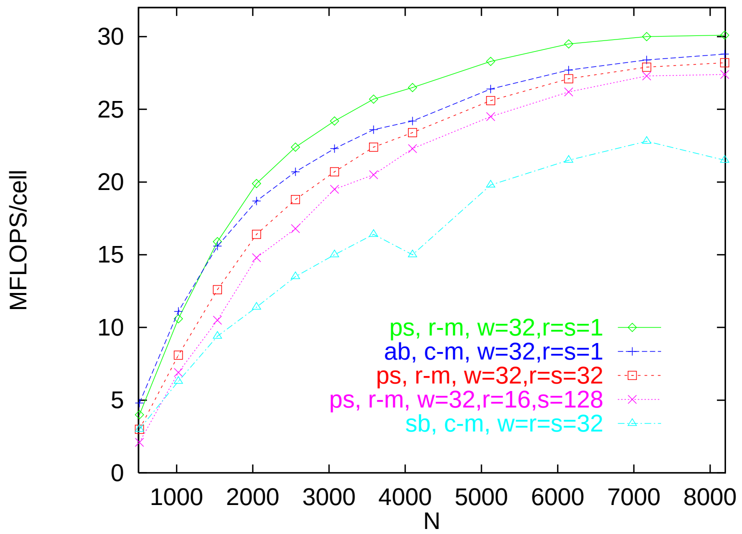
$\Rightarrow$ easy & elegant implementation using an existing block-cyclic || BLAS library

- To maintain these Properties, the layout of the $PQ \times 1$ logical grid is non-obvious if the leading block is not at cell $(0, 0)$.
eg. if at cell $(2, 1)$:

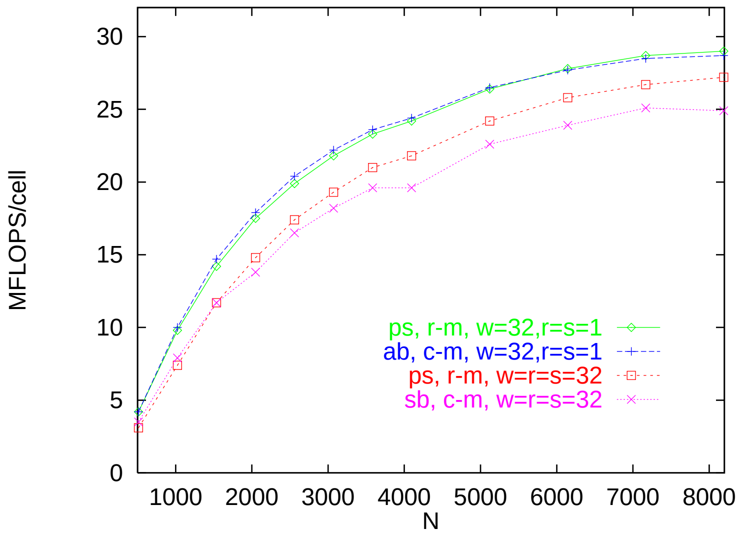
14	2	6	10
15	3	7	11
12	0	4	8
13	1	5	9

- panel scattering requires the addition of 4–6 extra procedure calls per panel:
  - must copy the (row-replicated) formed panel back into the original matrix
  - triangular factors (eg. T^i) must be fully-replicated
    - no extra commun. startups for $r = s = 1$ if do this *before* forming T^i!
- the only significant difference for PBMD is that column-replicated $(L^i)^T$ can be obtained directly from the row-scattered L^i

7 Performance Comparisons on an 8×8 AP+



(a) LU



(b) LLT

- optimal ω chosen for each case; QR faster but trends similar to LU
- for LU, panel scattering (ps) able to use row-major (r-m) matrices:
  - $\Rightarrow$ faster row swaps, but no (cache / TLB) penalty as scattered L^i is packed
- low r, s fastest for algorithmic blocking & panel scattering
  - $r = 16, s = 128$ corresponds to PBMD (still has significant load imbalance)

8 Performance Analysis

- algorithmic blocking has slower panel formation: only $\approx 80\%$ load balance, $\approx 40\%$ loss in cell computational speed ($\omega = 32$ on 8×8 AP+)
- for $r = 1$, LU has limited $||$ization factor of ≈ 2.3 in the row swaps
- the scattering of panels has very small cost at $r = 1$ (3x faster than $r = 32$ on 8×8 AP+)
- comparison of communication volume costs $(\times N^2/2\beta)$ in panel replication:

	L^i		U^i	
	AP+	*	AP+	*
storage blocking, $\omega = r = s$	1	2	1	$\lg_2 P$
algorithmic blocking, $r = s = 1$	1	2	1	κ_r
panel scattering	1	κ_r	1	κ_r

where $1 \leq \kappa_r \leq 2$ is relative cost of replication via 'ring-shift' methods

- comparison of number of commun. startups (assume $\omega > P, Q$; main $O(N)$ terms only):

	LU		LLT		QR	
	AP+	*	AP+	*	AP+	*
storage blocking, $\omega = r = s$	$11N$	$15N$	0	0	$8N$	$12N$
algorithmic blocking, $r = s = 1$	$12N$	$18N$	N	$2N$	$10N$	$15N$
panel scattering	$13N$	$22N$	0	0	$16N$	$24N$

- all these factors in a detailed performance model, highly accurate on the AP+

9 Conclusions

- panel scattering is a general technique for block-partitioned algorithms
- with suitable algorithmic and data abstractions in a || BLAS library, only a little more complex to implement than algorithmic blocking
- also requires small block sizes for optimal load balance in communication as well as computation (surprisingly, this need introduce no extra communication startups!)
- for small N, algorithmic blocking is faster due to reduced communication startup, memory copy and software overheads
- for moderate–large N, panel scattering has slight load balance and computation speed advantages:
  - more significant is the natural optimization of memory access patterns, due to the packing of the panels
- both are sufficiently superior to storage blocking to justify extra impl. effort
- moral: data-redistribution can sometimes reduce overall communication volume costs, as well as improved load balance!