

SPARSITY: Optimization Framework For Sparse Matrix Kernels

Eun-Jin Im, Katherine Yelick, Richard Vuduc

International Journal of High Performance Computing Applications 2004 18: 135

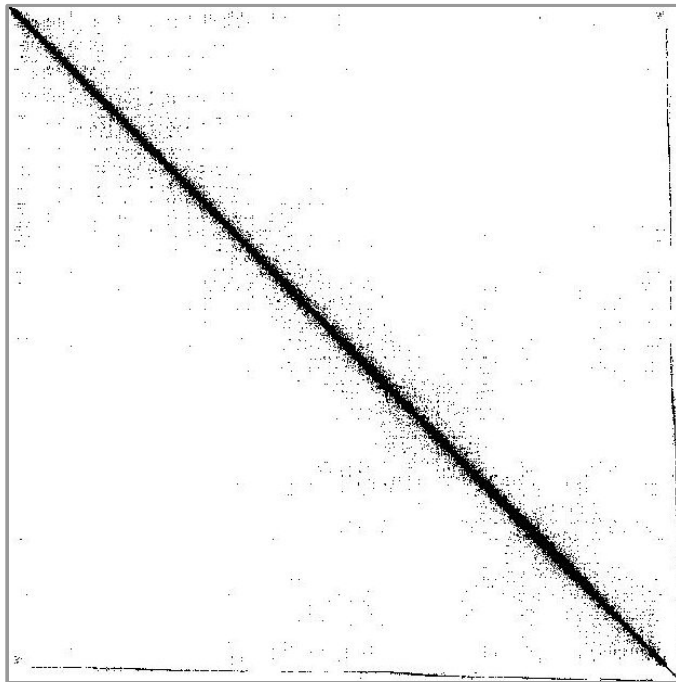
The online version of this article can be found at:

<http://hpc.sagepub.com/content/18/1/135>

Published by:

<http://www.sagepublications.com>

One Operation



.



=

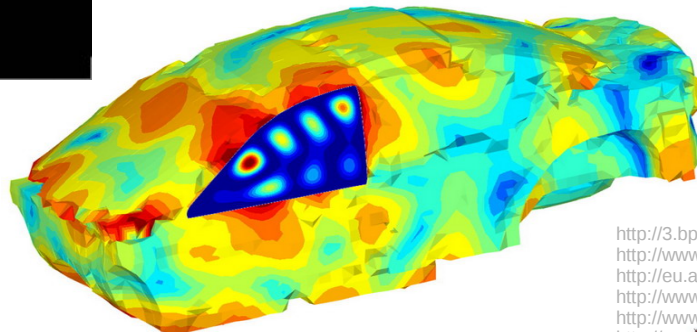
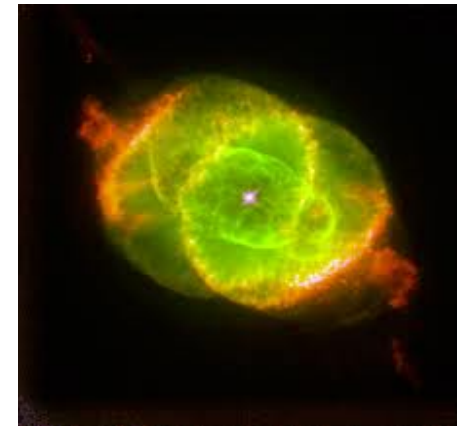
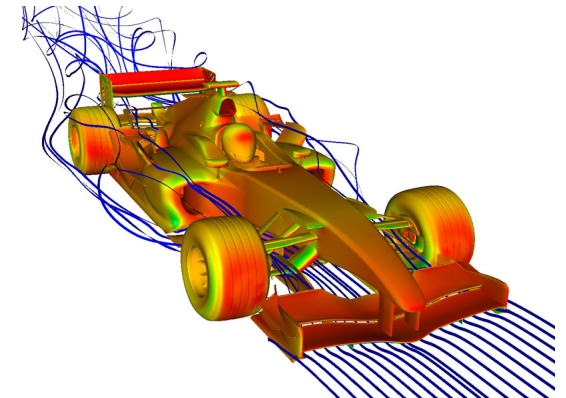
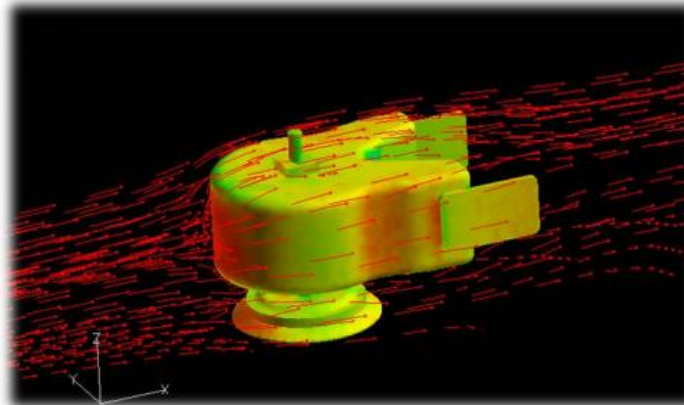
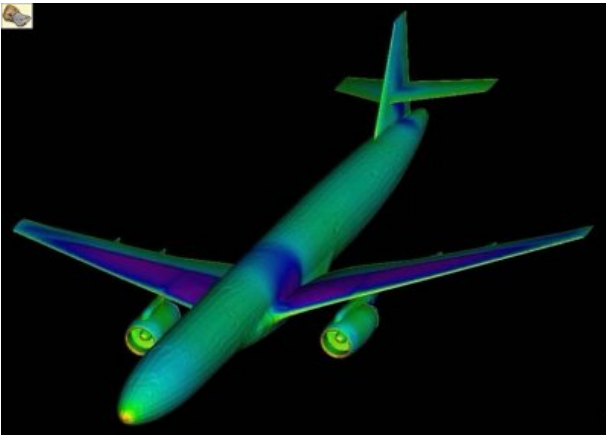


Motivation



TRENTO		BERGAMO	
CATEGORIA	ORARIO	CATEGORIA	ORARIO
DIR.	16:45	VEVEZIA P.N.	17:05
DIR.	16:45	TIRANO P.N.	17:25
DIR.	16:45	BERGAMO	17:45
DIR.	16:50	TORINO P.N.	17:05
DIR.	16:50	VERONA P.N.	17:25
DIR.	16:55	SESTRI P.N.	17:45
DIR.	17:00	PIACENZA	17:05

BINARIO NON DEFINITIVO
INFORMATION
SARMAZI



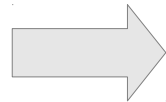
http://3.bp.blogspot.com/-jwj51xaDhsk/Thk3KtjWwsl/AAAAAAAAAOA/P8eNt0_MJUQ/s1600/Challenger2.gif
http://www.erneuerbareenergiequellen.com/pictures/other/oil_some_questions/oil_rig.jpg
<http://eu.art.com/products/p14342284-sa-i2886553/posters.htm?ui=BFBAB751660645AA8C02F859E5BAD142>
<http://www.aspsys.com/userfiles/image/fluent3.jpg>
http://www.bloodhoundssc.com/_db/_images/airliner_resized.jpg
<http://www.tif.be/images/documents/219.jpg>
http://www.onu.edu/files/images/alumni/Flow_around_object.jpg
http://t0.gstatic.com/images?q=tbn:ANd9GcQDP4JEXQNiqtR04rNdj2gBv8QpO1Sf1k2hcOMF9yXWqP_PCQb

Machines

Processor	Clock (MHz)	Data Cache sizes	DGEMV (MFLOPS)	DGEMM (MFLOPS)
Sun Ultra Sparc Ili	333	L1: 16 KB L2: 2 MB	58	425
Intel Pentium III-Mobile	800	L1: 16 KB L2: 256 MB	147	590
IBM Power 4	1300	L1: 64 KB L2: 1.5 MB L3: 32 MB	915	3500
Intel Itanium 2	900	L1: 16 KB L2: 256 KB L3: 3 MB	1330	3500

CSR: Compressed Sparse Row Format

3	0	0	5
0	1	7	0
0	0	2	0
0	0	0	4



Values:

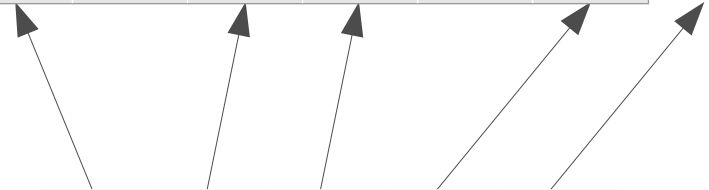
3	5	1	7	2	4
---	---	---	---	---	---

Column Index:

0	3	1	2	2	4
---	---	---	---	---	---

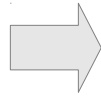
Row start Index:

0	2	3	5	6
---	---	---	---	---



Register-Blocking

3	0	0	5
0	1	7	0
0	0	2	0
0	0	0	4



Values:

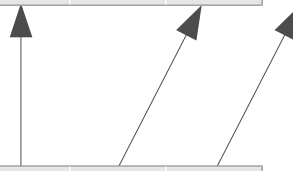
3	0	0	1	0	5	7	0	2	0	0	4
---	---	---	---	---	---	---	---	---	---	---	---

Column Index:

0	2	2
---	---	---

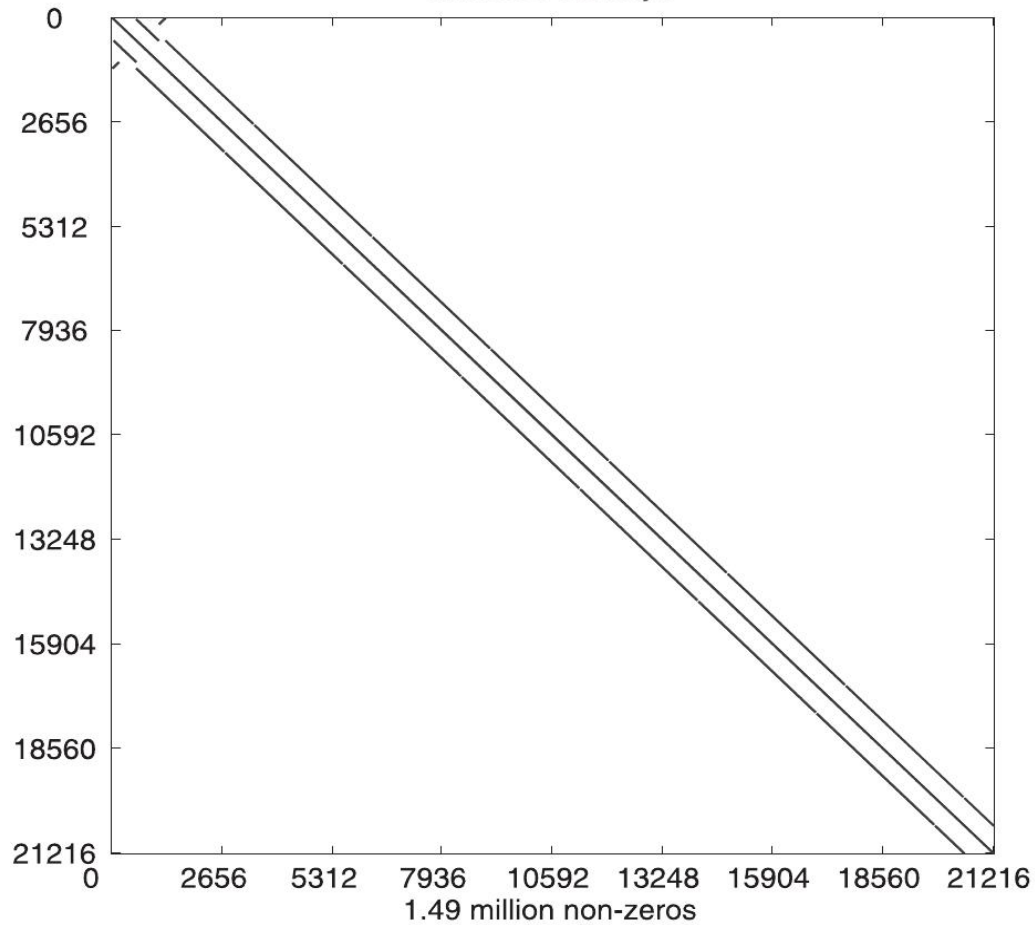
Row start Index:

0	2	3
---	---	---

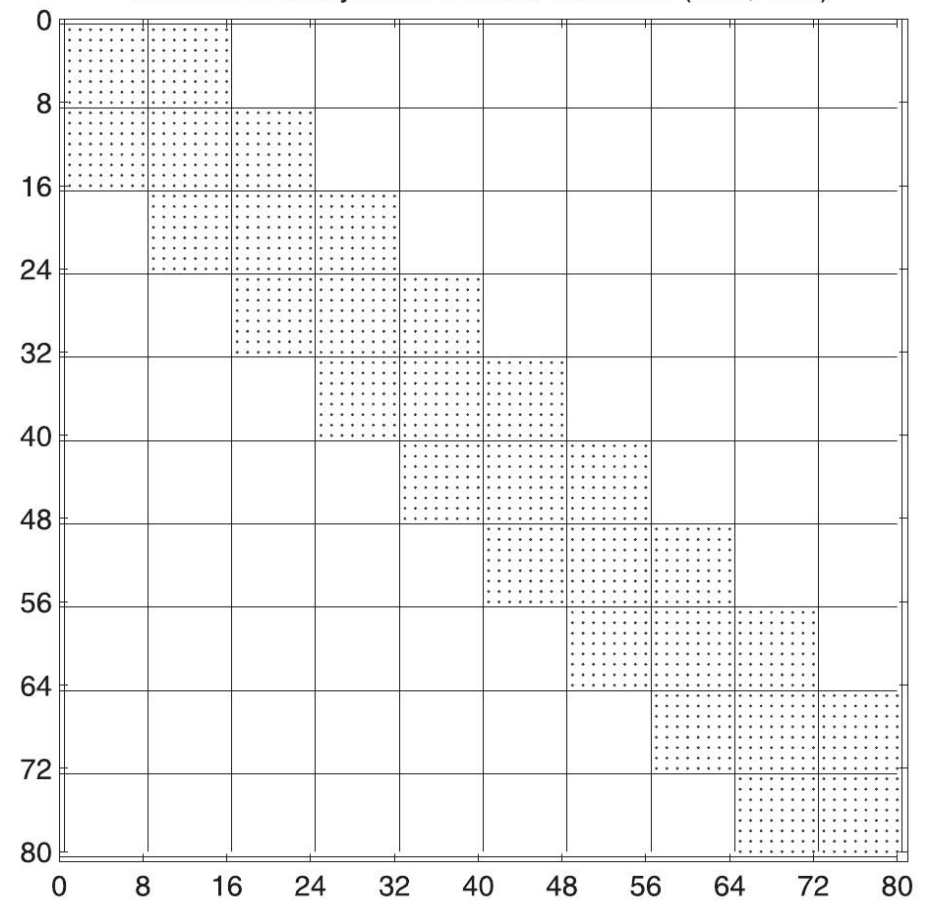


Example for Register-Blocking

Matrix 02-raefsky3

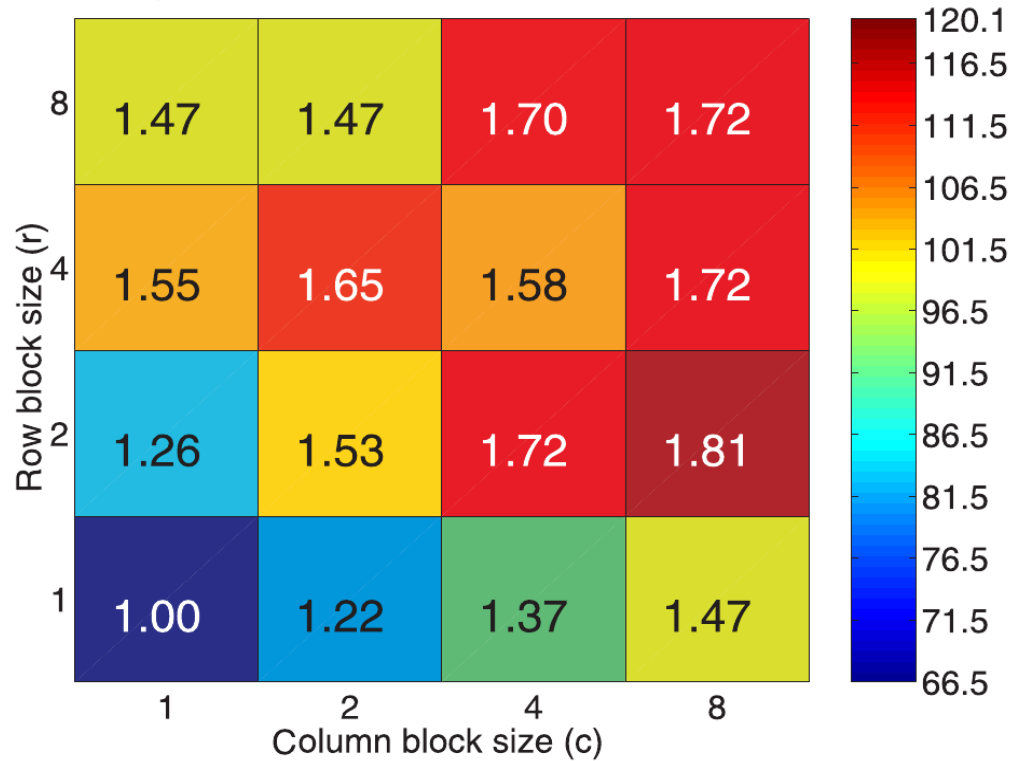


Matrix 02-raefsky3: 8x8 blocked submatrix (1:80, 1:80)

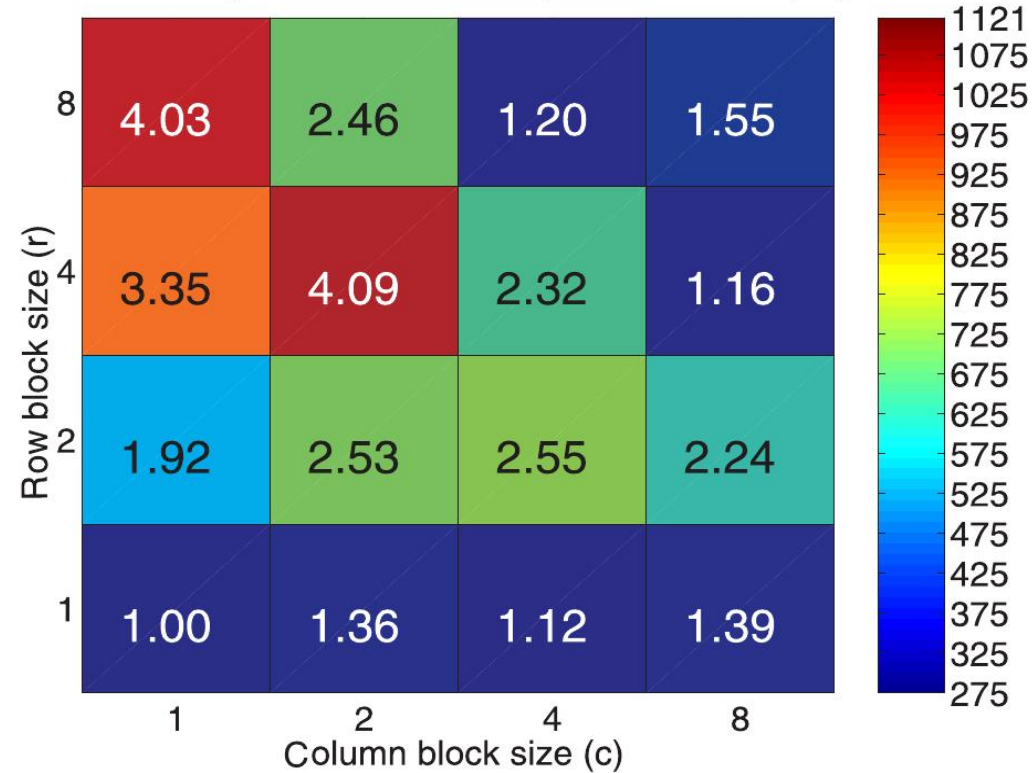


Example Results

#02-raefsky3.rua on Pentium III-Mobile [Ref=66.5 Mflop/s]

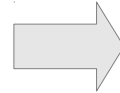


#02-raefsky3.rua on Itanium 2 [Ref=274.3 Mflop/s]



Performance Model: Fill-Overhead

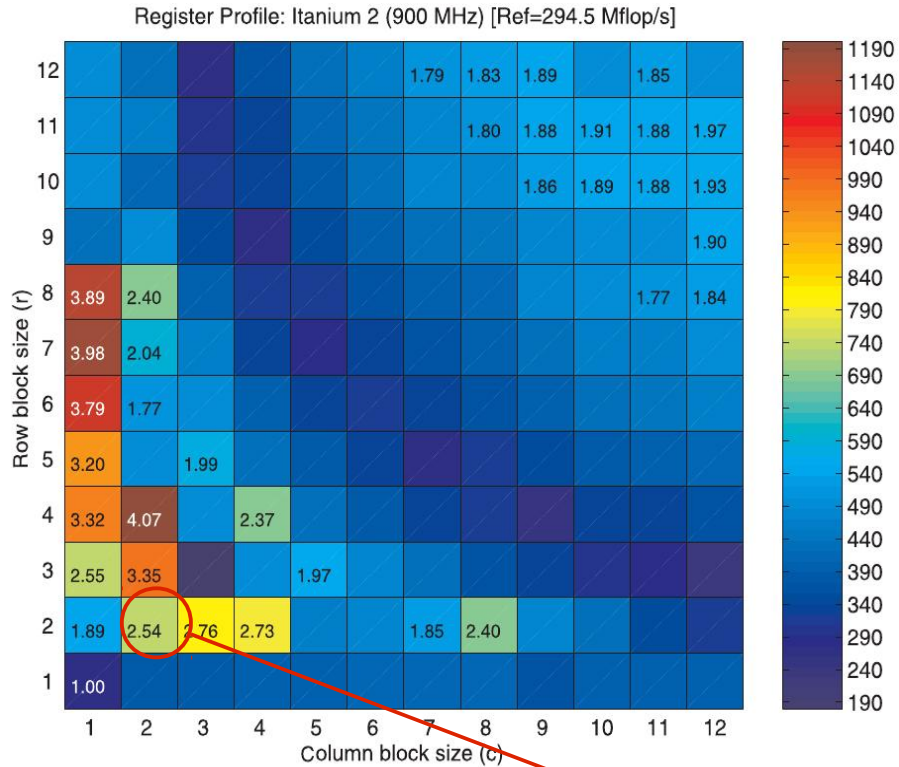
3	0	0	5
0	1	7	0
0	0	2	0
0	0	0	4



$$\frac{12}{6} = 2$$

Performance Model

Example on Intel Itanium 2 with 2x2 block-size:

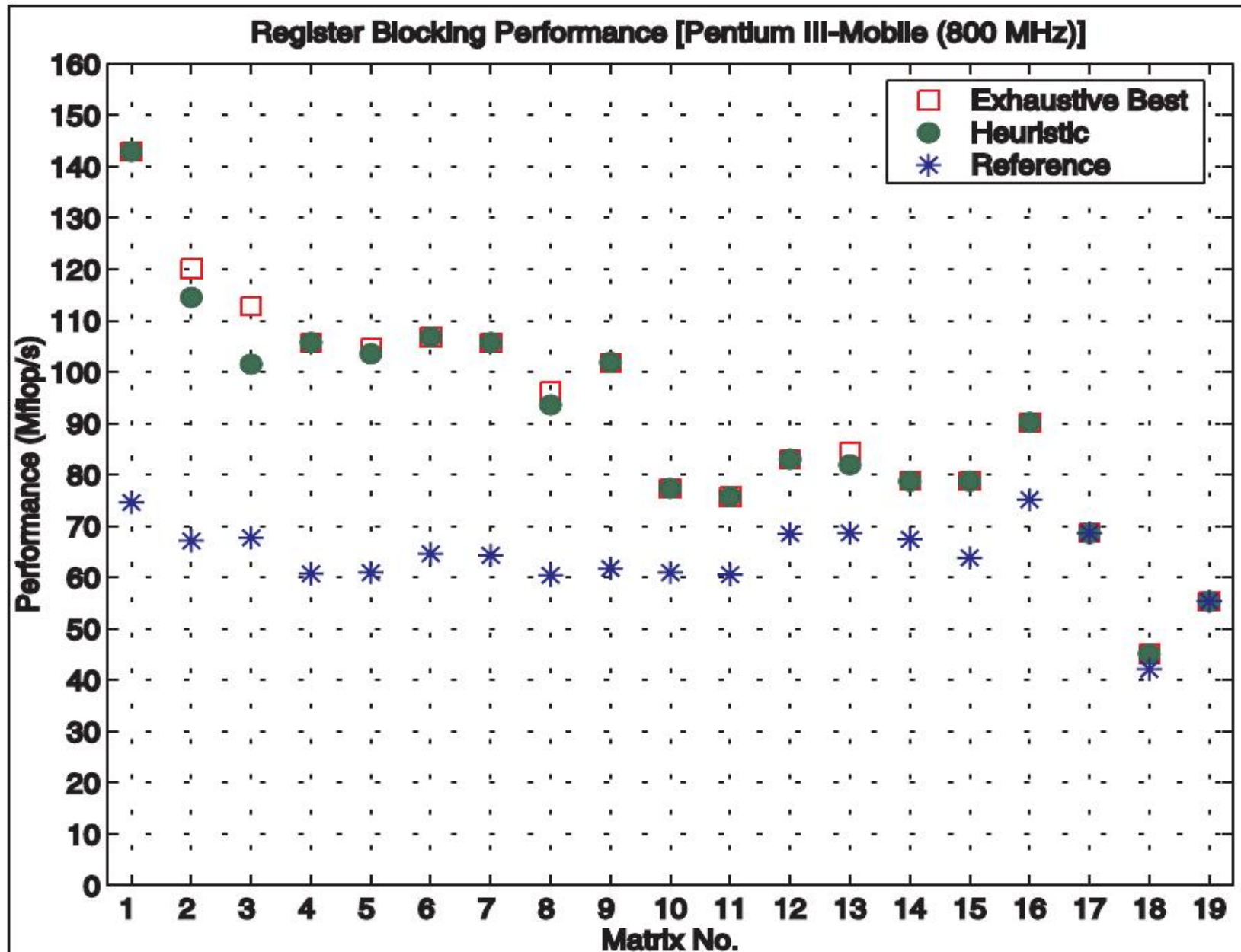


3	0	0	5
0	1	7	0
0	0	2	0
0	0	0	4

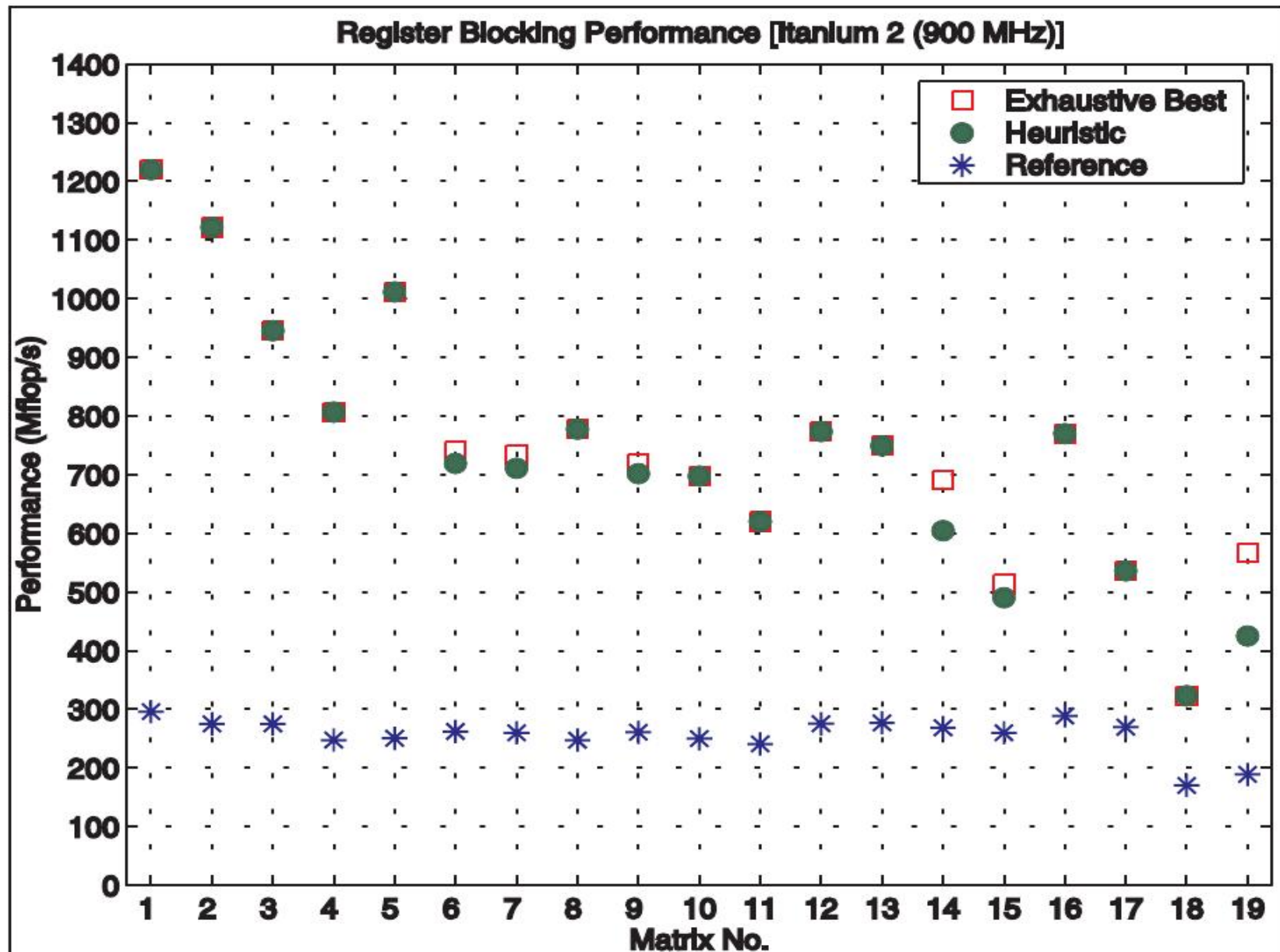
→ $\frac{12}{6} = 2$

$\frac{2.54}{2} = \underline{\underline{1.27}}$

Register-Blocking Speedup: Intel Pentium III-M

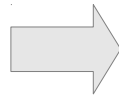


Register-Blocking Speedup: Intel Itanium 2



Cache-Blocking

3	0	0	5
0	1	7	0
0	0	2	0
0	0	0	4



Values:

3	1	5	7	2	4
---	---	---	---	---	---

Column Index:

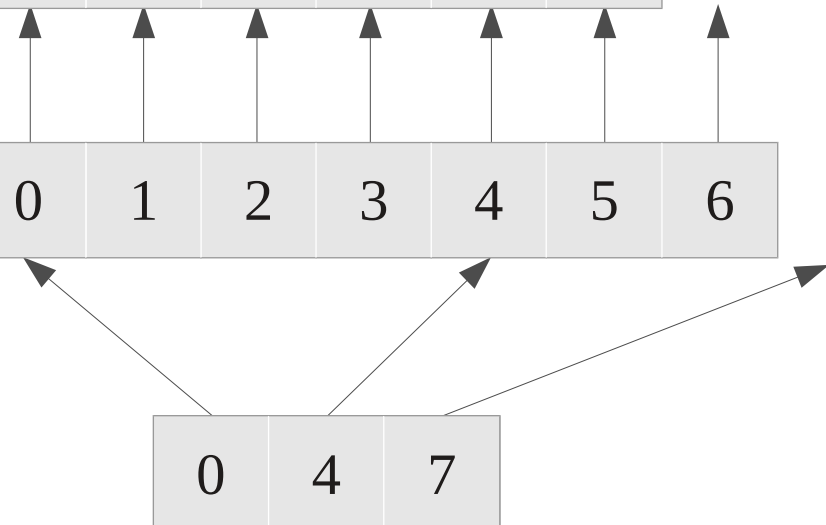
0	1	3	2	2	3
---	---	---	---	---	---

Block start Index:

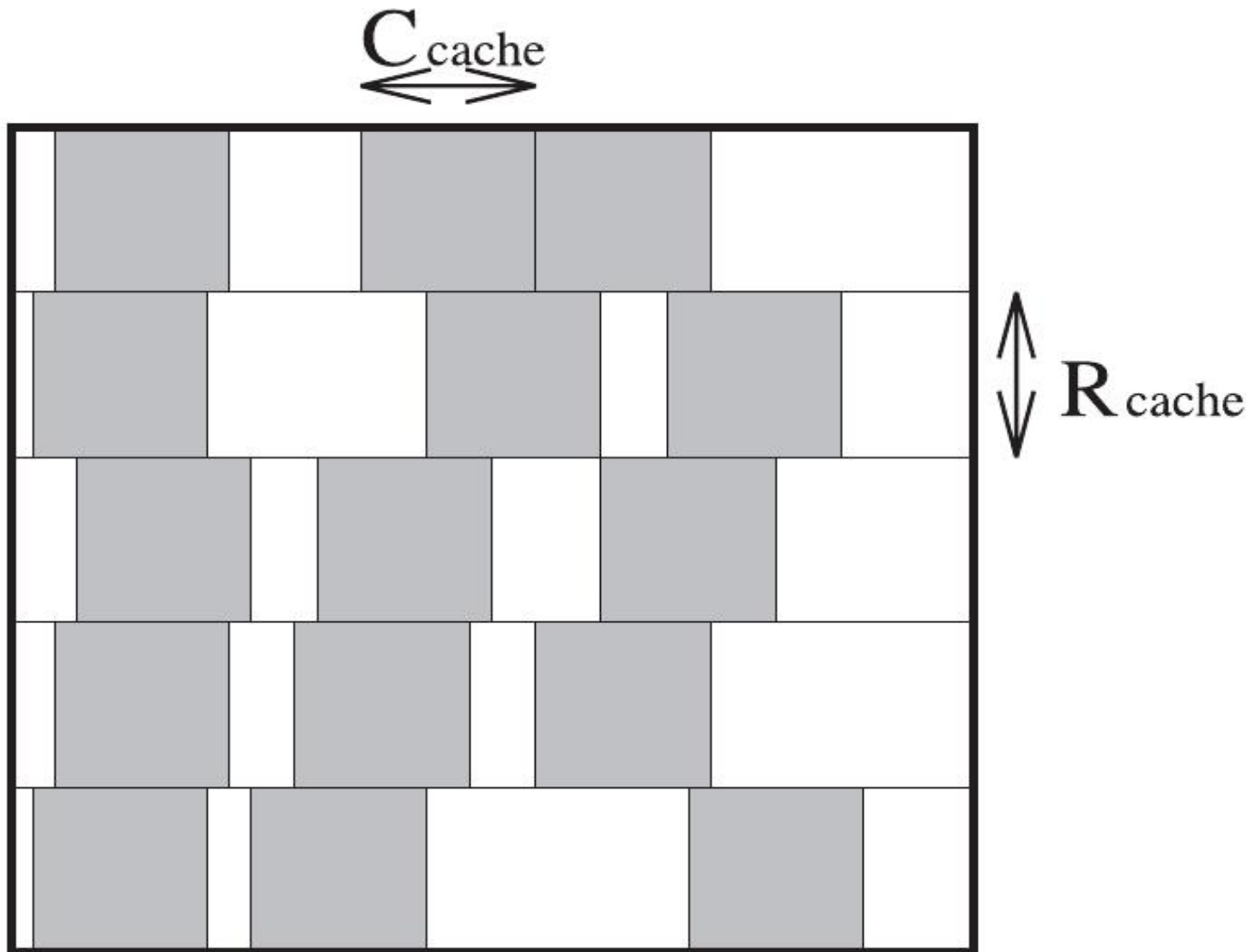
0	1	2	3	4	5	6
---	---	---	---	---	---	---

Block row start:

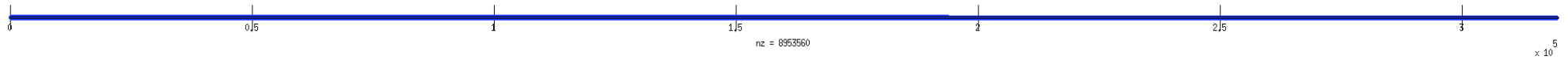
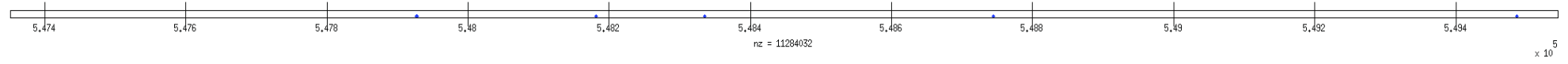
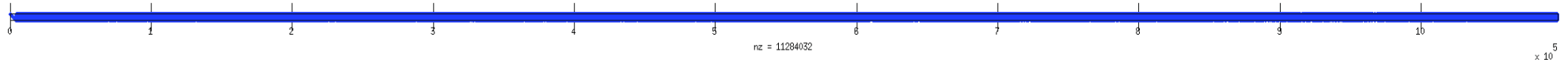
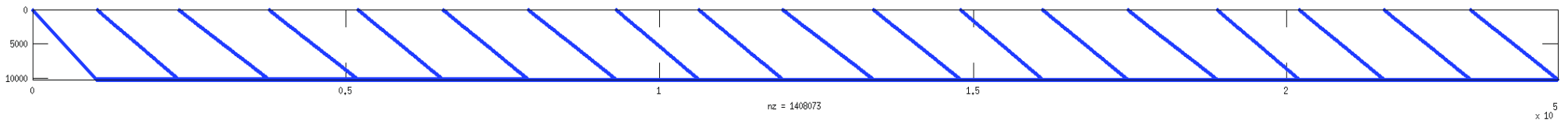
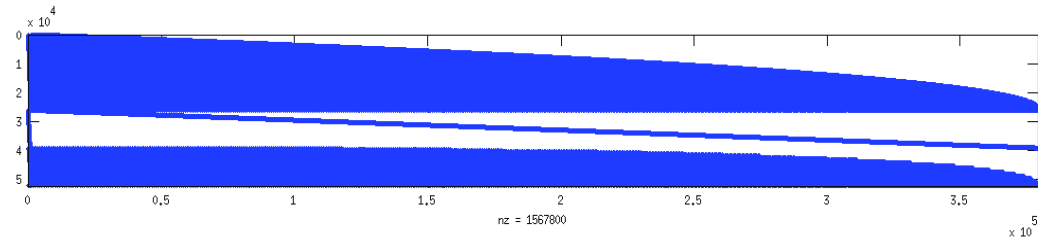
0	4	7
---	---	---



Cache-Blocking

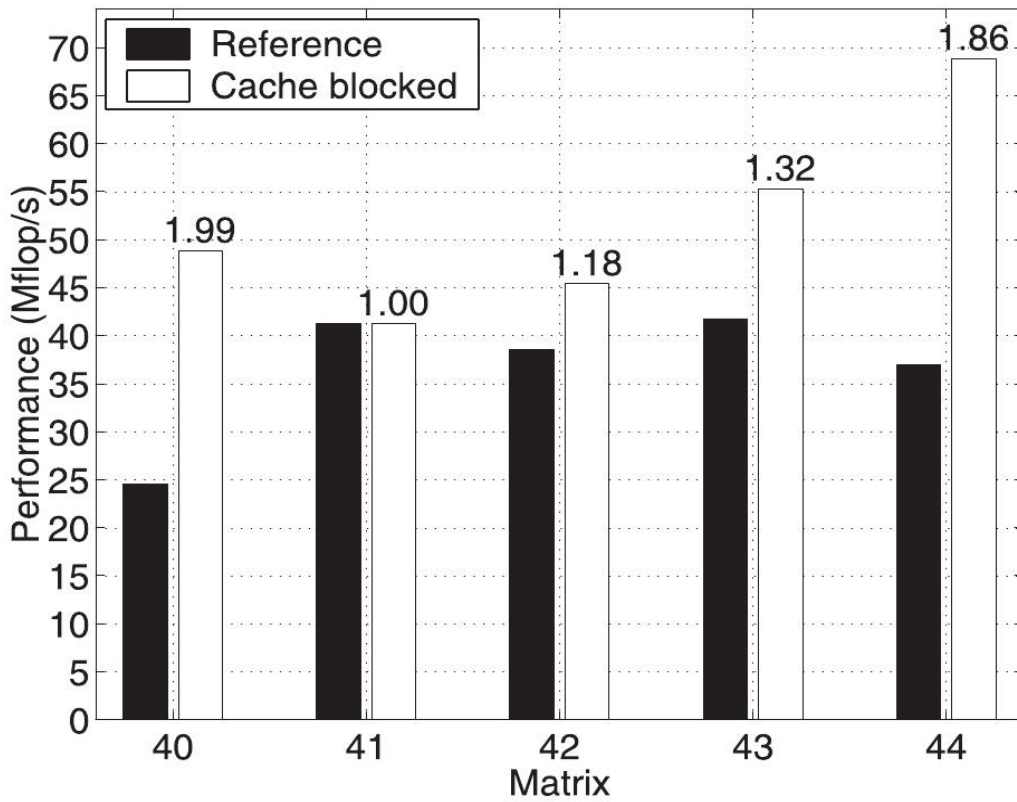


Benchmark Cache-Blocking

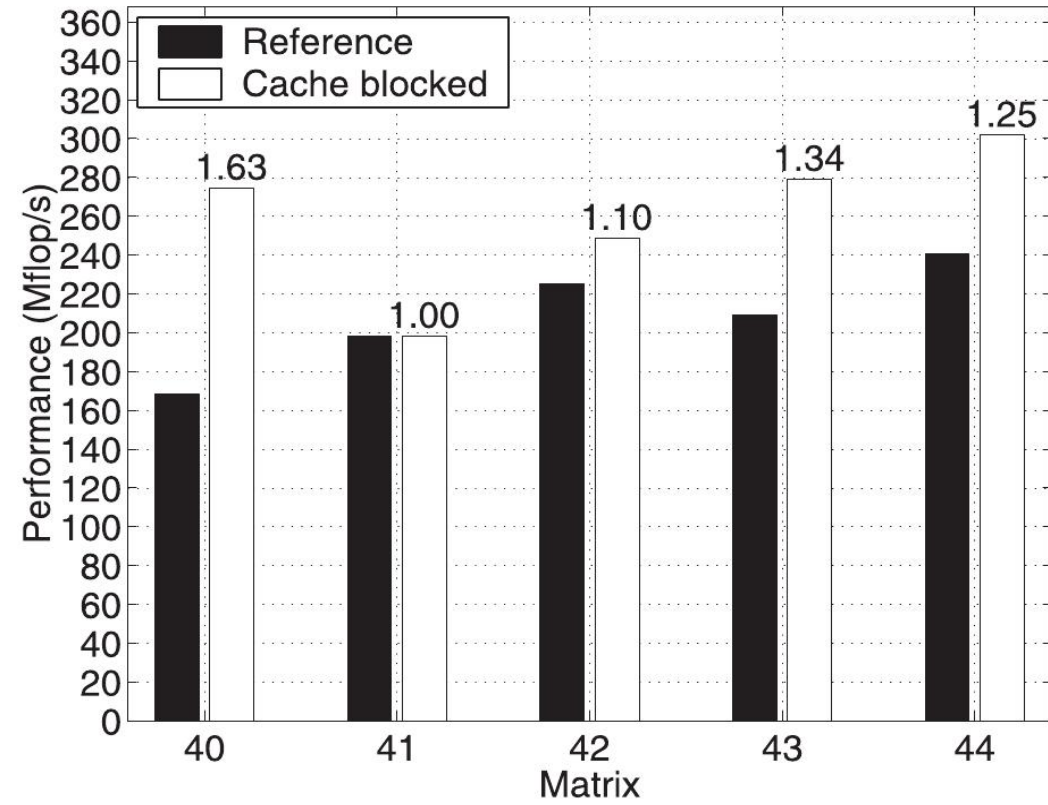


Cache-Blocking Speedup

Cache Blocking Performance: Intel Pentium III



Cache Blocking Performance: Intel Itanium 2



Multiple Vectors

3	0	0	5
0	1	7	0
0	0	2	0
0	0	0	4

·

u_0	v_0
u_1	v_1
u_2	v_2
u_3	v_3

=

y_{00}	y_{01}
y_{10}	y_{11}
y_{20}	y_{21}
y_{30}	y_{31}

$$(1) \quad 3 \cdot u_0 + 0 \cdot u_1 = y_{00}$$

$$(2) \quad 0 \cdot u_0 + 1 \cdot u_1 = y_{10}$$

...

$$(nz+1) \quad 3 \cdot v_0 + 0 \cdot v_1 = y_{01}$$

$$(nz+2) \quad 0 \cdot v_0 + 1 \cdot v_1 = y_{11}$$

$$(1) \quad 3 \cdot u_0 + 0 \cdot u_1 = y_{00}$$

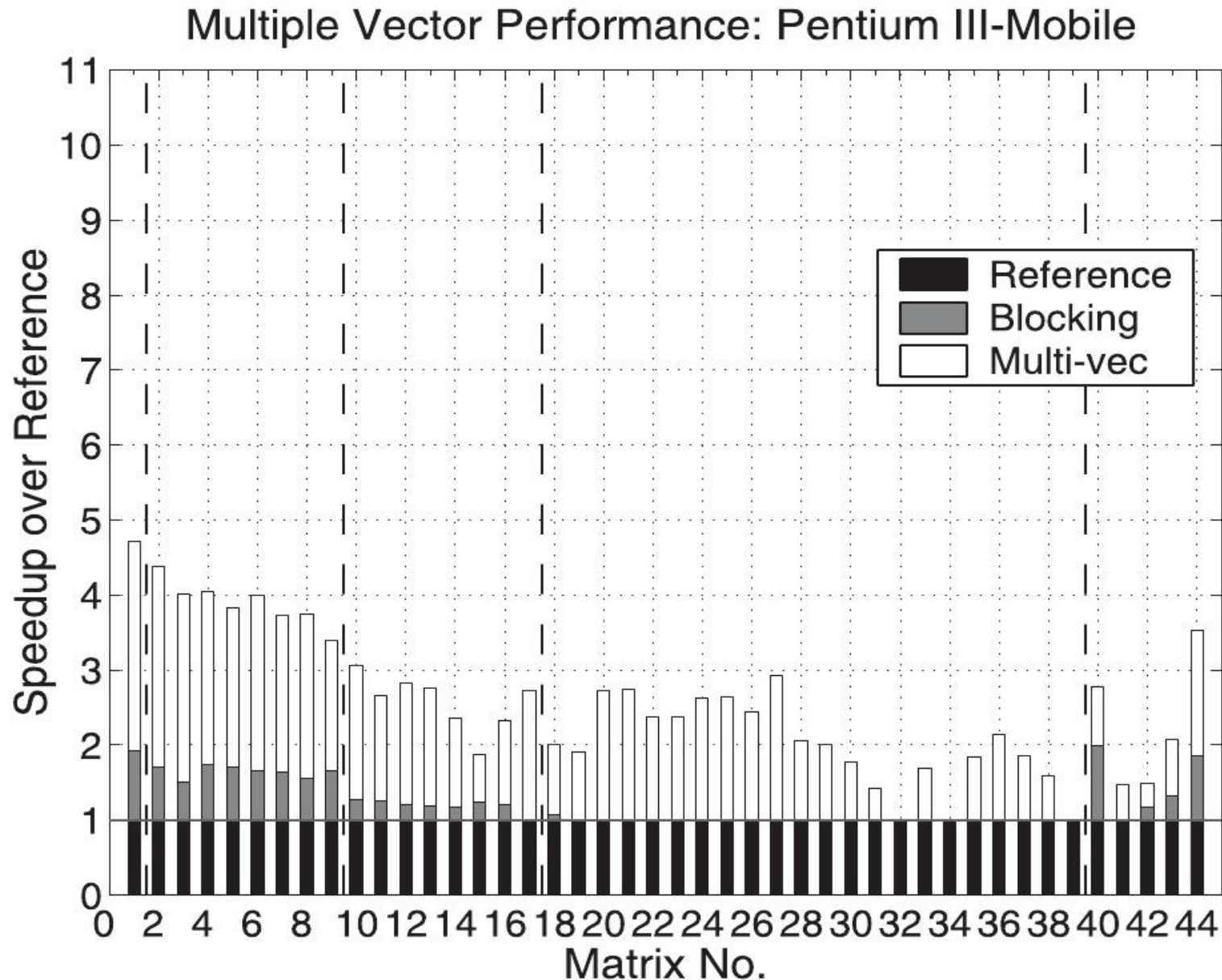
$$(2) \quad 0 \cdot u_0 + 1 \cdot u_1 = y_{10}$$

$$(3) \quad 3 \cdot v_0 + 0 \cdot v_1 = y_{01}$$

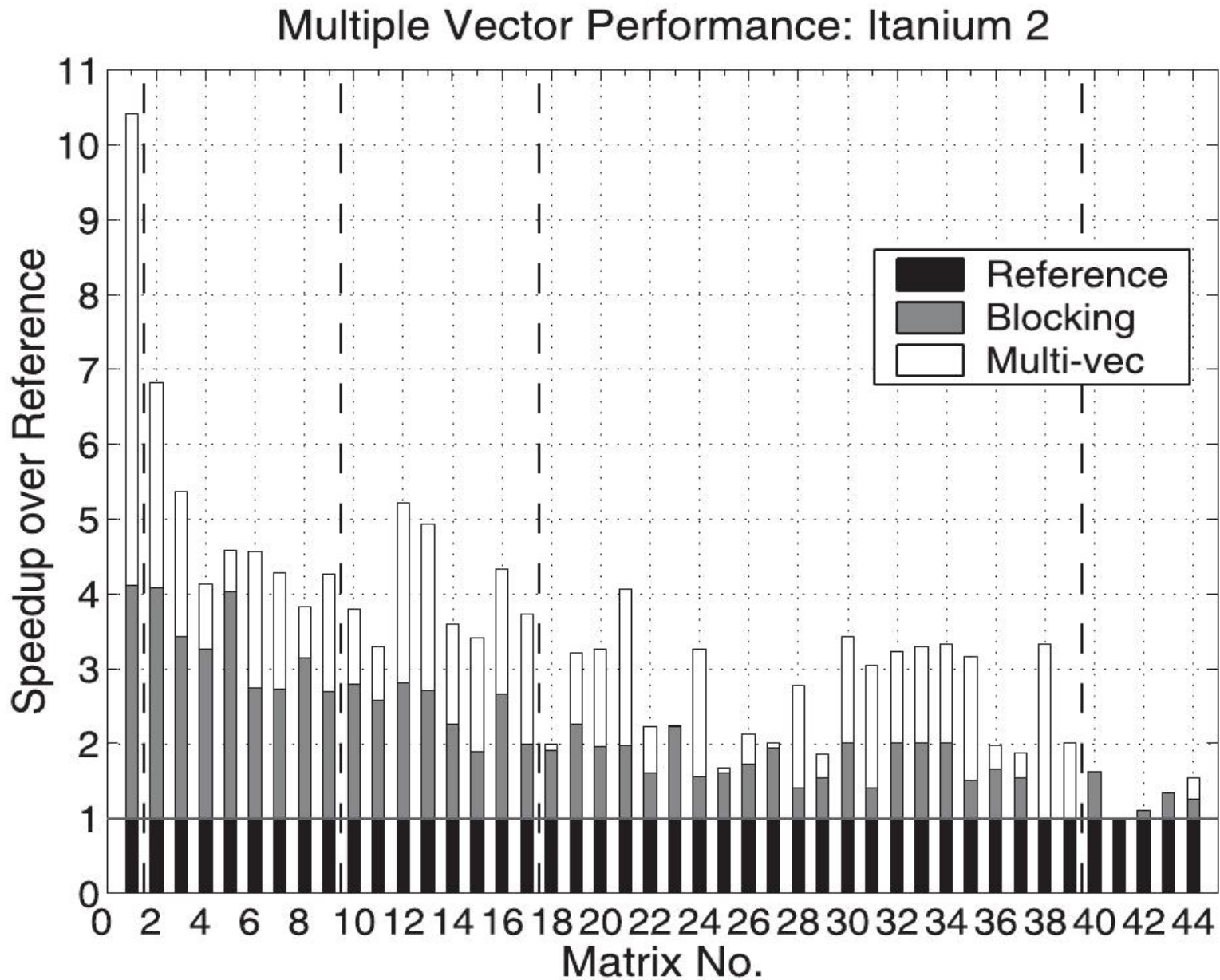
$$(4) \quad 0 \cdot v_0 + 1 \cdot v_1 = y_{11}$$

nz = number of non-zero elements in A

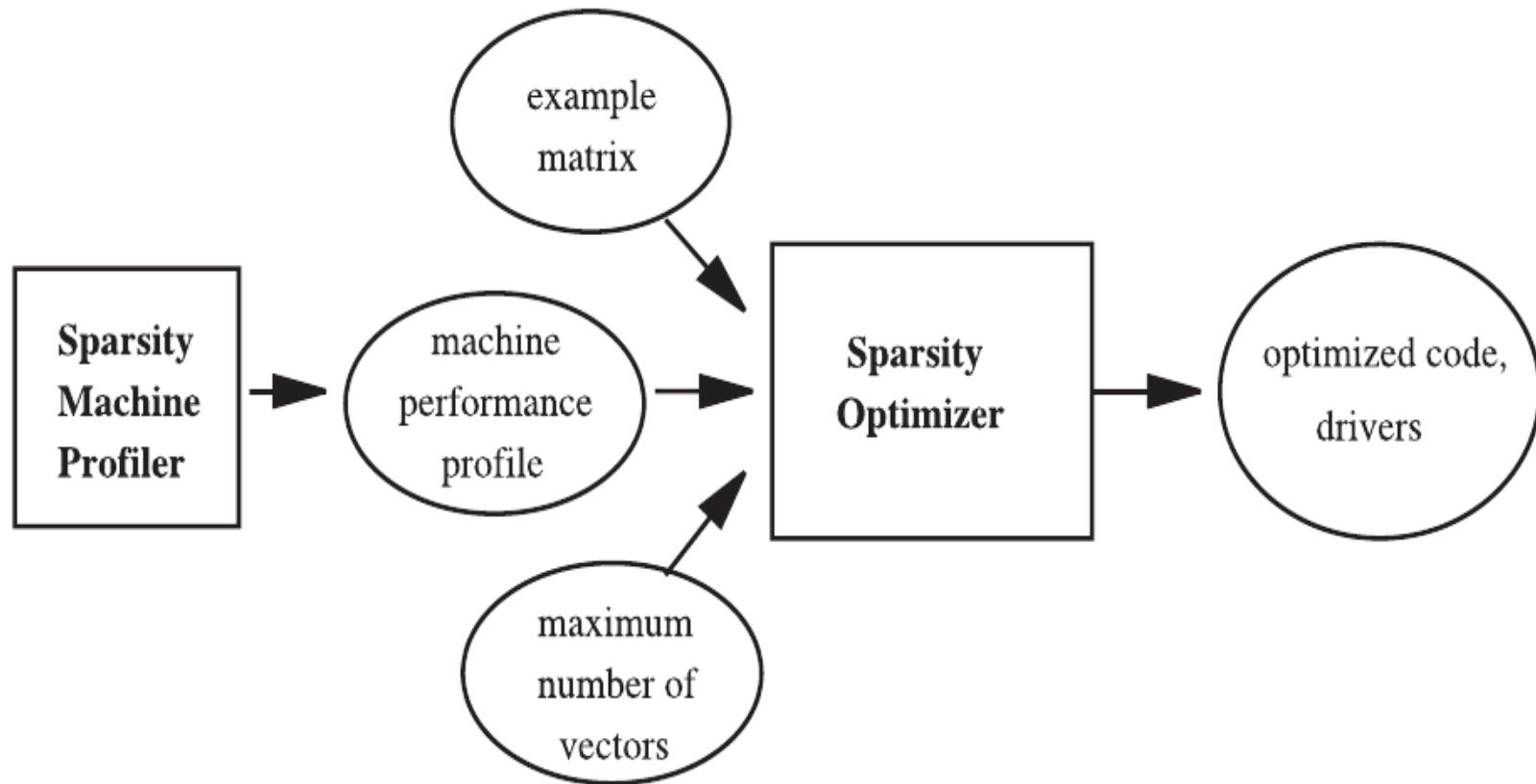
Multiple Vectors Speedup: Intel Pentium III-M



Multiple Vectors Speedup: Intel Itanium 2



SPARSITY System



Graph: Paper

Conclusion

- 4x improvement for register-blocking
- 2x for cache-blocking
- 10x for register-blocking combined with multiple vectors
- Lot of publications in reference to SPARSITY