

Adapting Linear Algebra Codes to the Memory Hierarchy Using a Hypermatrix Scheme

José R. Herrero, Juan J. Navarro

{josepr,juanjo}@ac.upc.edu

Computer Architecture Department
Universitat Politècnica de Catalunya



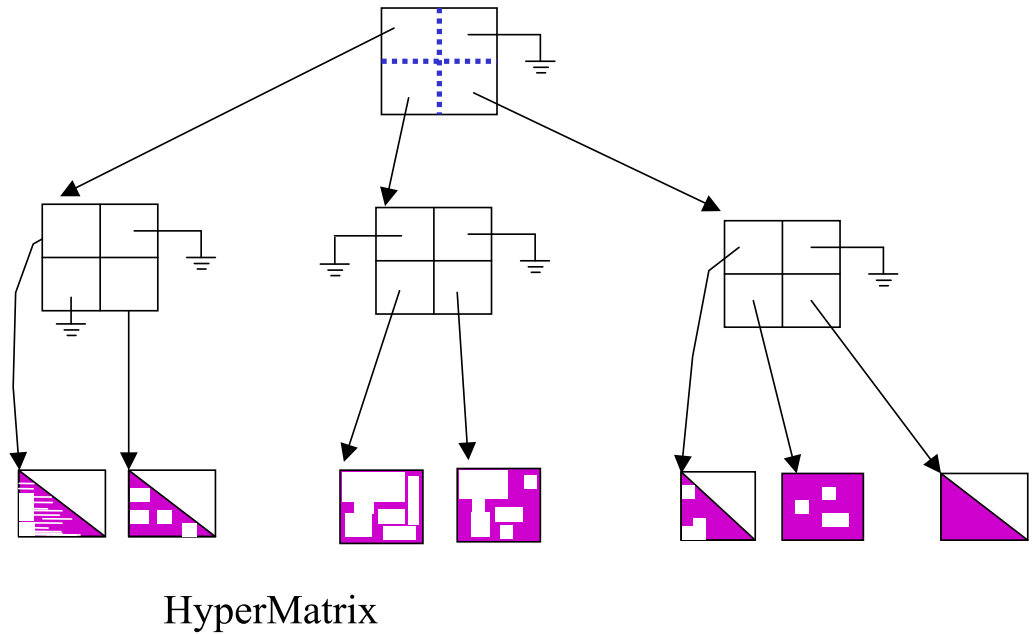
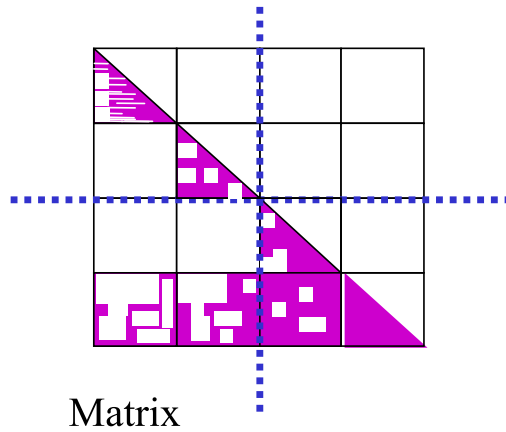
Barcelona

Spain

Outline

- Introduction
- Results
- Details
 - Inner kernel
 - Number of pointer levels and dimensions
 - Orthogonal Blocks
- Conclusions

Hypermatrix structure



Getting Efficient Kernels

Goal: The sparse code requires efficient operation on small data submatrices

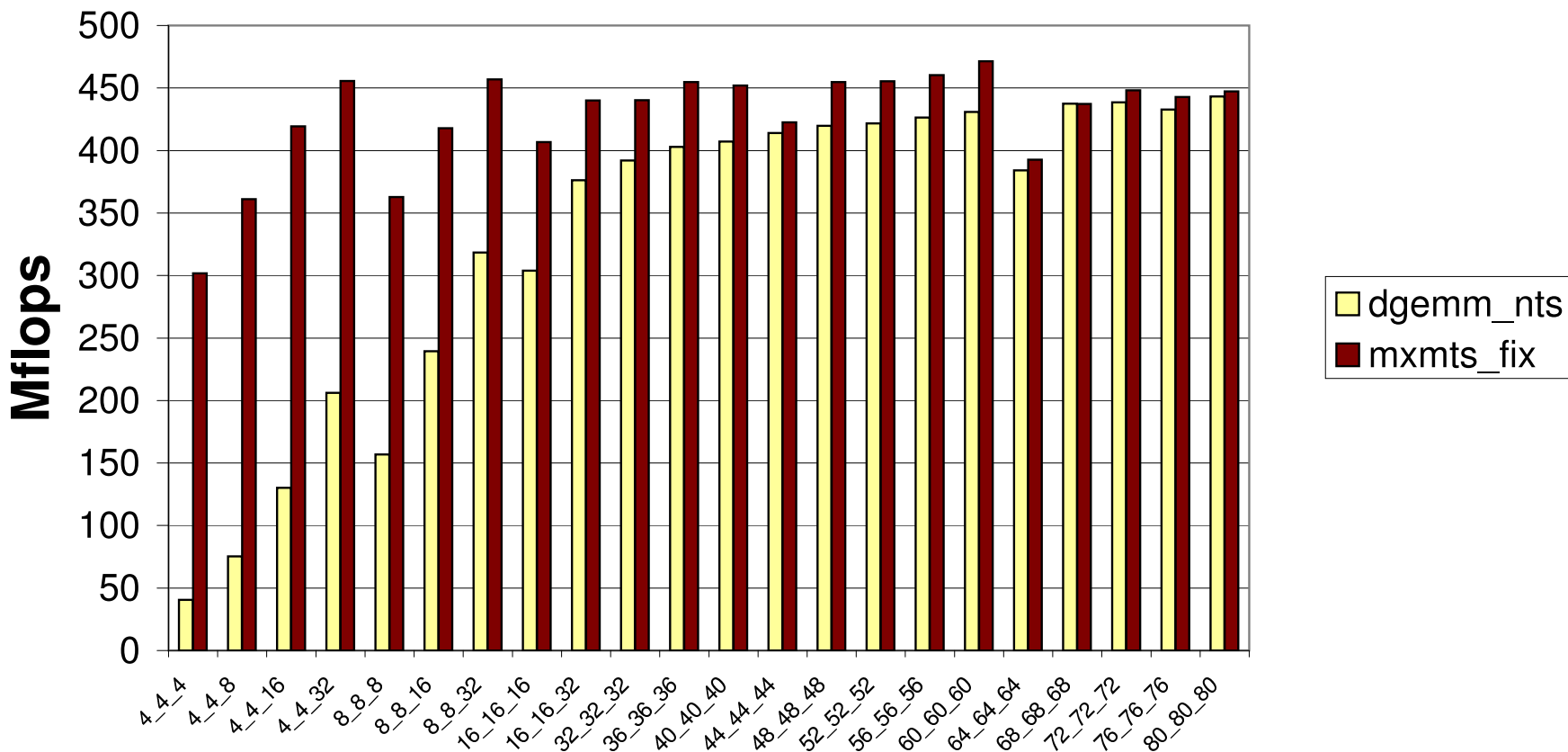
- Reduce data submatrix size while keeping good BLAS3 performance

Idea [Euro-Par'03]:

- Fix parameters at compilation time
- Choose best algorithm
 - Loop unrolling factors
 - Loop orders

Matrix multiplication performance on small matrices: R10K

$$C = C - A * B^t$$



R10000 250 MHz (500 Mflops peak)

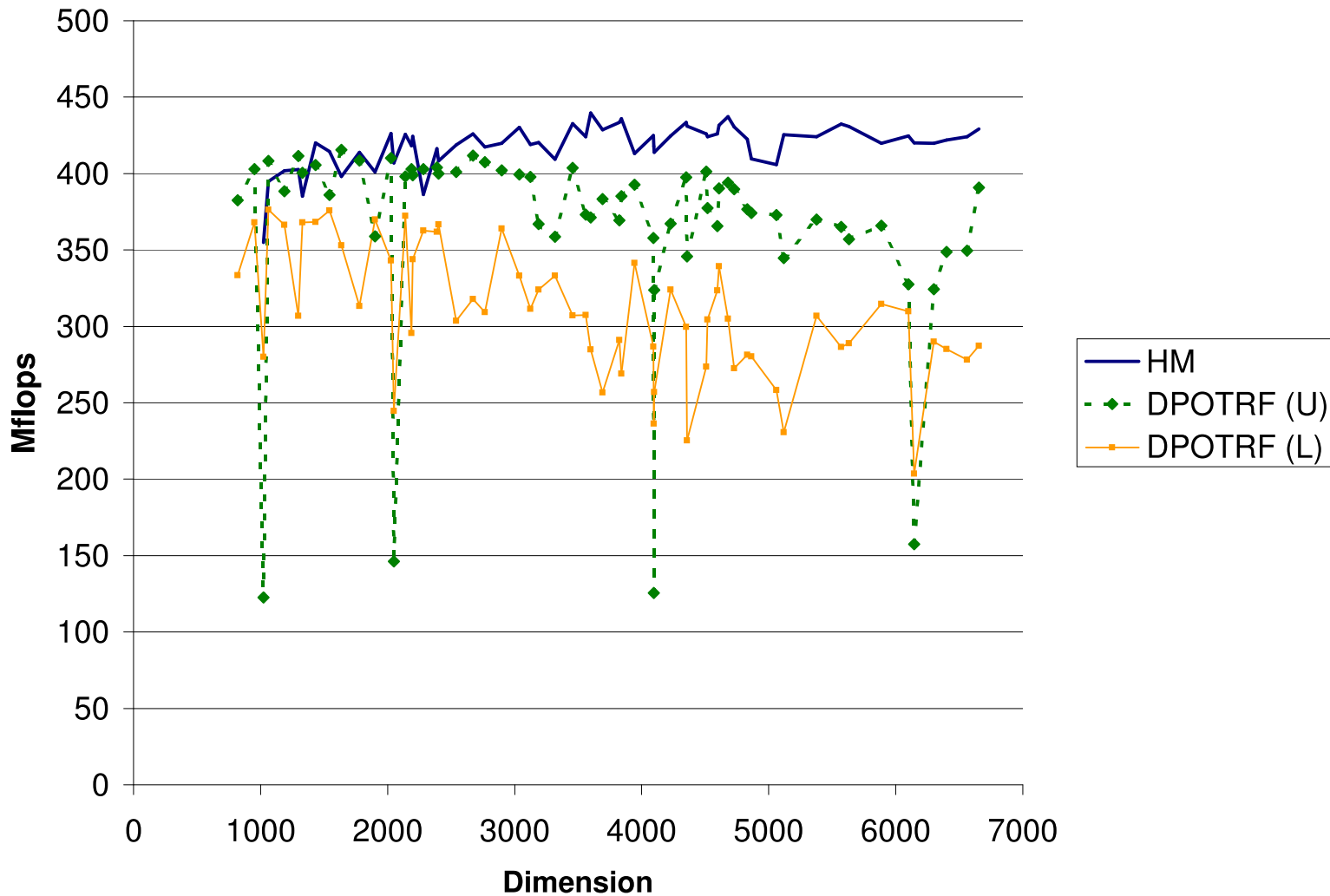
Operations on dense matrices

- Can this be used to produce high performance routines operating on **dense** matrices?

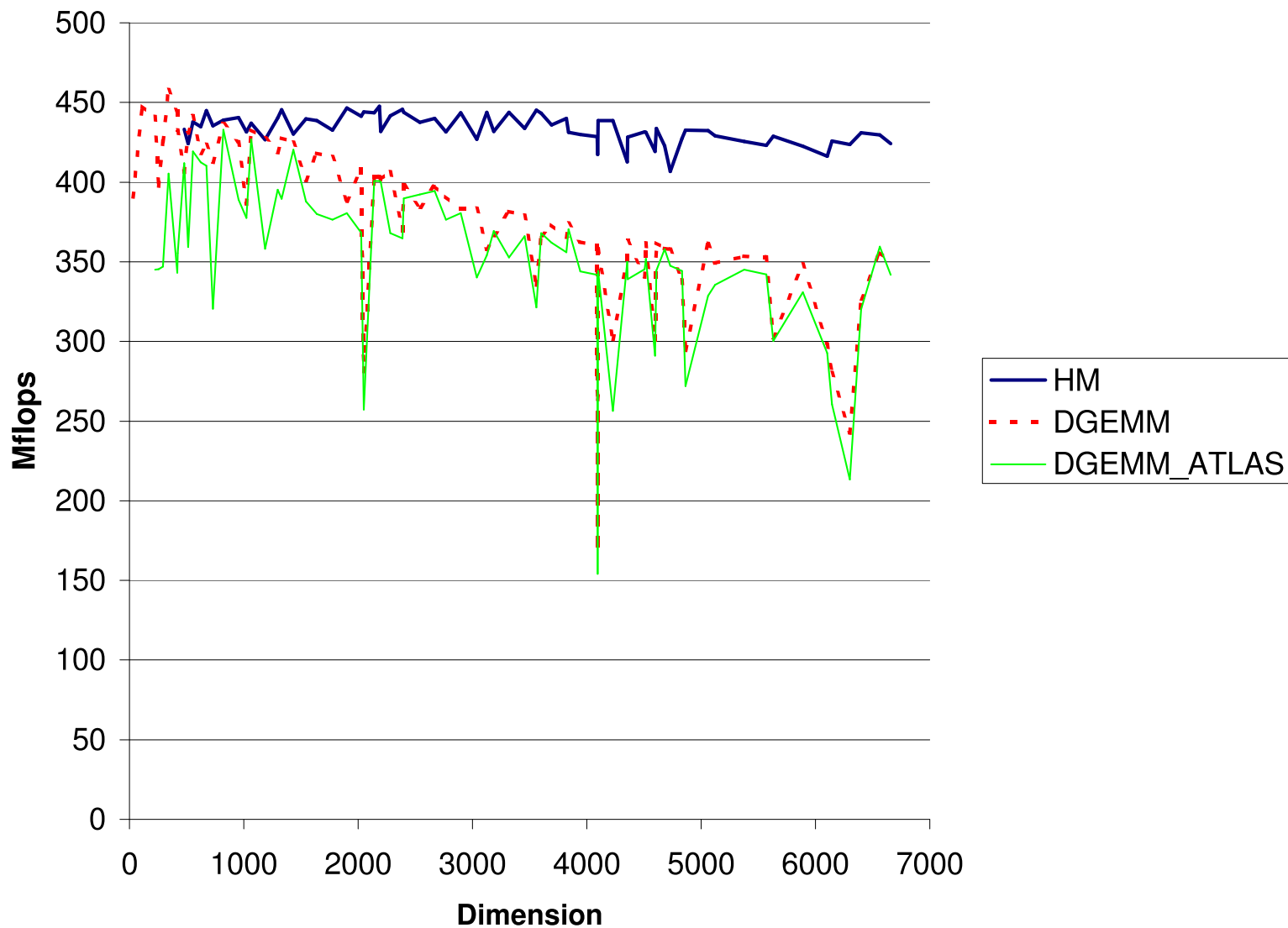
Outline

- Introduction
- **Results**
- Details
 - Inner kernel
 - Number of pointer levels and dimensions
 - Orthogonal Blocks
- Conclusions

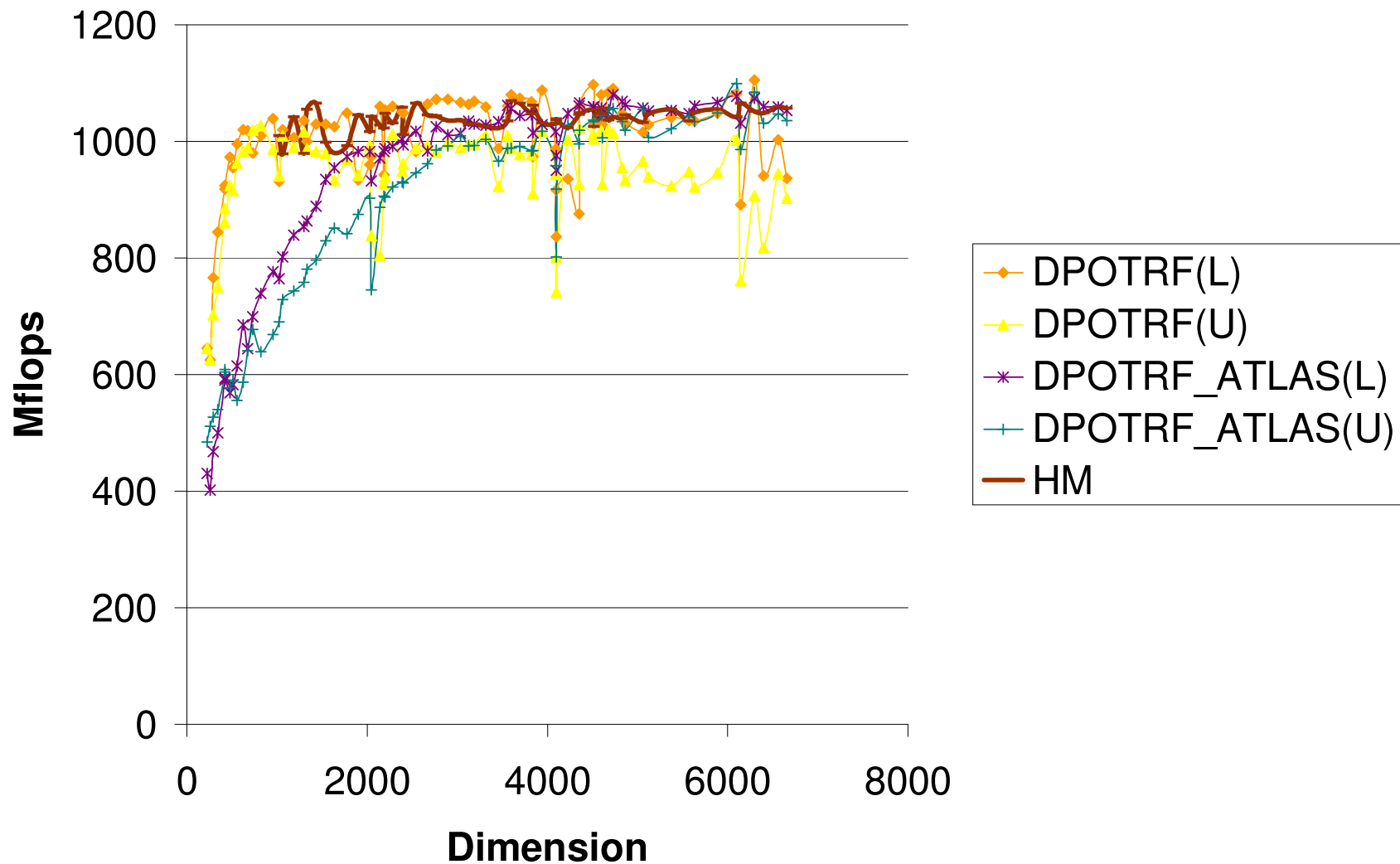
R10000: Dense Cholesky



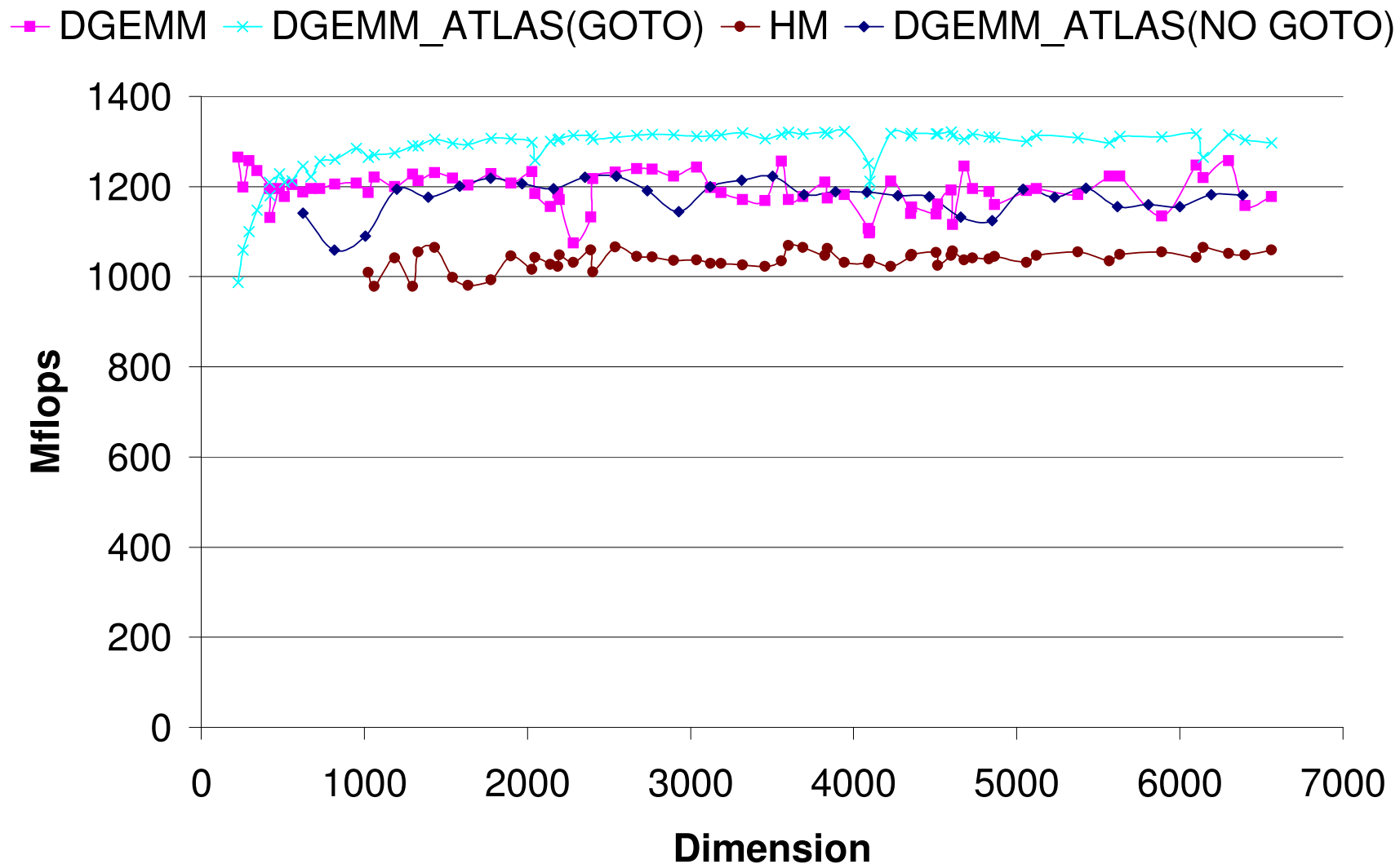
R10000: Dense Matrix Multiplication



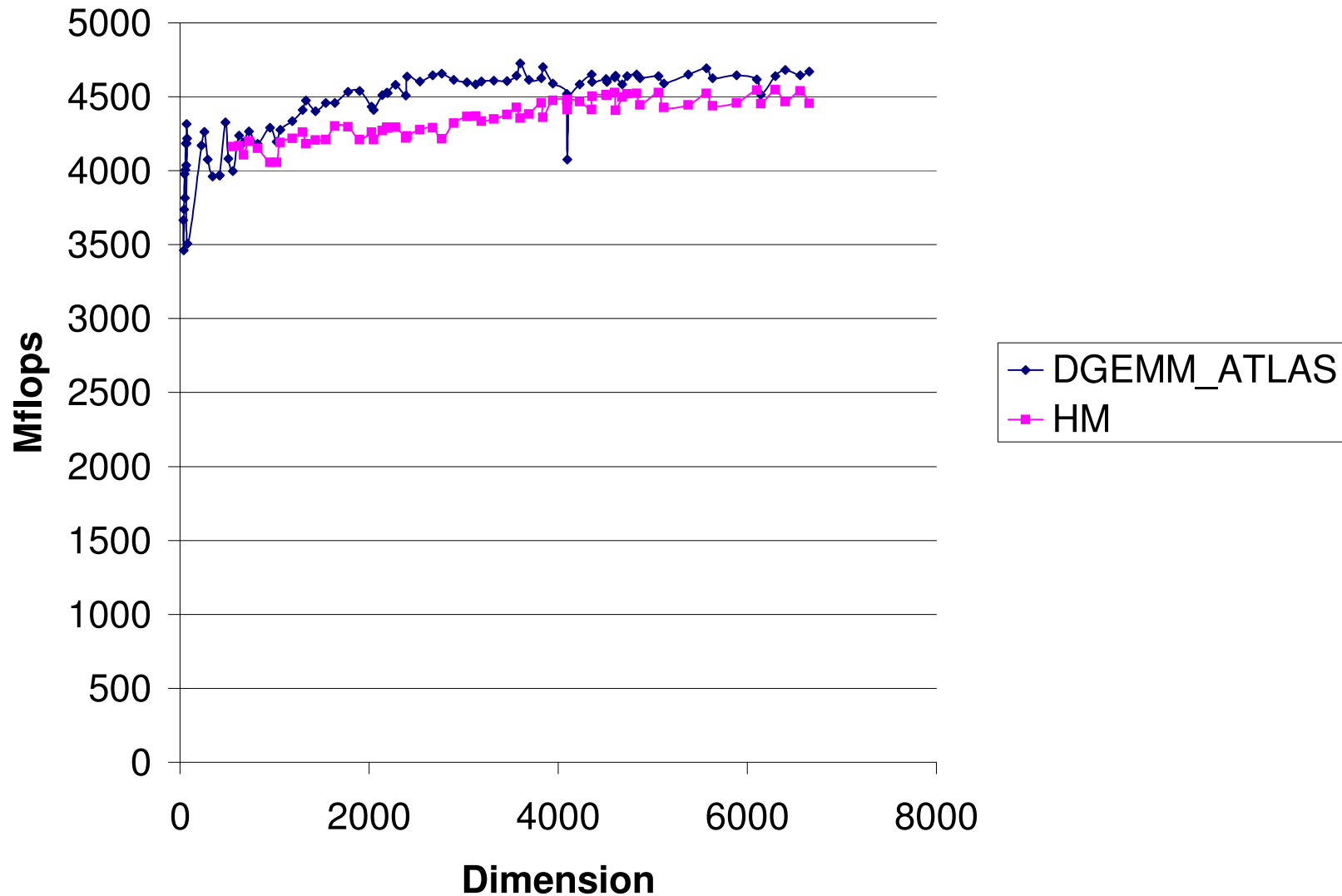
Alpha 21264: Dense Cholesky



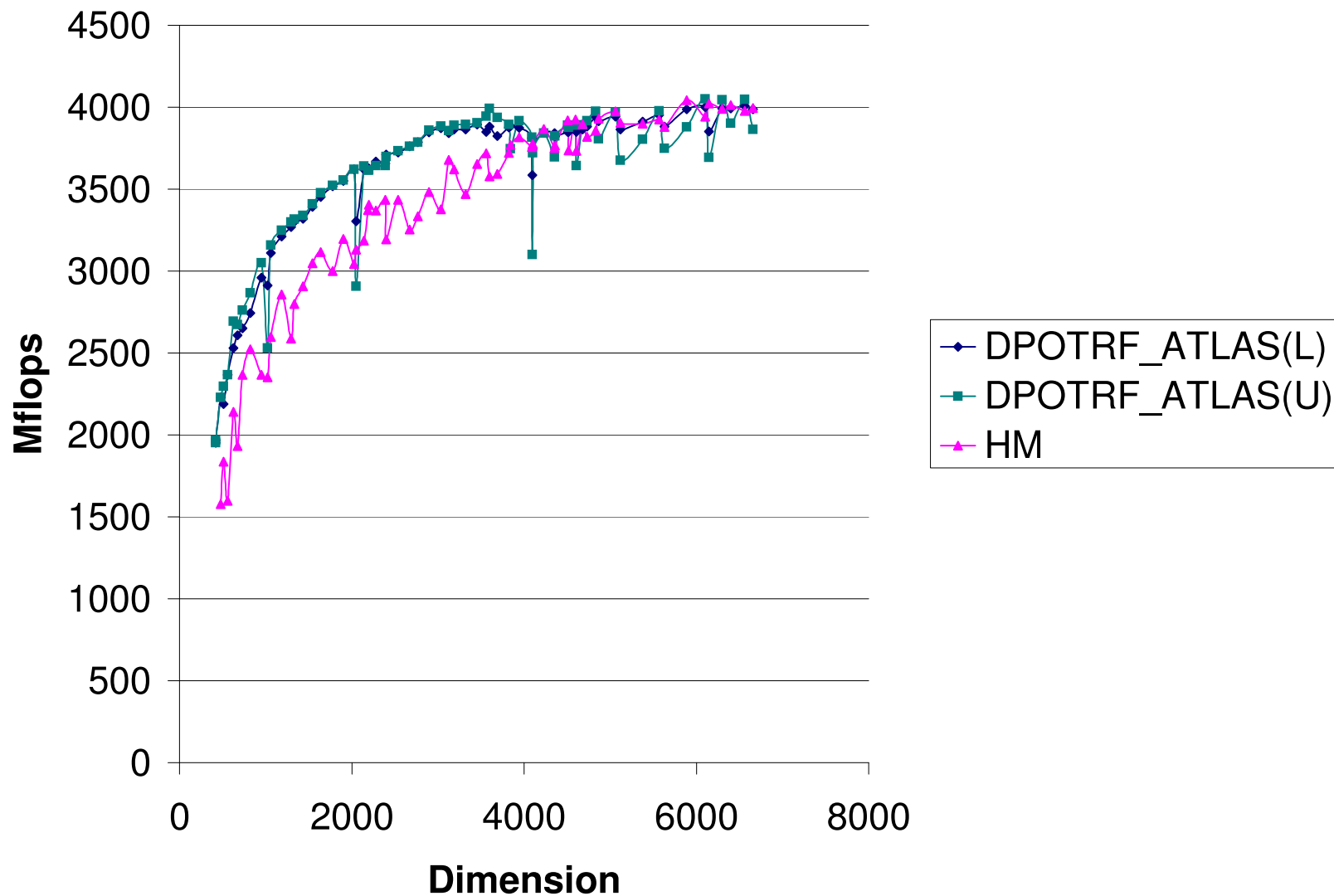
Alpha 21264: Dense Matrix Multiplication



Itanium2: Dense Cholesky



Itanium2: Dense Matrix Multiplication

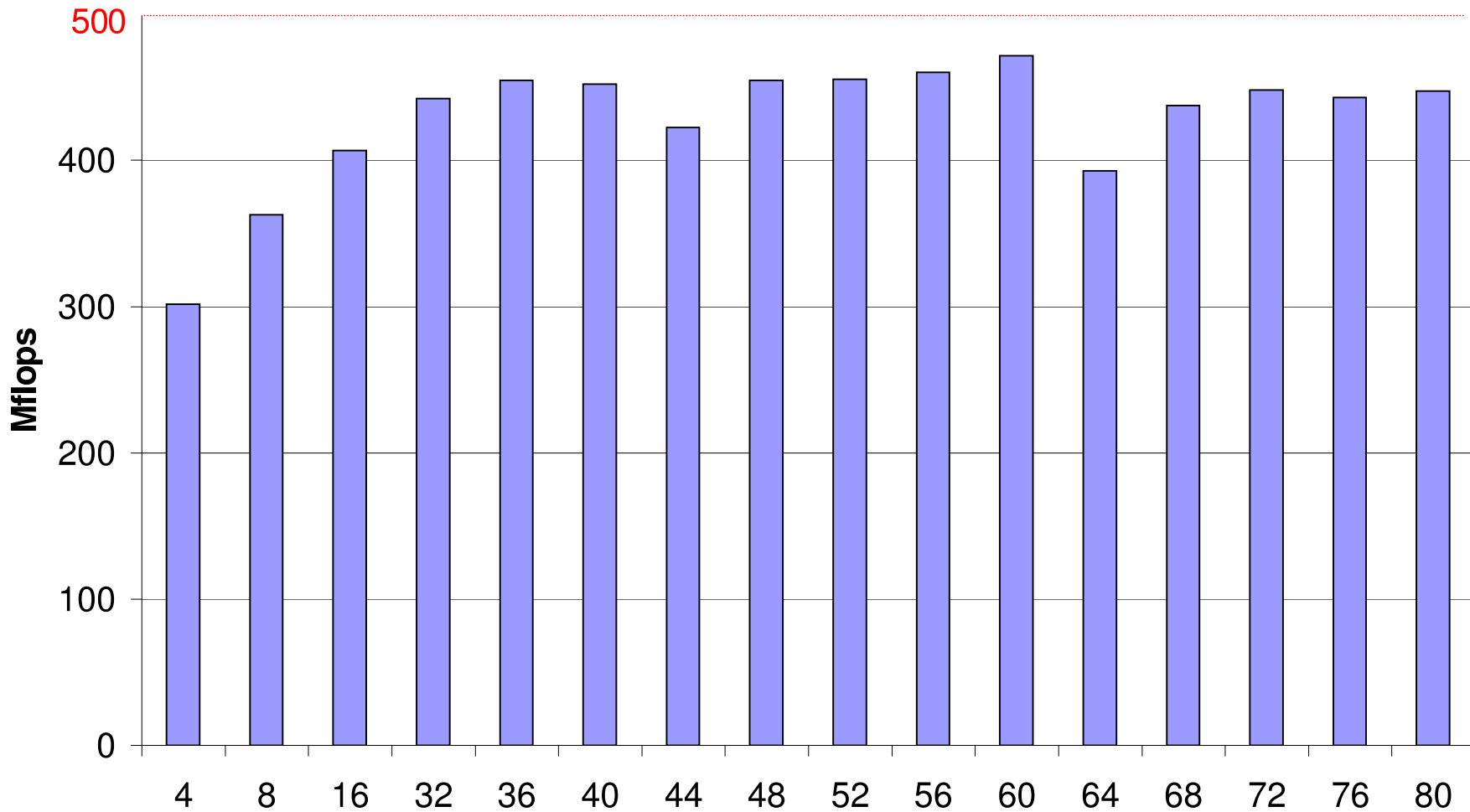


Outline

- Introduction
- Results
- **Details**
 - **Inner kernel**
 - Number of pointer levels and dimensions
 - Orthogonal Blocks
- Conclusions

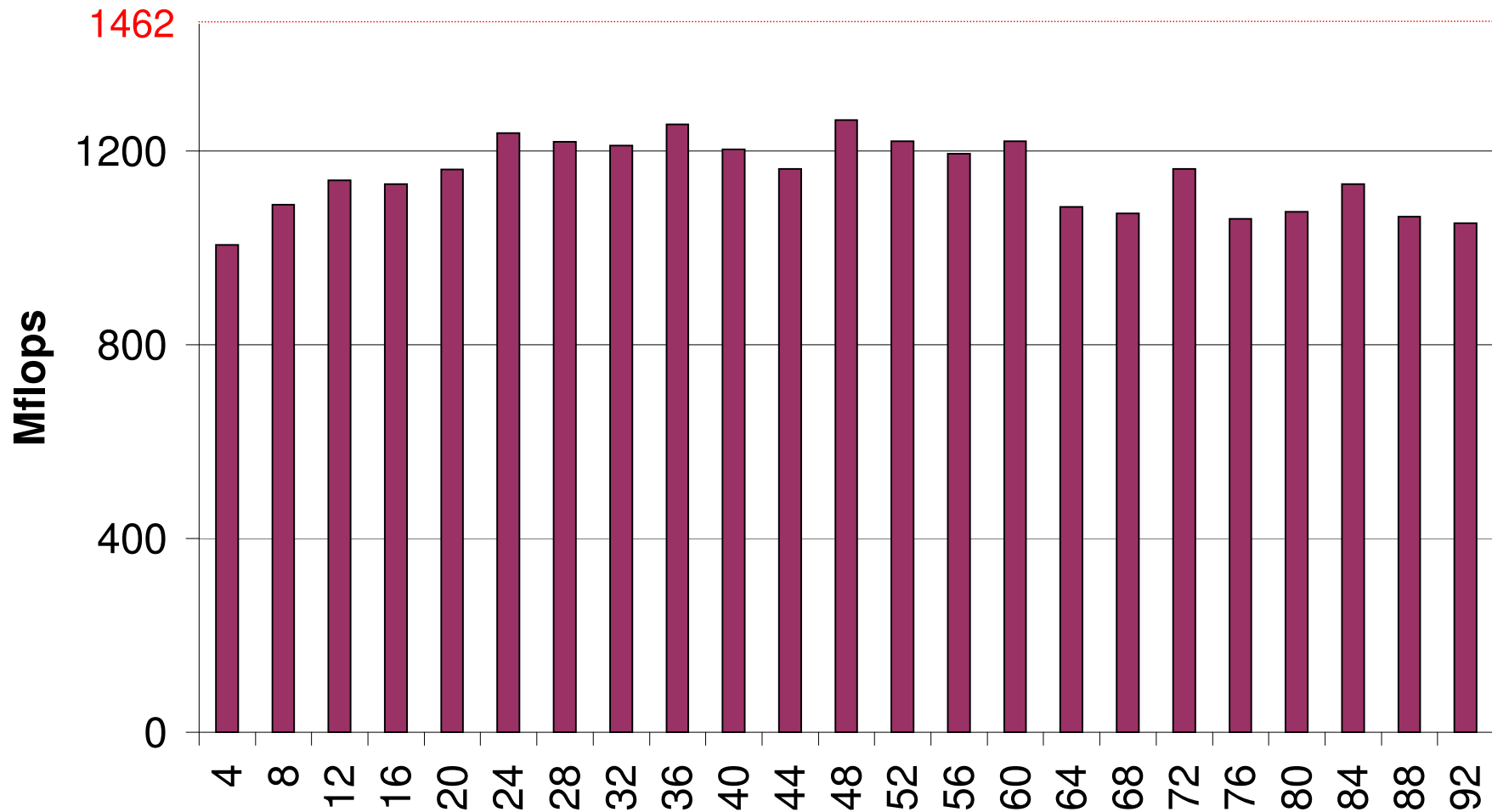
Matrix multiplication performance on small matrices: R10K

R10000 @ 250 MHz



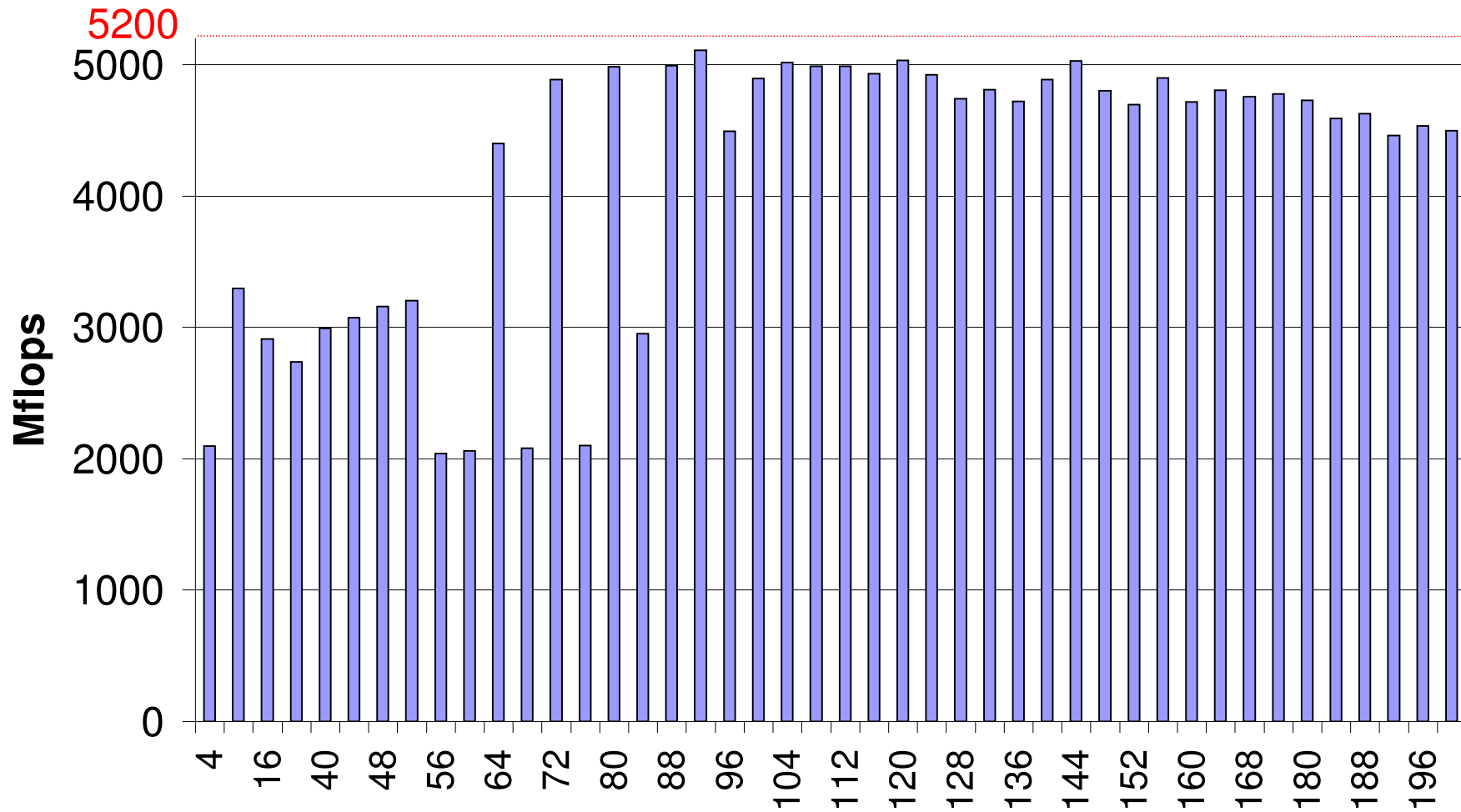
Matrix multiplication performance on small matrices: Alpha 21264

Alpha 21264A (ev67) @ 731 MHz



Matrix multiplication performance on small matrices: Itanium 2

Itanium2 @ 1,3 GHz



Memory hierarchy information

Table 1: Cache sizes

Cache Level	R10000	ALPHA 21264	Itanium2
L1	32 KB	64 KB	16 KB
L2	4 MB	4 MB	256 KB
L3	-	-	3 MB

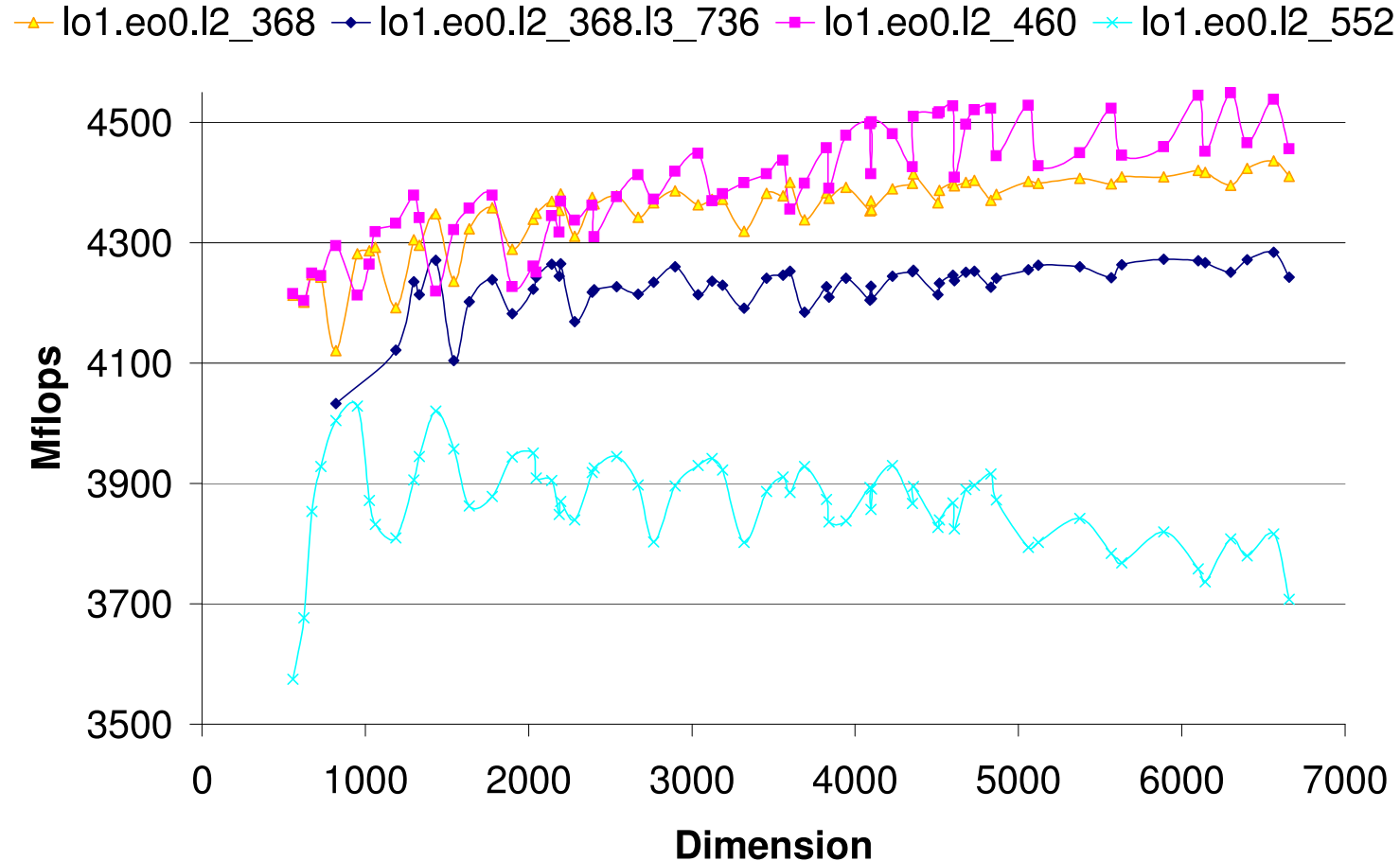
Table 2: Floating-point load latency (minimum) when load hits in cache.

Cache Level	R10000	ALPHA 21264	Itanium2
L1	3	4	1
L2	8-10	13	6
L3	-	-	23

Outline

- Introduction
- Results
- **Details**
 - Inner kernel
 - **Number of pointer levels and dimensions**
 - Orthogonal Blocks
- Conclusions

Number of levels and sizes



Dimension close to the value $\sqrt{C/2}$, where C is the cache size [Lam et al.(1991)Lam, Rothberg, and Wolf].

Outline

- Introduction
- Results
- **Details**
 - Inner kernel
 - Number of pointer levels and dimensions
 - **Orthogonal Blocks**
- Conclusions

Orthogonal blocks

Multilevel Orthogonal Block forms:

- each level is orthogonal to the previous
- directions of the blocks of adjacent levels are different
- exploit the data locality
- [Navarro et al.(1994)Navarro, Juan, and Lang]

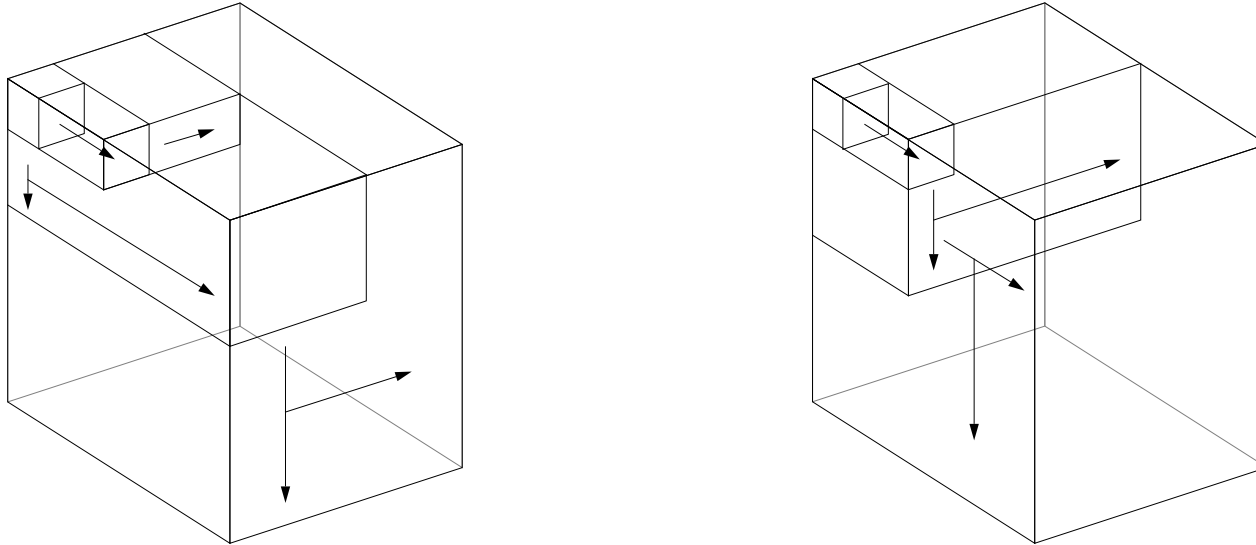
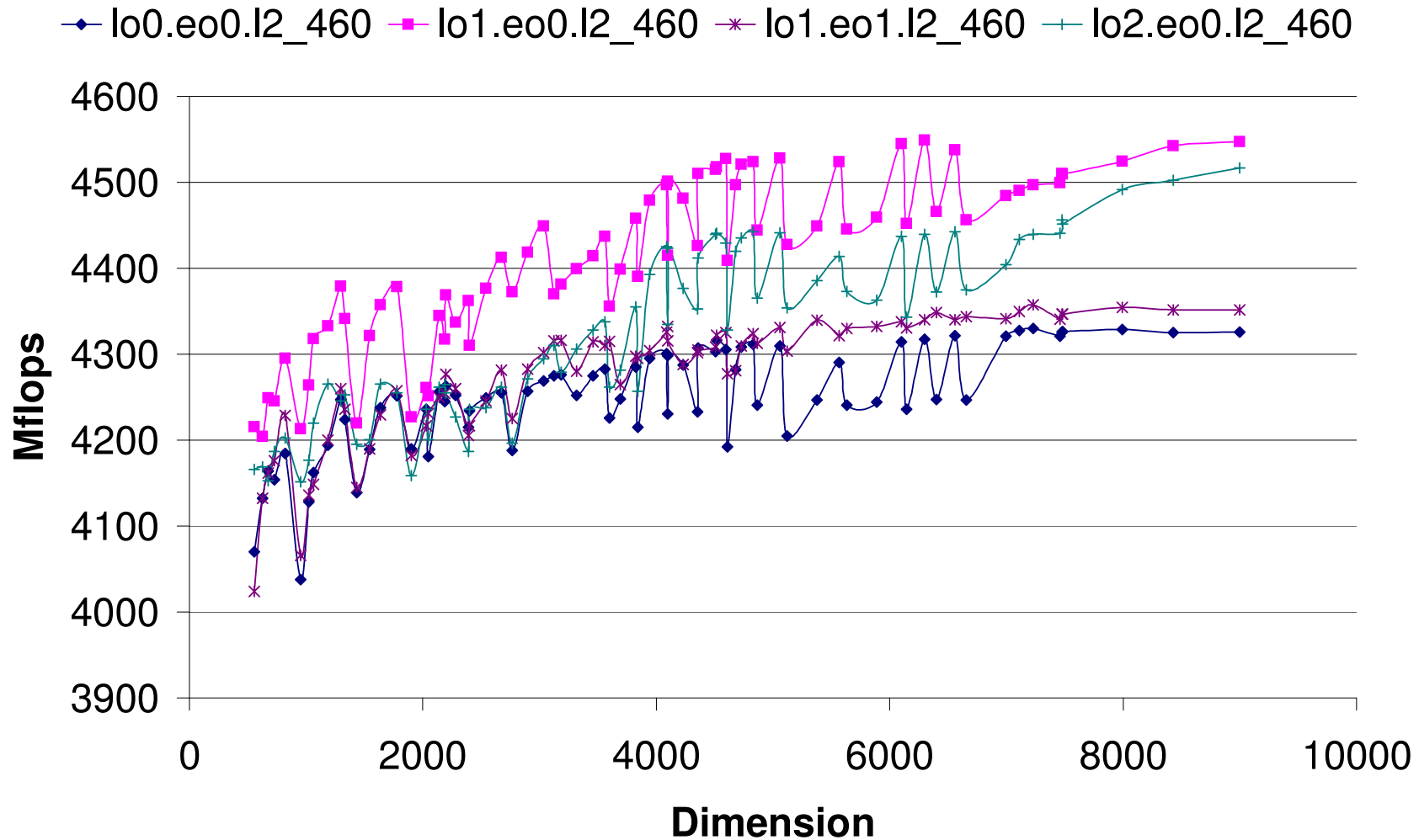
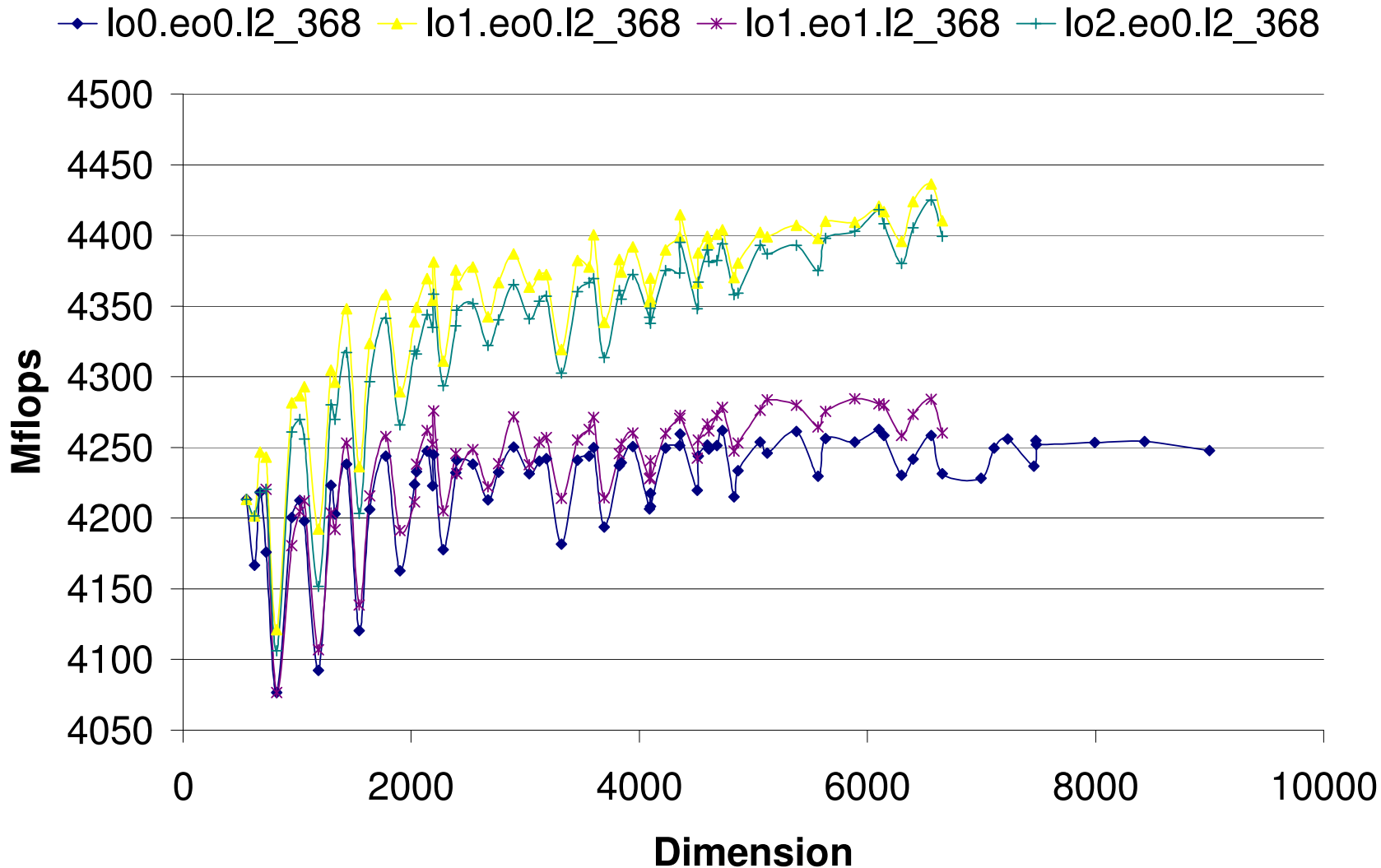


Figure 1: Two examples of Multilevel Orthogonal Block forms

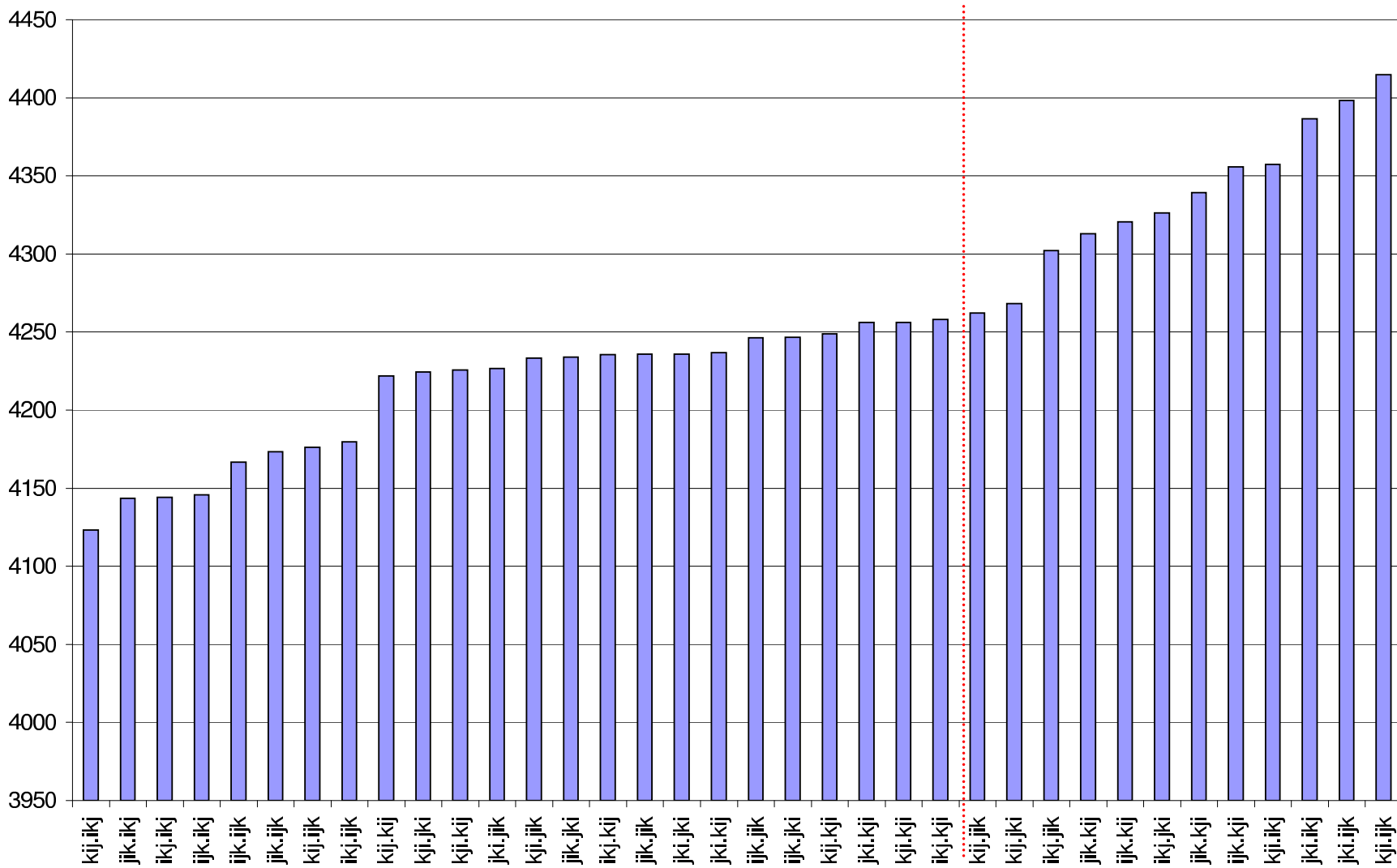
Itanium 2: Dense Matrix Multiplication: Orthogonal vs Non Orthogonal blocks



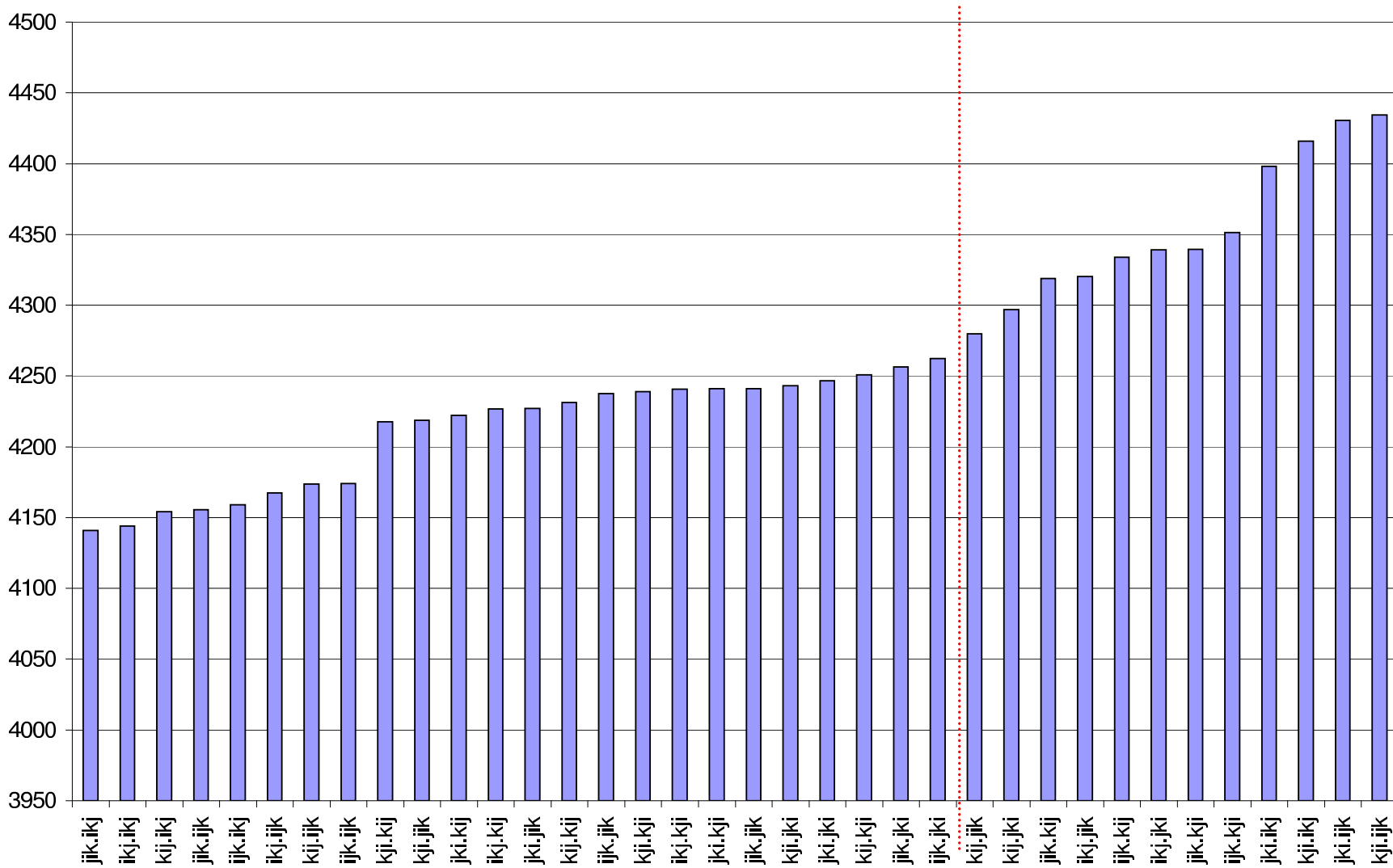
Itanium 2: Dense Matrix Multiplication: Orthogonal vs Non Orthogonal blocks



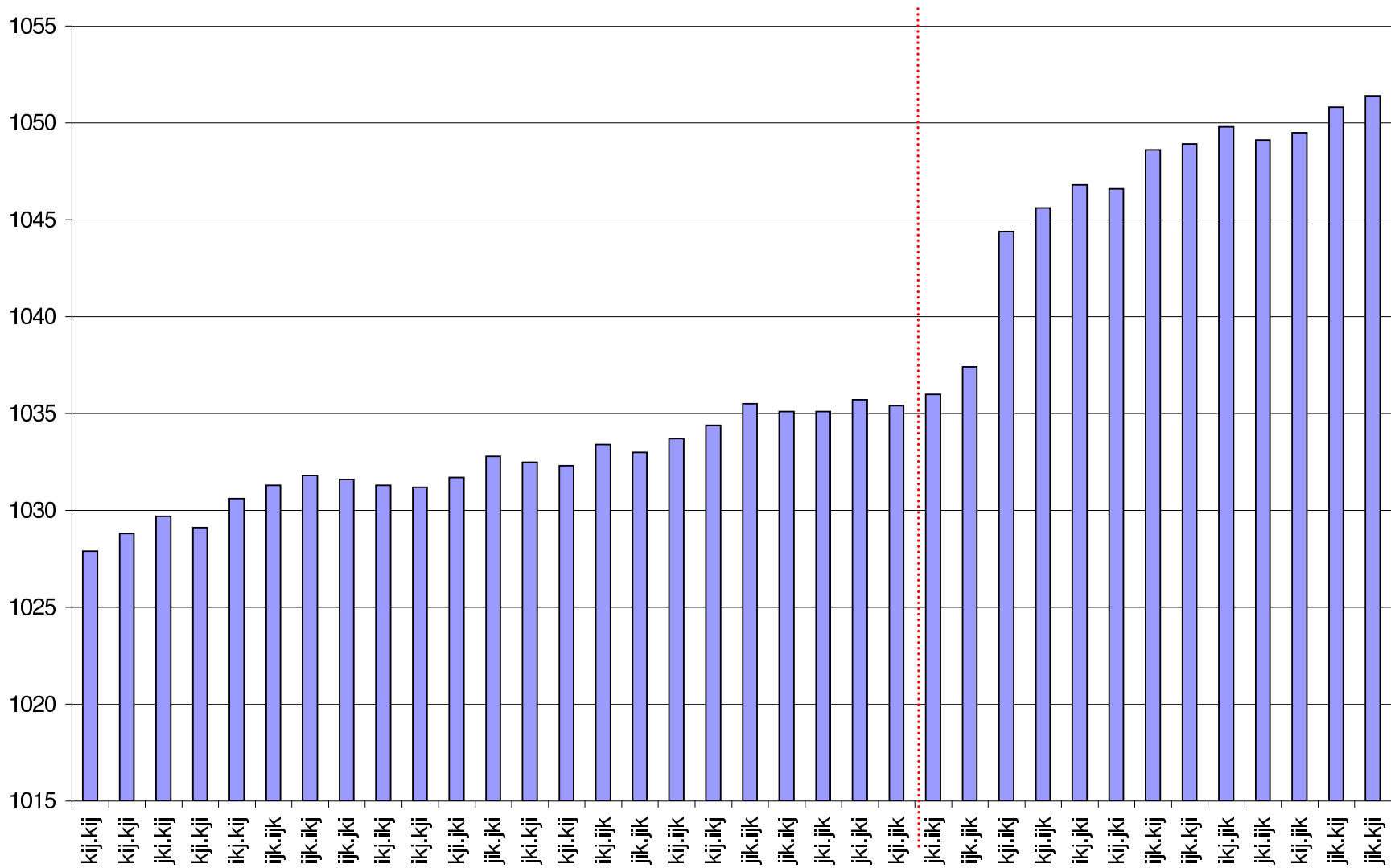
Itanium 2: HM multiplication (Dim 3125)



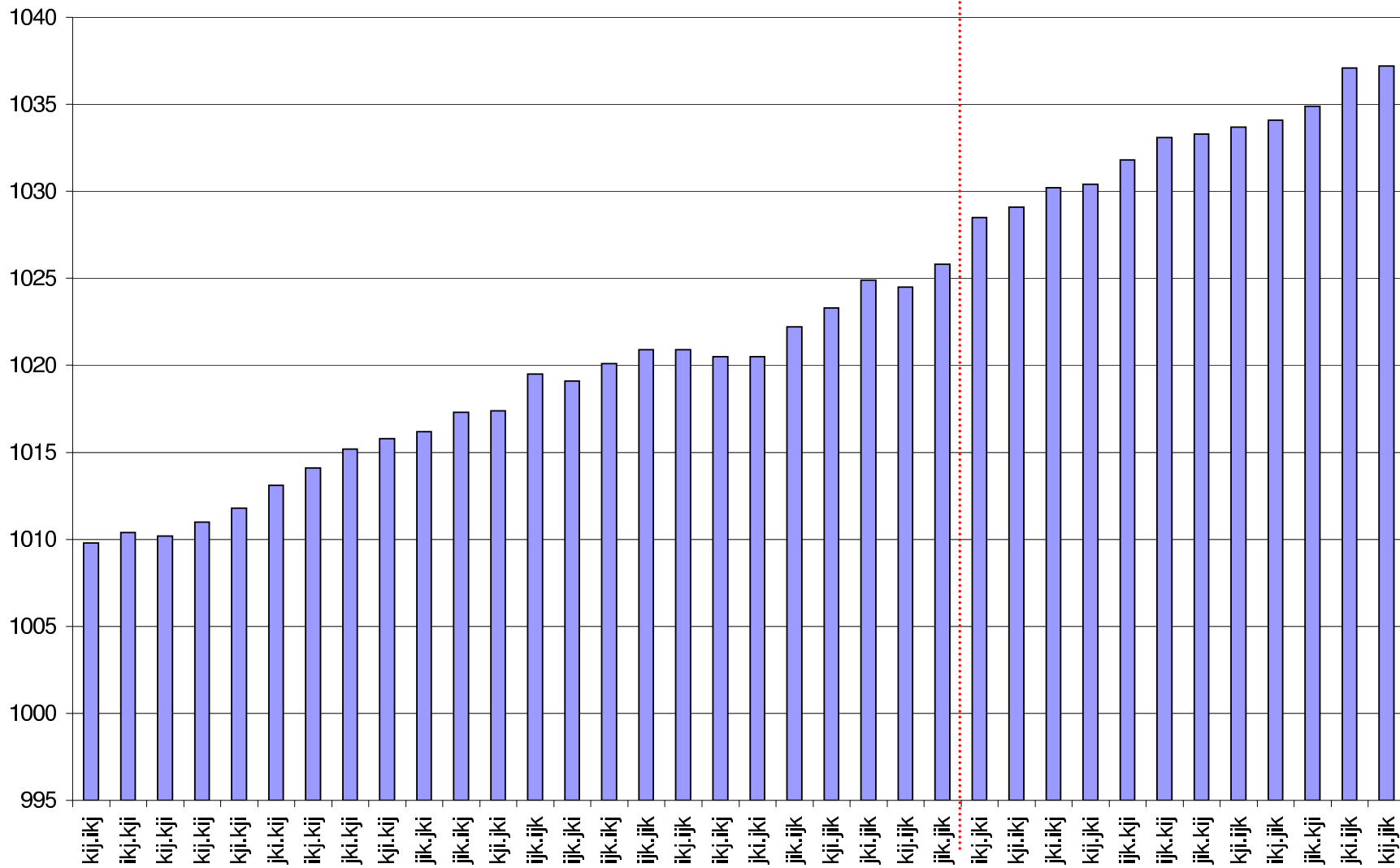
Itanium 2: HM multiplication (Dim 4507)



Alpha 21264: HM multiplication (Dim 3125)



Alpha 21264: HM multiplication (Dim 4507)



Outline

- Introduction
- Results
- Details
 - Inner kernel
 - Number of pointer levels and dimensions
 - Orthogonal Blocks
- **Conclusions**

Conclusions

- Inner kernels can aim at upper level caches if latency is low.
- Known techniques can be applied to Hypermatrix codes to improve performance
- **But** beware of the drawbacks:
 - Some overhead following pointers recursively
 - Load imbalance in parallel implementation
(Talk this afternoon in Track A: 17h.10' $\pm \epsilon$)

References

- [Lam et al.(1991)Lam, Rothberg, and Wolf] Lam, M., Rothberg, E., Wolf, M., 1991. The cache performance and optimizations of blocked algorithms. In: Proceedings of ASPLOS'91. pp. 67–74.
- [Navarro et al.(1994)Navarro, Juan, and Lang] Navarro, J. J., Juan, A., Lang, T., 1994. MOB forms: A class of Multilevel Block Algorithms for dense linear algebra operations. In: Proceedings of the 8th international conference on Supercomputing. ACM Press.