On the behaviour of the MKL library in multicore shared-memory systems

Domingo Giménez

Departamento de Informática
y Sistemas
Universidad de Murcia

Alexey Lastovetsky

School of Computer Science
and Informatics
University College Dublin

Jornadas de Paralelismo, Valencia, Septiembre 2010

Matrix multiplication on platforms composed of multicore

The goal:

- To identify the shape matrix multiplication has in a multicore as a function of the problem size and the number of threads, to decide the number of threads to use to obtain the lowest execution time
- To use this information to develop two-level (OpenMP+BLAS) versions of the multiplication, and select the number of threads in each level
- To use this information to develop three-level (MPI+OpenMP+BLAS) versions, and select the number of processes and threads in each level
- To use this information to develop heterogeneous/distributed three-level (MPI+OpenMP+BLAS) versions, and select the number of processes and its distribution or the data partition, and in each processor the number of threads in each level

Matrix multiplication on platforms composed of multicore

The goal:

- To identify the shape matrix multiplication has in a multicore as a function of the problem size and the number of threads, to decide the number of threads to use to obtain the lowest execution time
- To use this information to develop two-level (OpenMP+BLAS) versions of the multiplication, and select the number of threads in each level
- To use this information to develop three-level (MPI+OpenMP+BLAS) versions, and select the number of processes and threads in each level
- To use this information to develop heterogeneous/distributed three-level (MPI+OpenMP+BLAS) versions, and select the number of processes and its distribution or the data partition, and in each processor the number of threads in each level

Matrix multiplication on platforms composed of multicore

The goal:

- To identify the shape matrix multiplication has in a multicore as a function of the problem size and the number of threads, to decide the number of threads to use to obtain the lowest execution time
- To use this information to develop two-level (OpenMP+BLAS) versions of the multiplication, and select the number of threads in each level
- To use this information to develop three-level (MPI+OpenMP+BLAS) versions, and select the number of processes and threads in each level
- To use this information to develop heterogeneous/distributed three-level (MPI+OpenMP+BLAS) versions, and select the number of processes and its distribution or the data partition, and in each processor the number of threads in each level

Matrix multiplication on platforms composed of multicore

The goal:

- To identify the shape matrix multiplication has in a multicore as a function of the problem size and the number of threads, to decide the number of threads to use to obtain the lowest execution time
- To use this information to develop two-level (OpenMP+BLAS) versions of the multiplication, and select the number of threads in each level
- To use this information to develop three-level (MPI+OpenMP+BLAS) versions, and select the number of processes and threads in each level
- To use this information to develop heterogeneous/distributed three-level (MPI+OpenMP+BLAS) versions, and select the number of processes and its distribution or the data partition, and in each processor the number of threads in each level

Systems, basic components

name	architecture	icc	MKL
rosebud05	4 Itanium dual-core 8 cores	11.1	10.2
rosebud09	1 AMD quad-core 4 cores	11.1	10.2
hipatia8	2 Xeon E5462 quad-core 8 cores	10.1	10.0
hipatia16	4 Xeon X7350 quad-core 16 cores	10.1	10.0
arabi	2 Xeon L5450 quad-core 8 cores	11.1	10.2
ben	HP Integrity Superdome 128 cores	11.1	10.2
bertha	IBM 16 Xeon X7460 hexa-core 96 cores	11.0	11.0

Systems

- Rosebud (Polytechnic Univ. of Valencia):

  cluster with 38 cores

  2 nodes single-processors, 2 nodes dual-processors, 2 nodes with 4 dual-core, 2 nodes with 2 dual-core, 2 nodes with 1 quad-core
- Hipatia (Polytechnic Univ. of Cartagena):

  cluster with 152 cores

  16 nodes with 2 quad-core, 2 nodes with 4 quad-core, 2 nodes with 2 dual-core
- Ben-Arabi (Supercomputing Centre of Murcia):

  Shared-memory + cluster: 944 cores

  Arabi: cluster of 102 nodes with 2 quad-core

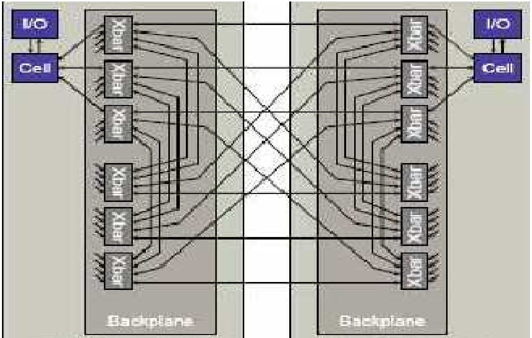
  Ben: HP Superdome, cc-NUMA with 128 cores
- Bertha (INRIA Bordeaux Ouest):

  Shared-memory cc-NUMA: 96 cores

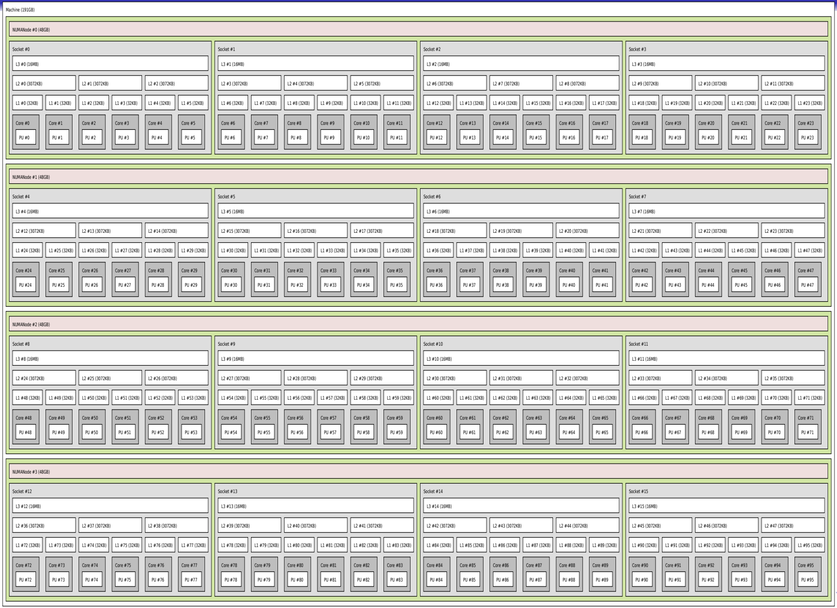
  4 nodes, each node 4 processors, each processor hexa-core

Ben architecture

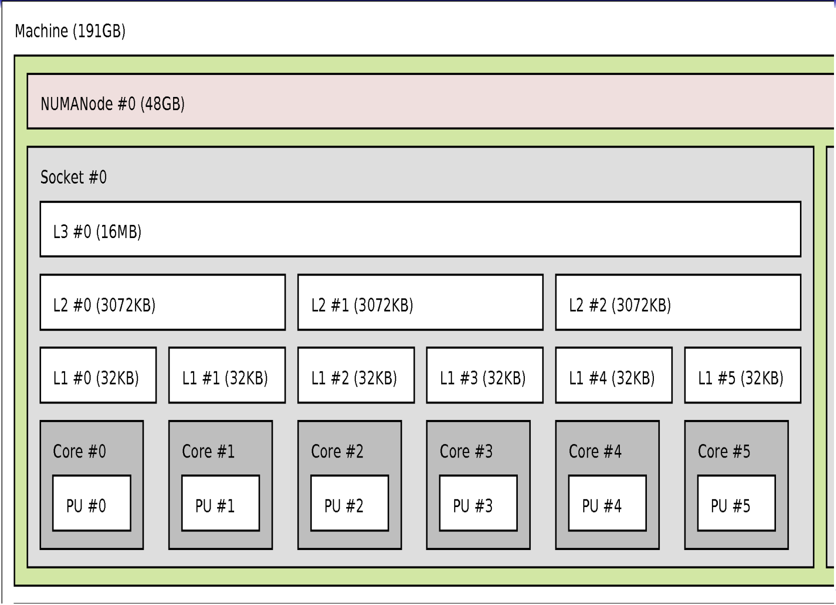
- Hierarchical composition with crossbar interconnection.
- Two basic components: the computers and two backplane crossbars.
- Each computer has 4 dual-core Itanium-2 and a controller to connect the CPUs with the local memory and the crossbar commuters.
- The maximum memory bandwidth in a computer is 17.1 GB/s and with the crossbar commuters 34.5 GB/s.
- The access to the memory is non uniform and the user does not control where threads are assigned.



Bertha architecture



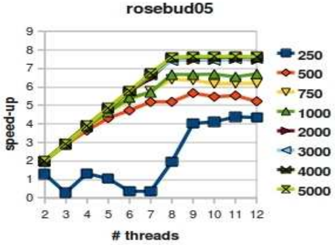
Bertha architecture

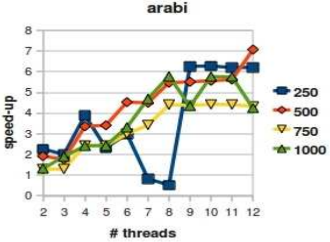


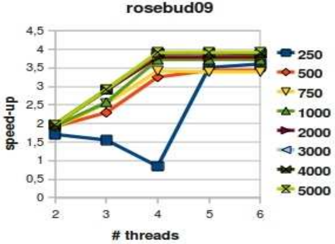
Using MKL

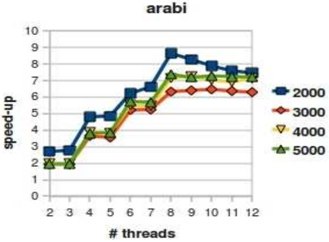
- The library is multithreaded.
- Number of threads estabished with the environment variable `MKL_NUM_THREADS` or in the program with the function `mkl_set_num_threads`.
- Dynamic parallelism is enabled with `MKL_DYNAMIC=true` or `mkl_set_dynamic(1)`. The number of threads to use in `dgemm` is decided by the system, and is less or equal to that established.
- To enforce the utilisation of the number of threads, dynamic parallelism is turned off with `MKL_DYNAMIC=false` or `mkl_set_dynamic(0)`.

MKL, results

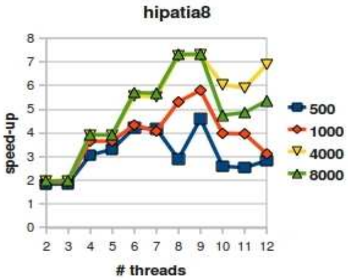

rosebud05
speed-up
0 1 2 3 4 5 6 7 8 9
2 3 4 5 6 7 8 9 10 11 12
threads
250
500
750
1000
2000
3000
4000
5000

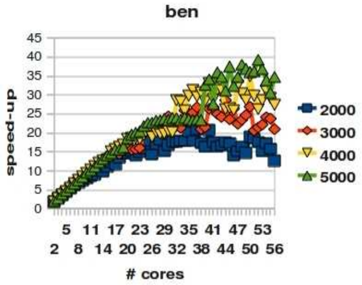

arabi
speed-up
0 1 2 3 4 5 6 7 8
2 3 4 5 6 7 8 9 10 11 12
threads
250
500
750
1000

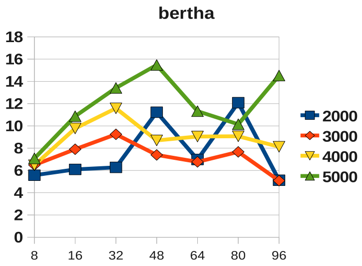

rosebud09
speed-up
0 0,5 1 1,5 2 2,5 3 3,5 4 4,5
2 3 4 5 6
threads
250
500
750
1000
2000
3000
4000
5000


arabi
speed-up
0 1 2 3 4 5 6 7 8 9 10
2 3 4 5 6 7 8 9 10 11 12
threads
2000
3000
4000
5000

MKL, results


hipatia8
speed-up
threads
2 3 4 5 6 7 8 9 10 11 12
0 1 2 3 4 5 6 7 8
500
1000
4000
8000


ben
speed-up
cores
0 5 10 15 20 25 30 35 40 45
5 11 17 23 29 35 41 47 53
2 8 14 20 26 32 38 44 50 56
2000
3000
4000
5000


bertha
0 2 4 6 8 10 12 14 16 18
8 16 32 48 64 80 96
2000
3000
4000
5000

MKL, results

size	Seq.	Max.	Low.
rosebud05			
250	0.0081	0.0042	0.0019 (11)
rosebud09			
250	0.0042	0.0050	0.0012 (5)
hipatia8			
250	0.0035	0.0021	0.0011 (7)
500	0.026	0.0088	0.0056 (9)
750	0.087	0.021	0.017 (9)
arabi			
250	0.0080	0.0015	0.0013 (9)
500	0.034	0.063	0.0049 (12)

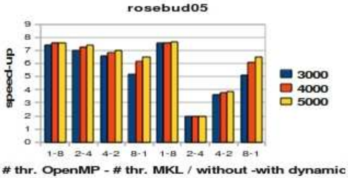
size	Seq.	Max.	Low.
bertha			
1000	0.25	0.50	0.058 (16)
2000	1.8	0.35	0.15 (80)
3000	6.2	1.2	0.67 (32)
4000	15	1.9	1.3 (32)
ben			
250	0.021	0.017	0.0014 (10)
500	0.042	0.033	0.0044 (19)
750	0.14	0.063	0.010 (22)
1000	0.32	0.094	0.019 (27)
2000	2.6	0.39	0.12 (37)
3000	8.6	0.82	0.30 (44)
4000	20	1.4	0.59 (50)
5000	40	2.1	1.0 (48)

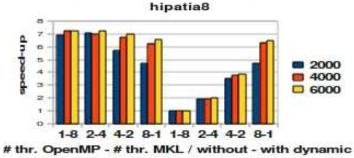
Two-level parallelism

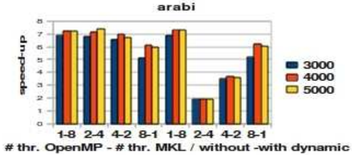
It is possible to use two-level parallelism: OpenMP + MKL.
The rows of a matrix are distributed to a set of OpenMP threads (*nthomp*).
A number of threads is established for MKL (*nthmkl*).
Nested parallelism must be allowed, with `OMP_NESTED=true` or `omp_set_nested(1)`.

```
omp_set_nested(1);
omp_set_num_threads(nthomp);
mkl_set_dynamic(0);
mkl_set_num_threads(nthmkl);
#pragma omp parallel
    obtain size and initial position of the submatrix of A to be
        multiplied
    call dgemm to multiply this submatrix by matrix B
```

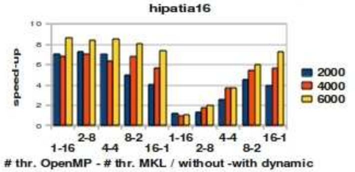
Two-level parallelism, results

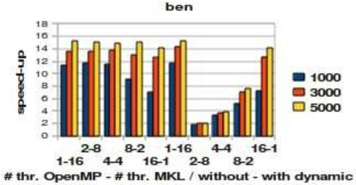




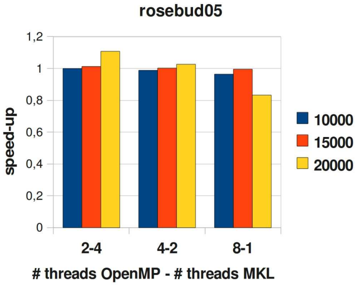


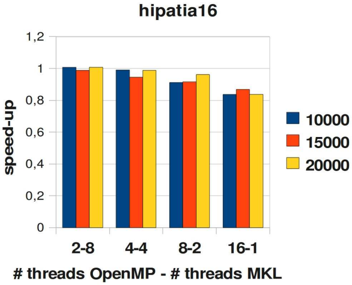
hipatia16

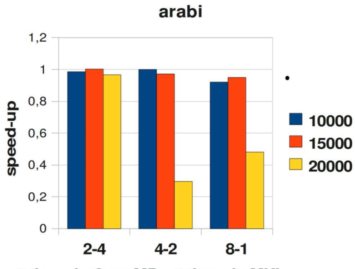


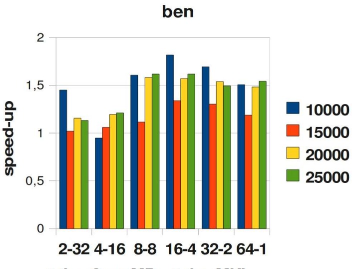


Two-level parallelism, results

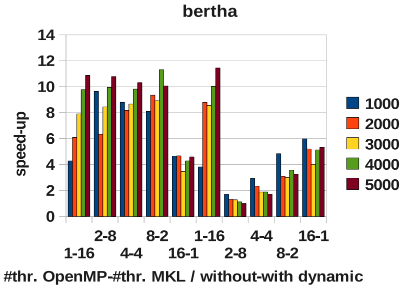


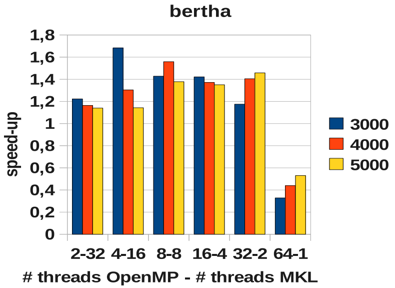






Two-level parallelism, results





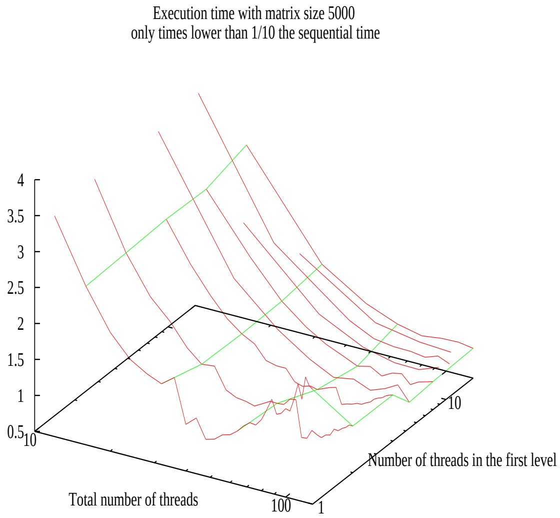
Two-level parallelism, conclusions

- In Hipatia (MKL version 10.0) the nested parallelism seems to disable the dynamic selection of threads.
- In the other systems, with dynamic assignation the number of MKL threads seems to be one when more than one OpenMP threads are running.
- When the number of MKL threads is established in the program bigger speed-ups are obtained.
- Normally the use of only one OpenMP thread is preferable.
- In large systems it is preferable to use a higher number of OpenMP threads: in Ben a speed-up between 1.2 and 1.8 is obtained with 16 OpenMP and 4 MKL threads, in Bertha between 1.4 and 1.6 with 8 and 8 threads.

Two-level parallelism, results

size	ben			bertha		
	MKL	2-levels	Sp.	MKL	2-levels	Sp.
250	0.0014 (10)	0.0014 (1-10)	1.0			
500	0.0044 (19)	0.0043 (4-11)	1.0			
750	0.010 (22)	0.0095 (4-11)	1.1			
1000	0.019 (27)	0.015 (4-10)	1.3	0.058 (16)	0,014 (2-24)	4.2
2000	0.12 (37)	0.072 (4-16)	1.6	0.15 (80)	0.053 (5-16)	2.8
3000	0.30 (44)	0.18 (4-24)	1.7	0.67 (32)	0.51 (16-3)	1.3
4000	0.59 (50)	0.41 (5-16)	1.4	1.3 (32)	0.98 (5-16)	1.3
5000	1.0 (48)	0.76 (6-20)	1.3	1.9 (48)	1.7 (3-32)	1.2
10000	10 (64)	5.0 (32-4)	2.0			
15000	25 (64)	12 (32-4)	2.1			
20000	65 (64)	22 (16-8)	3.0			
25000	130 (64)	44 (16-8)	3.0			

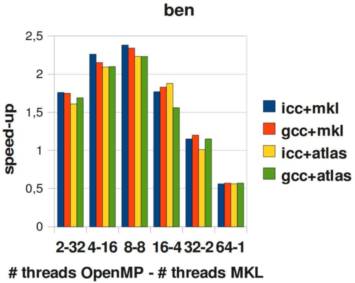
Two-level parallelism, surface shape, in Ben



Two-level parallelism, results

Similar results are obtained with other compilers and libraries.

Ben: gcc 4.4 and ATLAS 3.9.



Matrix multiplication: research lines

- Development of a 2lBLAS prototype,
  and application to scientific problems
- Simple MPI+OpenMP+MKL version
  Experiments in large shared-memory (ben), large clusters (arabi), and heterogeneous (rosebud)
- ScaLAPACK style MPI+OpenMP+MKL version
  Determine number of processors, and OpenMP and MKL threads
  From the model and empirical analysis or with adaptive algorithm
  In heterogeneous platform the number of processes per processor
- HoHe ScaLAPACK style MPI+OpenMP+MKL version
  Determine volume of data for each processors, and OpenMP and MKL threads
  From the model and empirical analysis or with adaptive algorithm
- Distributed style MPI+OpenMP+MKL version

Questions?

... and if somebody has access to large cc-NUMA systems, you could repeat some of the tests (code in `http://www.um.es/pcgum`) and send me (`domingo@um.es`) the results

thanks!