The MUMPS library: work done during the SOLSTICE project

MUMPS team, Lyon-Grenoble, Toulouse, Bordeaux

Sparse Days and ANR SOLSTICE
Final Workshop
June 2010

MUMPS Team since beg. of SOLSTICE (2007)

Permanent members in 2007

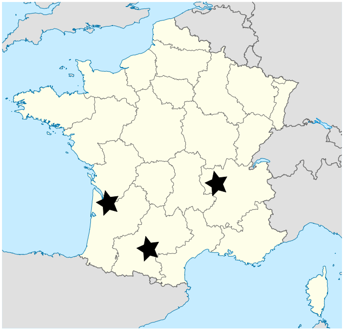
Patrick Amestoy (N7-IRIT, Toulouse)

Jean-Yves L'Excellent (INRIA-LIP, Lyon)

Abdou Guermouche (LABRI, Bordeaux)

Bora Uçar (CNRS-LIP, Lyon)

Alfredo Buttari (CNRS-IRIT, Toulouse)



MUMPS Team since beg. of SOLSTICE (2007)

Permanent members in 2010

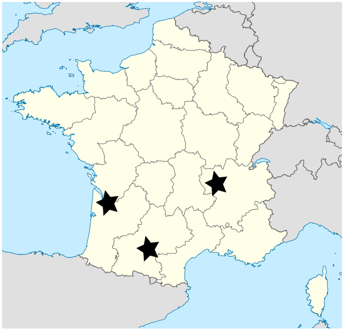
Patrick Amestoy (N7-IRIT, Toulouse)

Jean-Yves L'Excellent (INRIA-LIP, Lyon)

Abdou Guermouche (LABRI, Bordeaux)

Bora Uçar (CNRS-LIP, Lyon)

Alfredo Buttari (CNRS-IRIT, Toulouse)



MUMPS Team since beg. of SOLSTICE (2007)

- Post-docs funded by the ANR SOLSTICE project :
  Indranil Chowdhury (May 2009–March 2010)
  Alfredo Buttari (Jan. 2008-Oct. 2008)
  Bora Uçar (Jan. 2007-Dec. 2008)

- PhD. Students Emmanuel Agullo (ENS Lyon, 2005-2008)
  Mila Slavova(CERFACS, Toulouse, 2005-2009)
  François-Henry Rouet (INPT-IRIT, Toulouse, 2009-) *new!*

- Master Student Clément Weisbecker (INPT-IRIT, Toulouse) *new!*

- Engineers Aurélia Fèvre (INRIA, 2005-2007)
  Philippe Combes (CNRS, Dec. 2007-Dec. 2008)
  Maurice Brémond (INRIA, Oct. 2009-Oct. 2012) *new!*
  Guillaume Joslin (INRIA, Oct.2009-Oct. 2011) *new!*

What is MUMPS

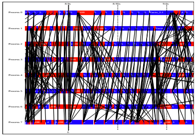
MUMPS (**MU**ltifrontal **M**assively **P**arallel sparse direct **S**olver) solves sparse systems of linear equations $Ax = b$
`http://graal.ens-lyon.fr/MUMPS` and `http://mumps.enseeiht.fr`

- Initially funded by European project PARASOL (1996-1999)
- Co-developed in Lyon-Toulouse-Bordeaux
- Latest release : MUMPS 4.9.2, Nov. 2009

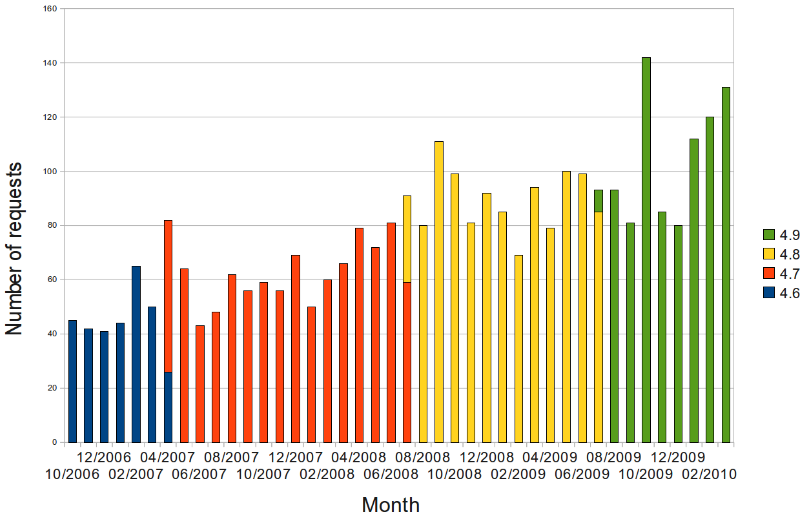
  $\approx$ 250 000 lines of C and Fortran code
- 1000+ downloads per year from our website

Main originalities

- Wide range of numerical features
- Numerical stability based on threshold partial pivoting
- Mainly MPI-based, with a dynamic and asynchronous approach to parallelism



Download requests forms filled on the MUMPS website



ANR SOLSTICE project ANR-06-CIS6-010

SOLSTICE project http://solstice.gforge.inria.fr

- Supported by ANR (project number ANR-06-CIS6-010)
- Academics : CERFACS, INRIA (Lyon and Bordeaux), INPT-IRIT
  Industrials : CEA-CESTA, EADS CCR, EDF, CNRS-CNRM-LA

SOLSTICE tasks

- T1 : Algorithmic tasks
  - T1.1 : Rank-revealing and null space
  - T1.2 : Out-of-core
  - T1.3 : Parallel graph partitioners – PT-SCOTCH
  - T1.4 : Parallel analysis and preprocessings
  - T1.5 : Hybrid direct/iterative solvers
- T2 : Applications related to industrial activities (Meteo, Electromagnetism, Structural mechanics)
- T3 : TLSE platform (http://www.gridtlse.org/)

ANR SOLSTICE project ANR-06-CIS6-010

SOLSTICE project http://solstice.gforge.inria.fr

- Supported by ANR (project number ANR-06-CIS6-010)
- Academics : CERFACS, INRIA (Lyon and Bordeaux), INPT-IRIT
  Industrials : CEA-CESTA, EADS CCR, EDF, CNRS-CNRM-LA

SOLSTICE tasks involving MUMPS team

- T1 : Algorithmic tasks
  - T1.1 : Rank-revealing and null space → collab. with X. Vasseur (CERFACS) and S. Gratton (INPT-IRIT)
  - T1.2 : Out-of-core
  - T1.3 : Parallel graph partitioners – PT-SCOTCH
  - T1.4 : Parallel analysis and preprocessings
    → see talk by Alfredo Buttari and Bora Uçar
  - T1.5 : Hybrid direct/iterative solvers
- T2 : Applications related to industrial activities (Meteo, Electromagnetism, Structural mechanics)
- T3 : TLSE platform (http://www.gridtlse.org/) → see talk by TLSE team

Contents of the talk

- Illustration of out-of-core work (T1.2)
- Illustration of additionnal work (concerning all tasks including industrial requirements)
- Multicores : some recent work to mix OpenMP and MPI
- Concluding remarks and future work

Out-of-core related work

Out-of-core storage : 2 PhD completed

- Emmanuel AGULLO (ENS Lyon, 2005-2008) *On the Out-of-core Factorization of Large Sparse Matrices*
- Mila Slavova (CERFACS, Toulouse, 2005-2009) *Parallel triangular solution in the out-of-core multifrontal approach*
- Tree traversals, low-level I/O mechanisms, task scheduling, ...

- Out-of-core factorization using a panel-oriented scheme
  (Needed to validate research ; numerical pivoting → more complex)

		I/O granularity for Factors	
Matrix	#procs	Written by fronts	Written by panels
AUDIKW_1	1	1067.1	12.8
AUDIKW_1	32	155.5	12.8
CONV3D64	1	3341.5	40.2
CONV3D64	32	757.6	40.2

Size of I/O Buffers (MB) with asynchronous I/O's

Out-of-core related work (cont')

Computing entries of the inverse of a sparse matrix

- PhD François-Henry Rouet, 2009-
  (continuation/extension of PhD work of M. Slavova)
  - Exploit sparsity of multiple right-hand sides (RHS) during the solution
  - Combinatorial problem to decide how to partition the right-hand-sides.
  - Illustration on an application from astrophysics

No exploit sparsity of RHS	43 396 sec
Natural ordering	721 sec
PostOrdering	199 sec

time to compute **ALL** diagonal of A^{-1}, OOC, N=148k

  - On going work : use of hypergraph partitionning, in-core and flop reduction, parallelism.

No exploit sparsity of RHS	4708 sec
Exploit sparsity	188 sec

time to compute **ALL** the diagonal of A^{-1}, in core, N=148k

Illustration of additionnal related work

- 64-bit integers to address large internal arrays (requested by users but also needed to experiment out-of-core and parallel analysis on large challenging problems)
- Redesign parts of the mapping algorithm ("Epicure" matrix from EDF)

	Factors Min/Max	InCore Memory (in MB) Avg	Max
Before	0.06	1,753	2,883
After	0.70	1,634	2,019

	Factor. time (seconds)				
Nprocs MPI	2	4	8	16	32
Before	337	229	132	86	52
After	316	163	103	53	33

Multicores : some recent work to mix OpenMP and MPI

Work started with Indranil Chowdhury, Sosltice postdoc (May 2009–March 2010)

Main goal : understand the limits of MPI + threaded BLAS + OpenMP

Work based on the use of TAU and Intel Vtune profiles :

- Insert OpenMP directives (assembly, stack, initialization,. . .)
- Experiment with various matrices on various architectures

Difficulties :

- Understand speed-down problems (use of `OMP IF` directives)
- Side effects of threaded BLAS libraries (MKL ok)
- Dependence on OpenMP implementations
- Threaded BLAS within OpenMP parallel regions

Hybrid MPI + multithreaded version of MUMPS

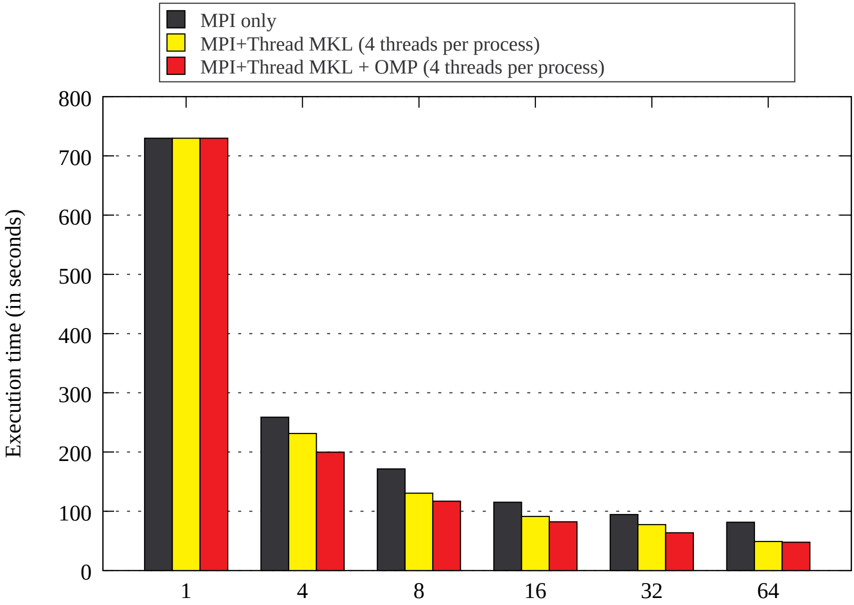
Initial results :

- best current compromise : use 4 threads per MPI process

  typical speed-up of 6 on 8 cores
- unsymmetric and Cholesky versions more efficient than LDL^t

Recent illustrative results :

- 96-core machine, INRIA Bordeaux Sud-Ouest
- Runs by A. Guermouche on a medium-size problem (ULTRASOUND80)
  - Order : 531 441, non-zeroes : 330 $\times 10^6$
  - Factors : 981×10^9 entries, 3.9×10^{12} flops

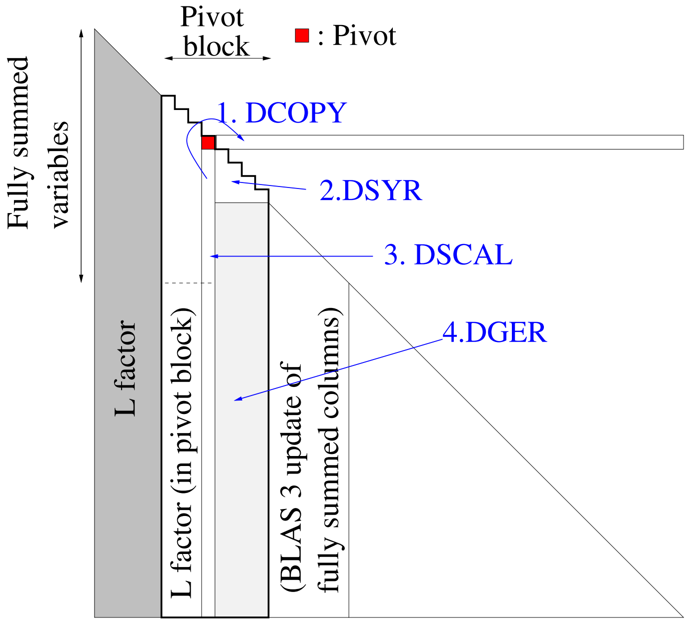

MPI only
MPI+Thread MKL (4 threads per process)
MPI+Thread MKL + OMP (4 threads per process)
Execution time (in seconds)
800
700
600
500
400
300
200
100
0
1
4
8
16
32
64

Remark on memory usage

Increasing the number of threads per MPI process is also critical for memory :

1 MPI 8 threads $\rightarrow$ 8.97 GB
2 MPI 4 threads $\rightarrow$ 11.0 GB
4 MPI 2 threads $\rightarrow$ 11.3 GB
8 MPI 1 thread $\rightarrow$ 13.3 GB

Locality issues with LDL^t factorization (row-major stor.)



Rewriting factorization of pivot block

Old algorithm (threaded BLAS)

1. DCOPY (column to row in upper part)
2. DSCAL (column)
3. DSYR (triangular block)
4. DGER (rectangular block)

New algorithm

Hand-written
No BLAS calls
OpenMP based
Locality better exploited

Results on matrix "Haltere" from CEA-CESTA, CINES SGI

# threads	1	2	4	8
Before (4.9.2)	16.7 (115)	16.7 (75)	19.9 (64)	29.0 (66)
Intermediate	15.8 (110.7)	10.1 (65.4)	8.5 (47.5)	7.6 (39.6)
After	15.7 (110.4)	9.8 (64.6)	6.0 (44.1)	4.7 (34.9)

Legend : Time factorization of pivot block (time for whole factorization)

Times in seconds Intermediate=without "OMP IF"

Rewriting factorization of pivot block

Old algorithm (threaded BLAS)

1. DCOPY (column to row in upper part)
2. DSCAL (column)
3. DSYR (triangular block)
4. DGER (rectangular block)

New algorithm

Hand-written
No BLAS calls
OpenMP based
Locality better exploited

Results on matrix "Haltere" from CEA-CESTA, CINES SGI

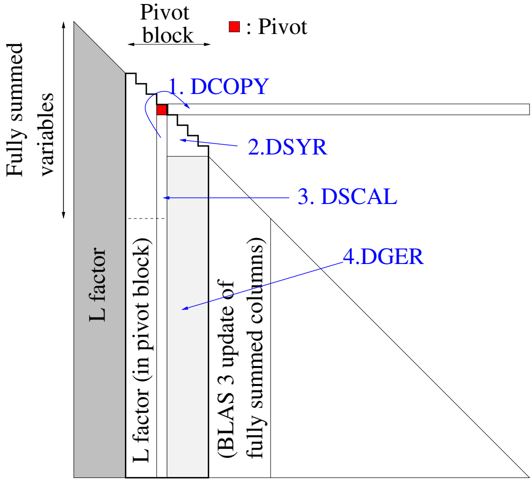
# threads	1	2	4	8
Before (4.9.2)	16.7 (115)	16.7 (75)	19.9 (64)	29.0 (66)
Intermediate	15.8 (110.7)	10.1 (65.4)	8.5 (47.5)	7.6 (39.6)
After	15.7 (110.4)	9.8 (64.6)	6.0 (44.1)	4.7 (34.9)

Legend : Time factorization of pivot block (time for whole factorization)

Times in seconds Intermediate=without "OMP IF"

Further improvements to LDL^t version

During update of pivot block with new algorithm, check for stability of pivot candidate from next column (while in cache !)



Further improvements to LDL^t version

During update of pivot block with new algorithm, check for stability of pivot candidate from next column (while in cache !)

# threads	algorithm	Pivot search	Pivot block update
1	old	2.77	15.6
	new	0.91	15.3
2	old	1.95	9.8
	new	0.72	9.7
4	old	1.56	6.0
	new	0.64	6.0
8	old	1.58	4.7
	new	0.64	4.8

Time reduction on pivot search without affecting pivot block update

LDL^t : Summary

- Huge gains obtained by studying more closely locality aspects :
  On 8 OpenMP threads : 66 $\rightarrow$ 34 seconds (1 thread : 111 seconds)

  (4 threads with 2 MPI processes would be better – see previous results)

On-going work

- Similar work needed for 2x2 pivots updates and distributed fronts
- Other kernels to be studied in more details
- For example on this matrix : 7 seconds out of 34 (8 threads) and the same 7 seconds out of 111 (1 thread) spent in assembly operations
- Clearly much scope for further improvements

Concluding remarks

Main new functionalities since beginning of SOLSTICE project

- T1.1 : null pivots and null space basis
- T1.2 : out-of-core factorization and solve phases
- T1.4 : parallel scalings and parallel symbolic factorization, use of PT-scotch and Parmetis
- 64-bit integers to address large arrays
- performance and memory improvements on SOLSTICE applications
- determinant (factor-less factorization)

Future work

- Improve multithreaded parallelism :
  - locality, affinity : continue on-going work
  - threaded tree parallelism at bottom of tree
  - serial BLAS and more OpenMP ? GPU ?
- Improve MPI parallelism, memory scalability and quality of memory estimates in a parallel context (PhD thesis of F.H. Rouet)
- Performance of solution phase (MPI/OpenMP/sparse right-hand sides)
- Reduce memory consumption and improve performance