IMA SUMMER SCHOOL
IN
KENTUCKY

Multifrontal methods ... Distributed memory

Iain S Duff
Rutherford Appleton Laboratory
CERFACS
Strathclyde University

`http://www.cse.clrc.ac.uk/Person/I.S.Duff`

PARASOL - Integrated Environment for Parallel Sparse Matrix Solvers

Partners

- European research organizations: CERFACS, GMD-SCAI, ONERA, RAL, Para//abb
- Industrial code developers: INPRO, MacNeal-Schwendler, Det Norske Veritas, Polyflow, Apex Technologies
- European HPC software companies: GENIAS, PALLAS

Project ran from January 1996 until June 1999

PARASOL

Objectives

- Develop parallel sparse matrix solvers
- Specify an open library interface (PARASOL Library)
- Implement portable prototypes on different HPC systems
- Integrate the solvers and additional tools into TRAPPER
- Evaluate the PARASOL library and utilities

Final Library in public domain

PARASOL

Main target machines were:

- IBM SP-2
- SGI Origin 2000

with tests also on

- CRAY T3D/E
- Fujitsu VPP

Codes were written in Fortran 90 using MPI

The direct solver in PARASOL is **MUMPS**:

M
Ultifrontal
Massively
Parallel
Solver

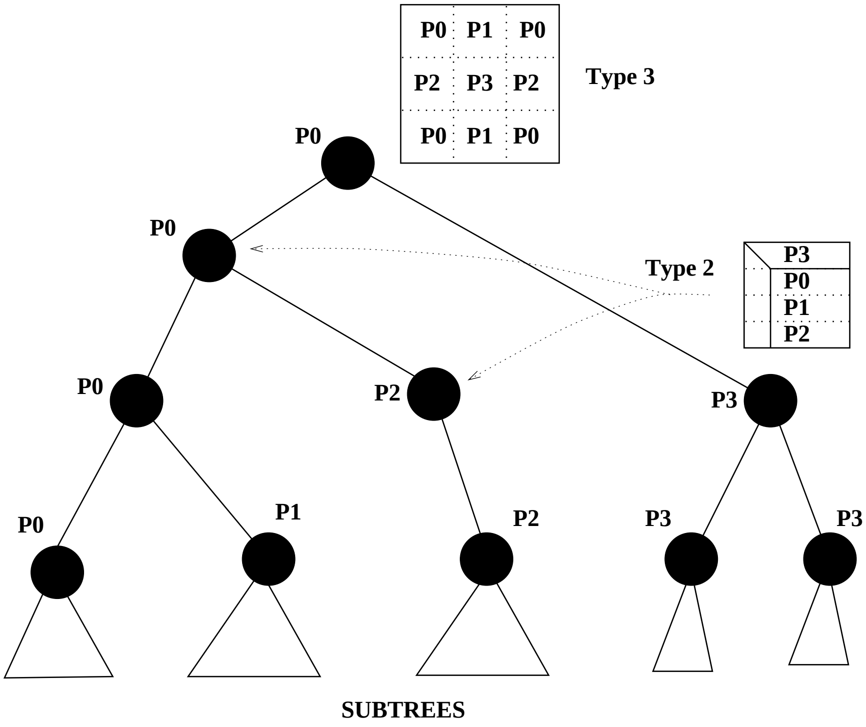
Patrick R. Amestoy (ENSEEIHT)
Iain S. Duff (CERFACS/RAL)
Jean-Yves L'Excellent (INRIA/Lyon)
Jacko Koster (Bergen)
Petr Plecháč (Delaware)
Miroslav Tůma (Prague)

mumps@cerfacs.fr

`http://www.enseeiht.fr/apo/MUMPS`

Distributed memory parallelism

- Type 1: Parallelism of the tree
- Type 2: 1D partitioning of frontal matrices with distributed assembly process
- Type 3: 2D partitioning of root nodes (ScaLAPACK)



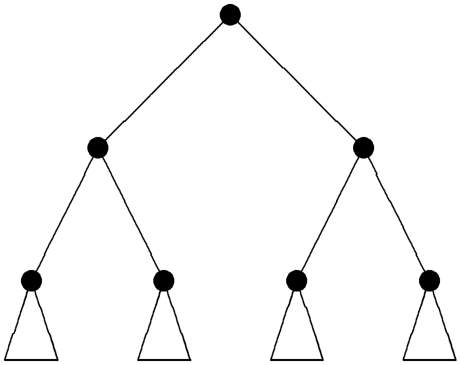
Performance of the unsymmetric and symmetric versions of the code on test problem QUER on an IBM SP2

Working processors	Time for factorization	
	unsymmetric	symmetric
1 (CPU)	64.9	41.1
1 (elapsed)	299.2	150.4
2 (elapsed)	109.1	21.0
4 (elapsed)	19.1	12.9
8 (elapsed)	15.2	9.3
16 (elapsed)	11.5	6.4
24 (elapsed)	10.1	6.6
32 (elapsed)	10.5	5.8

Results for the symmetric version of the code on CRANKSEG2.

Machine	Working processors	Time for factorization
SP2	16	42
	24	31
	32	27
Origin	1	217
	2	112
	4	66
	8	46
	16	29

Incomplete Nested Dissection



Order subtrees using minimum degree

Effect of ordering on Problem BMWCRA_1
Order 148,770. Entries 5,396,386

Analysis

	Entries in factors	Operations
AMD	**113,788,395**	**1.28 x** 10^{11}
ND/MMD	**85,301,445**	**6.72 x** 10^{10}

Factorization [Origin 2000 ... Bergen]
Time in seconds (speedups in parentheses)

	Factor		Solve	
No. procs	**AMD**	**ND+MMD**	**AMD**	**ND+MMD**
1	**687**	**307**	**12.0**	**10.1**
2	**408**	**178**	**7.5**	**5.4**
4	**236**	**82**	**6.7**	**4.2**
8	**143(4.8)**	**58(5.3)**	**4.2(2.9)**	**2.6(3.9)**
16	**112**	**36**	**2.9**	**1.9**

Time (in seconds) for factorization using `MUMPS` Version 4 on some PARASOL test problems on an Origin 2000.

Matrix	Flops	Number of processors					
	($\times 10^9$)	1	2	4	8	16	32
LDOOR	74.5	416	228	121	68	39	31 (2.4 GFlops)
BMWCRA_1	61.0	307	178	82	58	36	27 (2.3 GFlops)
BMW3_2	28.6	151	96	53	33	18	15 (1.9 GFlops)
INLINE_1	143.2	757	406	225	127	76	55 (2.6 GFlops)
SHIP_003.RSE	73.0	392	237	124	108	51	43 (1.7 GFlops)
SHIPSEC5.RSE	51.7	281	181	103	62	37	29 (1.8 GFlops)

AUDI-CRANKSHAFT

ORDER: 943,695
ENTRIES: 39,297,771

ANALYSIS

Entries in factors : 1,435,757,859

[11.2 GBytes]

Operations required : 5.9×10^{12}

FACTORIZATION [SGI ORIGIN at Bergen]

1 Processor	32000 secs 16 GBytes
2 Processor	22000 secs 20 GBytes

Distributed Memory Codes

Code	Technique	Scope	Availability
CAPSS	Multifrontal	SPD	`www.netlib.org/scalapack`
MUMPS	Multifrontal	SYM/UNS	`www.enseeiht.fr/apo/MUMPS`
PaStiX	Fan-in	SPD	see caption[§]
PSPASES	Multifrontal	SPD	`www.cs.umn.edu/~mjoshi/pspases`
SPOOLES	Fan-in	SYM/UNS	`www.netlib.org/linalg/spooles`
SuperLU	Fan-out	UNS	`www.nersc.gov/~xiaoye/SuperLU`
S+	Fan-out[†]	UNS	`www.cs.ucsb.edu/research/S+`
WSMP[‡]	Multifrontal	SYM	IBM product

[§] `dept-info.labri.u-bordeaux.fr/~ramet/pastix`
[†] Uses QR storage to statically accommodate any LU fill-in
[‡] Only object code for IBM is available. No numerical pivoting performed.

Shared Memory Codes

Code	Technique	Scope	Availability
GSPAR	Interpretative	UNS	Grund
MA41	Multifrontal	UNS	`www.cse.clrc.ac.uk/Activity/HSL`
MA49	Multifrontal QR	RECT	`www.cse.clrc.ac.uk/Activity/HSL`
PanelLLT	Left-looking	SPD	Ng
PARDISO	Left-right looking	UNS	Schenk
PSLDLT[†]	Left-looking	SPD	SGI product
PSLDU[†]	Left-looking	UNS	SGI product
SPOOLES	Fan-in	SYM/UNS	`www.netlib.org/linalg/spooles`
SuperLU	Left-looking	UNS	`www.nersc.gov/~xiaoye/SuperLU`
WSMP[‡]	Multifrontal	SYM/UNS	IBM product

[†] Only object code for SGI is available
[‡] Only object code for IBM is available

Comparison of sparse direct approaches

- Frontal codes
  - Require a well ordered matrix
  - Good on vector machines
  - Good for one-off solution
  - Can easily be run with little main memory
- Multifrontal codes
  - Good for reducing number of entries in factors
  - Efficient if significant fill-in
  - Natural way to parallelize method
- Markowitz/threshold
  - Good for very sparse systems
  - Good when structure very irregular
  - Good at keeping storage requirements low
  - Often good for repeated solutions
  - Not easy to parallelize

SUMMARY

- There is no one best buy
- The best code in any situation will depend on
  - The solution environment
  - The computing platform
  - The structure of the matrix

In short, it is

HORSES FOR COURSES