Consequences of modern hardware design for numerical simulations and their realization in FEAST

Ch. Becker, S. Kilian, S. Turek

Institut für Angewandte Mathematik,
Universität Heidelberg

email: featflow@featflow.de

WWW: http://www.featflow.de

Motivation

- CPU peak performance increases with every processor generation (in 10 years one PC faster than a complete Cray today)
- memory size grows also but not at a similar rate

Ideal situation:

- Example I (3D, Poisson problem)
  - $100 \times 100 \times 100$ grid points $\longrightarrow$ problem size $N = 10^6$
  - Complexity of GE: $N^{7/3} \approx 10^{14}$ FLOP
    100 sec on a 1 TFLOP/s computer
  - Complexity of (opt.) multigrid: $1000N \approx 10^9$ FLOP
    100 sec on a 10 MFLOP/s computer
- Example II (3D, Poisson problem)
  - $1000 \times 1000 \times 1000$ grid points $\longrightarrow$ problem size $N = 10^9$
  - Complexity of GE: $N^{7/3} \approx 10^{21}$ FLOP
    10^6 sec on a 1 PFLOP/s computer
  - Complexity of (opt.) multigrid: $1000N \approx 10^{12}$ FLOP
    1000 sec on a 1 GFLOP/s computer

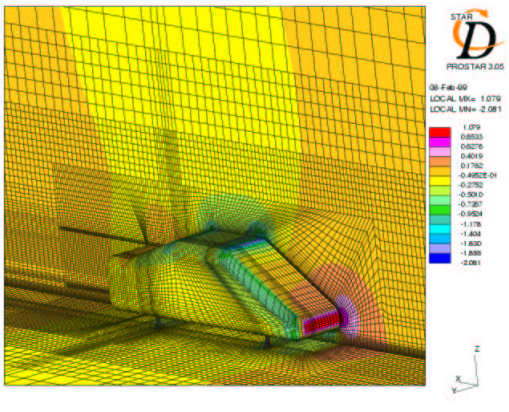
Motivation

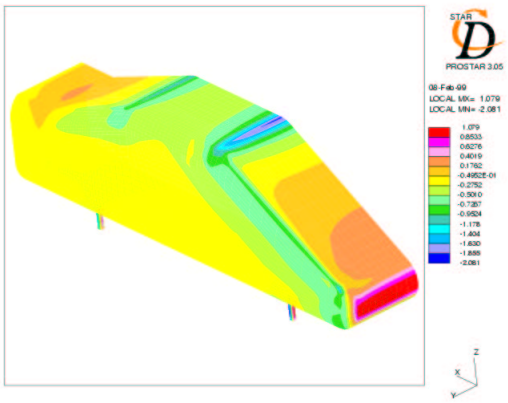
'Optimal' multigrid $\sim$ 1 sec/subproblem/processor???

But typical situation in high performance computing today:

Example: STAR-CD ($k-\epsilon$), 500 000 cells (by Daimer Chrysler), SGI Origin2000 (6 processors), 6.5 CPU h

Quantity	**Experiment**	**Simulation**	**Difference**
Drag ('c_w')	0.165	0.317	**92 %**
Lift ('c_a')	-0.083	-0.127	**53 %**





- Are recent software able to benefit from the performance race?
- Are special strategies necessary to exploit the performance?

Sparse Matrix Vector Multiplication

Standard sparse matrix vector algorithm:

```
        DO 10 IROW=1,N
        DO 10 ICOL=KLD(IROW),KLD(IROW+1)-1
10      Y(IROW)=Y(IROW)+A(ICOL)*X(KCOL(ICOL))
```

Performance rates of the FEATFLOW code with different numbering schemes (Cuthill–McKee, TwoLevel, Stochastic) for matrix vector multiplication:

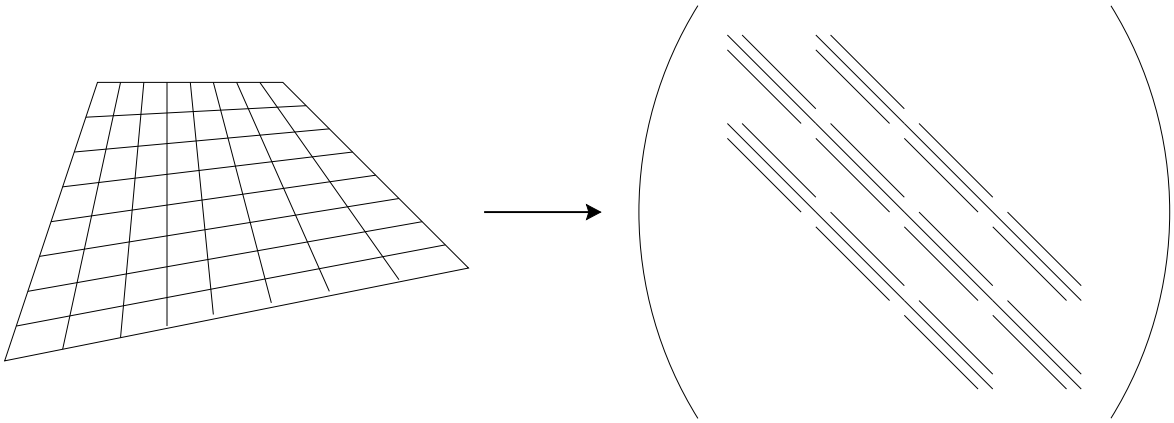
Computer	#Unknowns	**CM**	**TL**	**STO**
SUN E450 ($\sim$ 250 MFLOP/s)	13,688	22	20	**19**
	54,256	17	15	**13**
	216,032	16	14	**6**
	862,144	16	15	**4**

- sparse techniques basis for most of the recent software packages
- different numbering schemes can lead to identical numerical results and work (w.r.t. arith.ops and data accesses) but to huge differences in CPU time
- sparse techniques are 'slow' and depend on problem size and kind of data access

Sparse Banded Techniques

Question: How to exploit more performance?

Line- or rowwise numbering:

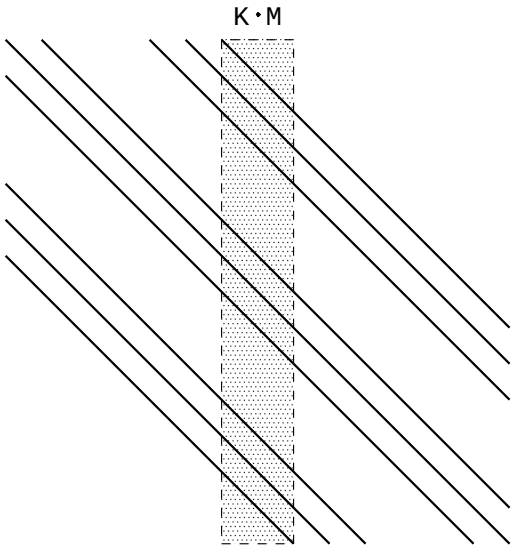


Sparse *banded* matrix vector multiplication:

- FD discretization leads to band structure on generalized tensor product meshes
- storing of matrix elements in diagonals
- matrix vector multiplication bandwise
- in equidistant case for certain operators diagonals are constant

Sparse Banded Techniques

Bandwise windowed multiplication (variable, constant):




```
      DO 10 IM=1,M/K
       DO 100 I=1,K*M
100     Y(I)  =Y(I)  +DD(I)*X(I)+DL(I)*X(I-1)+DU(I)*X(I+1)
       DO 200 I =1,K*M
200     Y(I-M)=Y(I-M)+LD(I)*X(I)+LL(I)*X(I-1)+LU(I)*X(I+1)
       DO 300 I=1,K*M
300     Y(I+M)=Y(I+M)+UD(I)*X(I)+UL(I)*X(I-1)+UU(I)*X(I+1)
10    CONTINUE
```

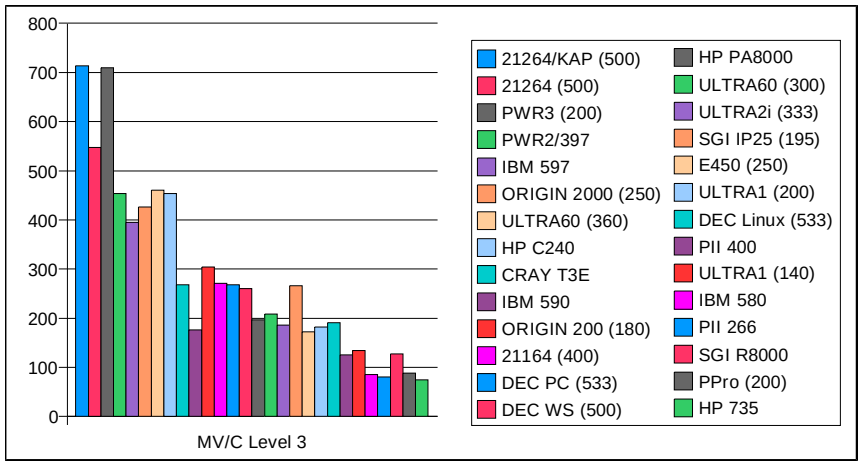
3D case	N	STO	LINE	SBB-V	SBB-C	**MG-V**	**MG-C**
DEC 21264	17^3	150	164	446	765	342	500
(500 MHz)	33^3	54	64	240	768	233	474
'**DS20**'	65^3	**24**	**72**	**249**	**713**	**196**	**447**
IBM RS6K/597	17^3	81	86	179	480	171	368
(160 MHz)	33^3	16	81	170	393	152	300
'**SP2**'	65^3	**8**	**81**	**178**	**393**	**150**	**276**
INTEL PII	17^3	28	29	56	183	48	136
(400 MHz)	33^3	24	29	53	139	47	116
'**ALDI**'	65^3	**19**	**29**	**54**	**125**	**45**	**101**

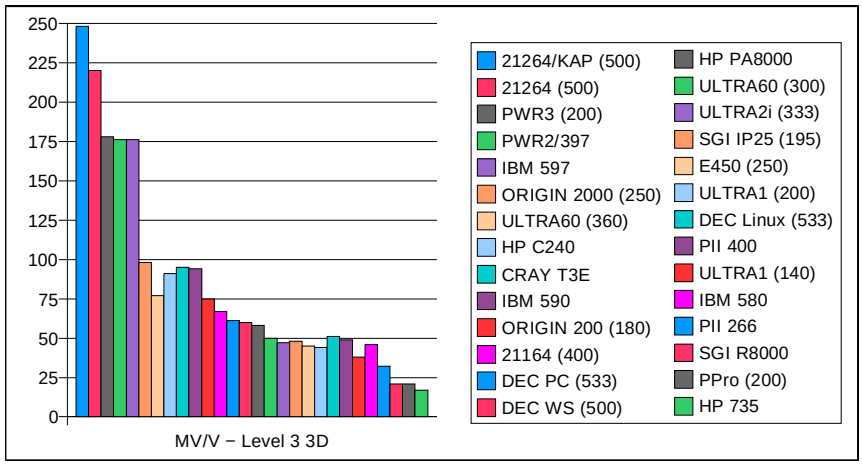
Question: How to use these techniques on complex domains?

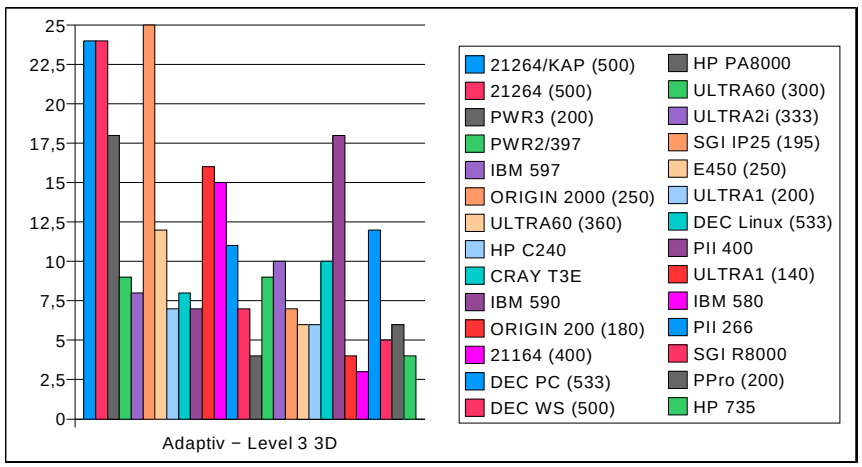
$\Longrightarrow$ **ScaRC**

FEAST Indices

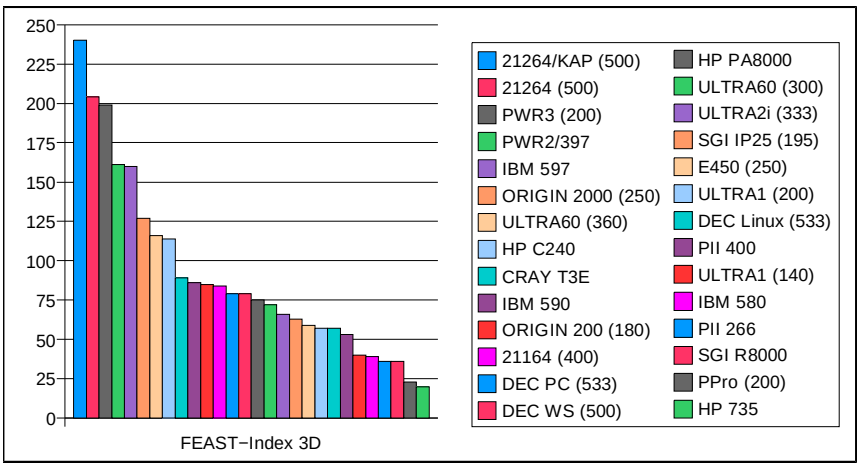
Framework of different numerical linear algebra tests for different problem sizes







FEAST Indices



Observations:

- MFLOP rate for standard sparse techniques far away from peak performance, depending on problem size and numbering of the unknowns
- higher performance achievable by appropriate techniques
- realization highly architecture dependent (processor, operating system, compiler version), no easy way
- simple PC partially faster than supercomputer nodes

Solver

Improvements for a faster simulation required in:

- implementation (higher performance)
- numerics and algorithm design

Requirements for the solver:

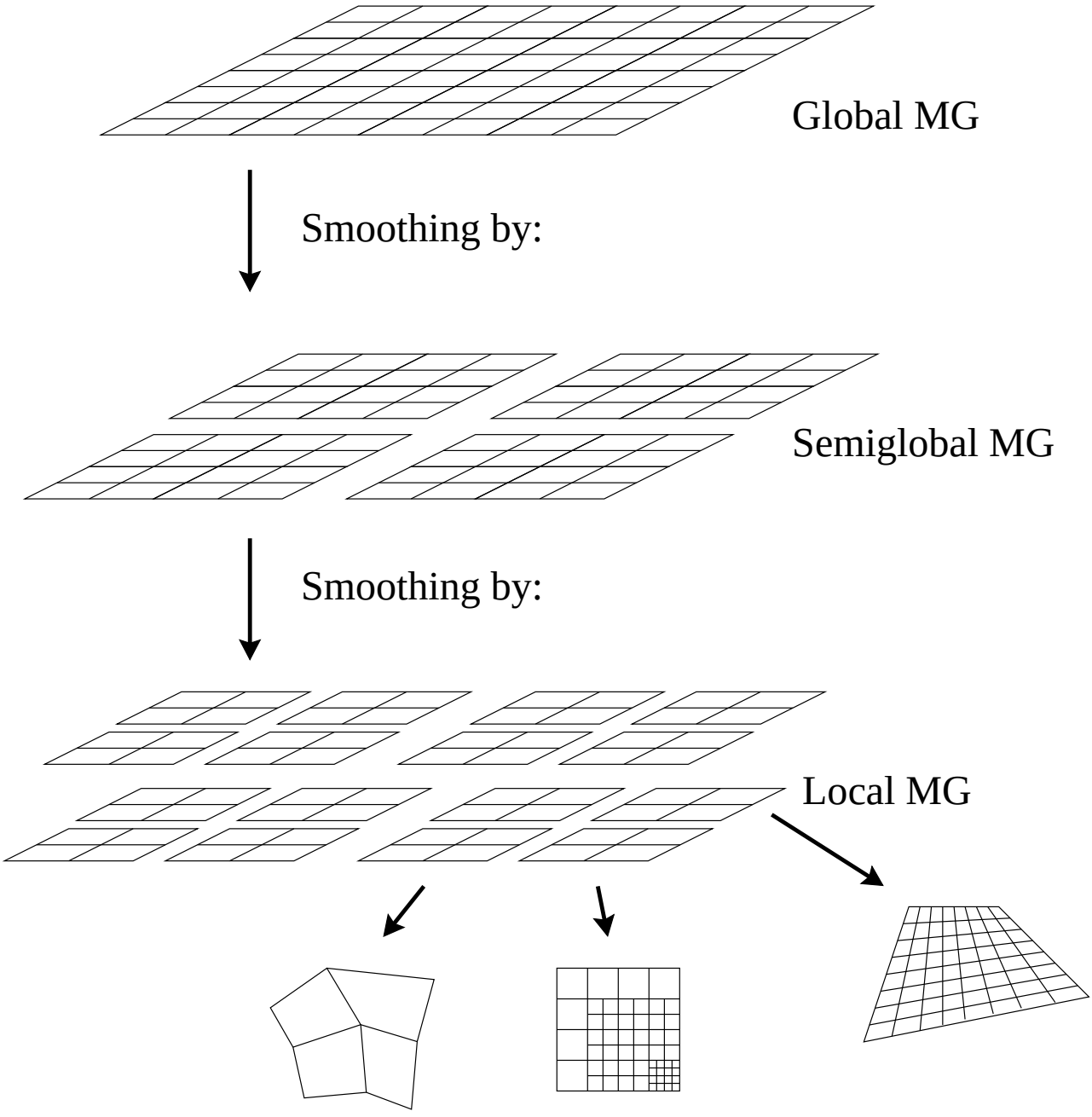
- Domain Decomposition required because of need for parallelization (w.r.t. CPU power and storage)
- sophisticated numercial algorithms (multigrid) required
- exploitation of local regularity required because of performance

ScaRC (**Sca**lable **R**ecursive **C**lustering):

- strong hierachical solver concept
- recursive divide and conquer strategy
  - find locally structured parts (high performance)
  - find locally anisotropic parts (hide locally, robustness)

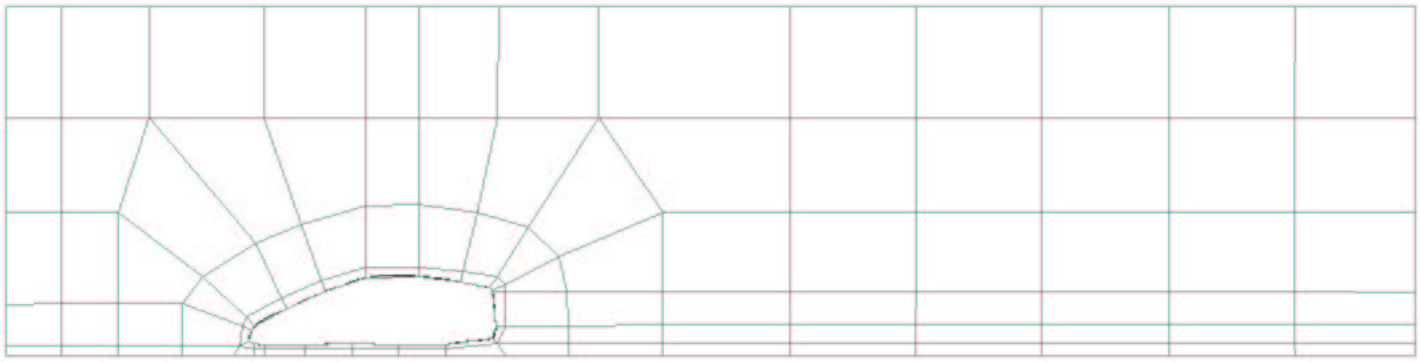
Solver

3 level ScaRC example:



ScaRC example

Example topology (2 level ScaRC, Poisson problem):



Comparison between conventional approach and ScaRC

	N_l	l	*Jacobi*		*SOR*		*ScaRC*	
			ite	ρ	*ite*	ρ	*ite*	ρ
$h \approx 10^{-4}$	8	2	**90**	0.858	**33**	0.653	**6**	0.093
	16	2	**100**	0.886	**45**	0.734	**6**	0.096
	32	2	**100**	0.894	**55**	0.776	**7**	0.116
$h \approx 10^{-7}$	8	2	**82**	0.844	**32**	0.647	**6**	0.092
	16	2	/	/	**42**	0.719	**6**	0.090
	32	2	/	/	**48**	0.748	**6**	0.080

Local convergence behavior

	'**Typ I**'	N_l	$AdiGS_l$	'**Typ II**'	$AdiGS_l$
$h \approx 10^{-4}$	$1:1$	32	0.02	$1:10$	0.03
		64	0.02		0.04
		128	0.02		0.04
$h \approx 10^{-7}$	$1:10^4$	32	0.02	$1:10^5$	0.09
		64	0.03		0.08
		128	0.03		0.08

FEAST conceptional notes

- closer to peak performance on recent and future architectures
- typical multigrid behavior (efficiency, robustness)
- parallelization directly included on low level
- automatic expert system for solver selection
- open for different adaptivity concepts and FEM error control
- application tools for several 'real life' problems (FEATFLOW 2.0 for CFD)

Outlook

- first version of FEAST end 1999 expected
- first version of Sparse Banded BLAS available
- first version of FEAST Indices available

Further information on:

- FEAST home page

  `http://www.featflow.de`

- The “Virtual Album of Flow Motion“

  `http://www.featflow.de/album/book.html`