Scalable Preconditioned Solvers for Internal and External Flow Computations on Many-Core Systems

Dr.-Ing. Achim Basermann, Dr. Hans-Peter Kersken, Melven Zöllner**

German Aerospace Center (DLR)
Simulation- and Software Technology
Distributed Systems and Component Software
Porz-Wahnheide, Linder Höhe, D-51147 Cologne, Germany

achim.basermann@dlr.de

**also RWTH Aachen University
DLR
German Aerospace Center

- Research Institution
- Space Agency
- Project Management Agency
Locations and employees

Germany: 6,900 employees across 33 research institutes and facilities at
- 15 sites.

Survey

- Internal and external flow computations at DLR
- Storage Schemes for sparse matrices
- The *Distributed Schur Complement* method (DSC)
- Experiments with TRACE and TAU matrices
- Conclusions
Parallel Simulation System TRACE

TRACE: Turbo-machinery Research Aerodynamic Computational Environment

- Developed by the Institute for Propulsion Technology of the German Aerospace Center (DLR-AT)

- Calculates internal turbo-machinery flows

- Finite volume method with block-structured grids

- The linearized TRACE modules require the parallel, iterative solution with preconditioning of large, sparse, non-symmetric real or complex systems of linear equations
Preconditioners for TAU: Background

- TAU: developed for the aerodynamic design of aircrafts by the DLR Institute of Aerodynamics and Flow Technology
- Unstructured RANS solver (Reynolds-averaged Navier-Stokes), exploits finite volumes
- Requires the parallel, iterative solution with preconditioning of large, sparse, real, non-symmetric systems of linear equations
- Solvers used: geometric Multigrid, AMG preconditioned GMRes
- Here: experiments with DSC methods
Storage Schemes for Sparse Matrices

Compressed Row Storage (CSR) and Block Compressed Row Storage (BCSR)

Matrix:

\[
\begin{array}{ccccccc}
1 & 0 & 0 & 2 & 0 & 0 \\
0 & 3 & 4 & 5 & 0 & 0 \\
0 & 0 & 0 & 0 & 6 & 7 \\
0 & 0 & 0 & 0 & 8 & 9 \\
\end{array}
\]

Non-zero values, row-wise:

1 2 3 4 5 6 7 8 9

Column indices, row-wise:

1 4 2 3 4 5 6 5 6

Row pointers:

1 3 6 8 10

→ TRACE and TAU apply BCSR with 5x5 blocks.

→ Advantage: less indirect addressing

→ Disadvantage: A few zeros are stored.
DSC Method (1)

Distributed matrix, 2 processors

Processor 1

Local rows

Internal rows

A₁

X₁

Local interface rows

External interface rows

Processor 2

A₂

X₂
DSC Method (2)

DSC Algorithm

Schematic view on each processor

BiCGstab or FGMRes iteration for all local rows (unknowns)

\[ \ldots \]

BiCGstab or GMRes iteration for the local interface rows (unknowns)

\[ \ldots \]

Matrix-vector multiplication: communication of external interface unknowns

\[ \ldots \]

Matrix-vector multiplication: communication of external interface unknowns

\[ \ldots \]
Preconditioning within the DSC algorithm

Processor $i$

Local rows

Local interface rows

Block incomplete $LU$ for the local rows

$U_i$

$L_i$

$U_{i,s}$

$L_{i,s}$

Block incomplete $LU$ for the local interface rows
DSC Method: Strong Scaling (CSR, real)
(Dual-processor nodes; Quad-Core Intel Harpertown; 2.83 GHz)

TAU matrix: $n=541,980; nz=170,610,950; \text{ threshold } = 10^{-3}; |\text{rel. residual}| < 10^{-5}$
DSC Method: CSR versus BCSR Format (real) (2x Intel XEON E5520 with 4 cores each, 2.26 GHz)

Results on 8 cores

TAU matrix:
n=541,980;
nz=170,610,950;
threshold = 10^{-3};
|rel. residual| < 10^{-7}
DSC Method: Effect of the Interface Iteration (real) (2x Intel XEON E5520 with 4 cores each, 2.26 GHz)

Results on 8 cores

TAU matrix:
n=541,980;
nz=170,610,950;
threshold = $10^{-3}$;
$|\text{rel. residual}| < 10^{-7}$
DSC Method: Strong Scaling (CSR, complex) (Dual-processor nodes; Quad-Core Intel Harpertown; 2.83 GHz)

TRACE matrix THD
(n=378,400; nz=45,456,500; threshold = 10^{-3}; |rel. residual| < 10^{-5})

TRACE matrix UHBR
(n=4,497,520; nz=552,324,700; threshold = 10^{-3}; |rel. residual| < 10^{-10})
DSC Solver: CSR versus BCSR Format
(2x Intel XEON E5520 with 4 cores each, 2.26 GHz)

linearTRACE matrix
(8 processes, dim = 56,240, nnz = 2.6 Mio)

- real
- real blocked (bs=5)
- complex
- complex blocked (bs=5)

Time in seconds

<table>
<thead>
<tr>
<th></th>
<th>total</th>
<th>ilut construction</th>
<th>solver iteration</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>real</td>
<td>4.5</td>
<td>1</td>
<td>76</td>
</tr>
<tr>
<td>real blocked</td>
<td>4</td>
<td>1</td>
<td>40</td>
</tr>
<tr>
<td>complex</td>
<td>3.5</td>
<td>1</td>
<td>32</td>
</tr>
<tr>
<td>complex blocked</td>
<td>3.5</td>
<td>1</td>
<td>29</td>
</tr>
</tbody>
</table>
linearTRACE Performance: Internal versus DSC Solver

dsc2011 solver for linearTRACE
(8 processes, test case "THD stator": dim = 0.8 Mio, nnz = 90 Mio)

Time in seconds

---

internal solver (gmres100, ssor(0.7,3))

# 140 iterations

---

dsc2011 (fgmres40, dsc gmres 5, ilut(0.01,1))

# 57 iterations

---

trace (setup matrix etc)
solver iteration
prec. preparation (ilut)
Conclusions

- Complex computations significantly faster than real ones (higher locality, better ratio of calculation to memory access)
- BCSR format application significantly outperforms CSR format application for real TRACE and TAU problems.
- DSC method achieves higher scalability and faster iteration than block-local methods.
- DSC method very suitable for TRACE and TAU problems
Questions?