Fast solvers for Eulerian convection schemes
Semester Thesis FS 2010

Andreas Hiltebrand
Supervisor: Holger Heumann
Professor: Ralf Hiptmair

Seminar for Applied Mathematics
ETH Zürich

23.12.10
Goal and discretization

- **Goal:**
  solve quickly pure advection and advection dominated problems

- **Discretization:**
  finite elements
  discontinuous Galerkin
  upwind formulation
Permutated block triangular systems

\[ Au = b \]
Block triangular systems

\[(PAP^\top) Pu = Pb\]
Block triangular systems

\[(PAP^T)Pu = Pb\]
Solution of lower block triangular systems

- lower block triangular systems
  \[ \Rightarrow \text{easily solvable} \]
  by block-wise forward substitution

- for \( i = 1, \ldots, n_B \)
  \[ u_i^B = (D_i^B)^{-1}(b_i^B - \sum_{j=1}^{i-1} L_{i,j}^B u_j^B) \]

\[
\begin{pmatrix}
D_1^B & 0 & \cdots & \cdots & 0 \\
L_{2,1}^B & D_2^B & \cdots & & \\
& \ddots & \ddots & \ddots & \\
& & \ddots & \ddots & \ddots \\
L_{n_B,1}^B & \cdots & \cdots & D_{n_B-1}^B & 0
\end{pmatrix}
\begin{pmatrix}
u_1^B \\
u_2^B \\
\vdots \\
u_{n_B-1}^B \\
u_{n_B}^B
\end{pmatrix}
= 
\begin{pmatrix}
b_1^B \\
b_2^B \\
\vdots \\
b_{n_B-1}^B \\
b_{n_B}^B
\end{pmatrix}
\]
Advection problems ↔ block triangular systems

- pure advection problem
  - finite elements
  - discontinuous Galerkin
  - upwind formulation

→ permutation of block triangular system
Introduction

Construction of permutation

Problems and results

Conclusions

Appendix

Advection problems ↔ block triangular systems

- pure advection problem
  - finite elements
  - discontinuous Galerkin
  - upwind formulation

⇒ permutation of block triangular system
Advection problems ↔ block triangular systems

- pure advection problem
  - finite elements
  - discontinuous Galerkin
  - upwind formulation

  \[ \Rightarrow \text{permutation of block triangular system} \]

- advection dominated problem

  \[ \Rightarrow \text{permutation of almost block triangular system, use block Gauss-Seidel method} \]

- construction of permutation?
1 Introduction
   • Goal and discretization
   • Solution of lower block triangular systems
   • Relationship between advection problems and block triangular systems

2 Construction of permutation
   • Matrix graph
   • Consistent ordering
   • Cycles and strongly connected components
   • Tarjan’s algorithm

3 Problems and results
   • Advection-diffusion equation

4 Conclusions
Matrix graph

capturing the dependencies $\Longrightarrow$ matrix graph
Matrix graph

capturing the dependencies $\Rightarrow$ matrix graph
Consistent ordering

Find an ordering $\pi$ such that

$$(i, j) \in E \Rightarrow \pi(i) < \pi(j) \quad \forall i, j \in V$$
Cycles and strongly connected components

- No cycles $\implies$ no problem (Topological sorting)
- Cycles $\implies$ no consistent ordering

Condensate strongly connected components

$\implies$ consistent ordering possible
Tarjan’s algorithm

- Determination of strongly connected components: Tarjan’s algorithm
- depth first search
- $\Theta(|V| + |E|)$
- here: $\Theta(n)$

$\longrightarrow$ construction of ordering: $\Theta(n)$
Steady state advection-diffusion equation in 2D/3D

\[-\epsilon \Delta u + \mathbf{b} \cdot \nabla u = f\]

- on the unit square \([0, 1]^2\)/unit cube \([0, 1]^3\)
- Dirichlet boundary conditions
- \(\mathbf{b}\) velocity field
- \(f\) source term
- \(\epsilon\) diffusivity coefficient
- \(u\) unknown scalar function
Fast solvers for Eulerian convection schemes

Andreas Hiltebrand
Compared methods

Krylov solver:
Biconjugate gradient stabilized method (BiCGSTAB)

Preconditioner:
- SOR: SSOR
- SORTSOR: sorting the system and then SSOR
- BLOCKGS: implicitly sorting the system and then block Gauss-Seidel method
Fast solvers for Eulerian convection schemes

Andreas Hiltebrand
Fast solvers for Eulerian convection schemes

Andreas Hiltebrand
Comparison of different parts

- Construct ordering
- Sort system
- Solve system
- Total time

Graph showing the comparison of different parts with time in seconds on the y-axis and n on the x-axis, where n is multiplied by 10^5.
Conclusions

- pure advection problems (with this discretization):
  permuted lower block triangular system
- permutation can be found in $\Theta(n)$
  using Topological sorting and Tarjan’s Algorithm
- advection dominated problems (with this discretization):
  permuted almost lower block triangular system
- solve system with block Gauss-Seidel preconditioner:
  only few iterations
- the more dominating the advection the more efficient
Appendix

Topological sorting

Tarjan's algorithm
Topological sorting

Algorithm 1: Topological sorting

input : graph $G = (V, E)$
output: ordering $\pi$
for $v \in V$ do $attr(v) = C$
for $v \in V$ do SetAttr($v$)
for $v \in V$ do
  if $attr(v) = C$ then $\pi(first) = v$
end

Procedure SetAttr($v$)
if $attr(v) = C$ then SetF($v$);
if $attr(v) = C$ then SetL($v$);
Topological sorting

Procedure SetF(v)

if \( \forall w \in \text{pred}(v) : \text{attr}(w) = F \) then
  \( \text{attr}(v) = F; \)
  \( \pi(\text{first}) = v; \)
  for \( w \in \text{succ}(v) \) do if \( \text{attr}(w) = C \) then SetF(w)
end

Procedure SetL(v)

if \( \forall w \in \text{succ}(v) : \text{attr}(w) = L \) then
  \( \text{attr}(v) = L; \)
  \( \pi(\text{last}) = v; \)
  for \( w \in \text{pred}(v) \) do if \( \text{attr}(w) = C \) then SetL(w)
end
Algorithm 2: Tarjan’s Algorithm

**input**: graph $G = (V, E)$

**output**: strongly connected components $components$

$index = 1$

$S = \{\}$

$components = \{\}$

**for** $v \in V$ **do**

**if** $index(v)$ is undefined **then** tarjan($v$)

**end**
Tarjan’s algorithm

**Procedure** `tarjan(v)`

- `index(v) = index`
- `lowlink(v) = index`
- `index = index + 1`
- `S.push(v)`

for `(v, v') ∈ E` do

  if `index(v') is undefined` then
    `tarjan(v')`  
    `lowlink(v) = min(lowlink(v), lowlink(v'))`
  end

  else if `v' ∈ S` then
    `lowlink(v) = min(lowlink(v), index(v'))`
  end

end
**Procedure** tarjan(v)

...  
**if** lowlink(v) = index(v) **then**  
  \( c = \{ \} \)  
  **repeat**  
    \( v' = S.pop() \)  
    \( c = c \cup \{ v' \} \)  
  **until** \( v' = v \)  
  \( \text{components} = \text{components} \cup \{ c \} \)  
**end**
**Tarjan’s algorithm**

```
\[
\begin{array}{cccc}
\text{v} & \text{index}(v) & \text{lowlink}(v) & S & c \\
1 & 1 & 1 & \{1\} & \{} \\
5 & 2 & 2 & \{1,5\} & \{} \\
7 & 3 & 3 & \{1,5,7\} & \{} \\
7 & 3 & 1 & \{1,5,7\} & \{} \\
5 & 2 & 1 & \{1,5,7\} & \{} \\
1 & 1 & 1 & \{1,5,7\} & \{} \\
\hline
\text{v} & \text{index}(v) & \text{lowlink}(v) & S & c \\
2 & 4 & 4 & \{2\} & \{7,5,1\} \\
4 & 5 & 5 & \{2,4\} & \{} \\
3 & 6 & 6 & \{2,4,3\} & \{} \\
3 & 6 & 4 & \{2,4,3\} & \{} \\
8 & 7 & 7 & \{2,4,3,8\} & \{} \\
\end{array}
\]
```

Fast solvers for Eulerian convection schemes

Andreas Hiltebrand
**Tarjan’s algorithm**

```
<table>
<thead>
<tr>
<th>v</th>
<th>index(v)</th>
<th>lowlink(v)</th>
<th>S</th>
<th>c</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>5</td>
<td>4</td>
<td>{2, 4, 3}</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>4</td>
<td>{2, 4, 3}</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>{2, 4, 3}</td>
<td>{3, 4, 2}</td>
</tr>
<tr>
<td>6</td>
<td>8</td>
<td>8</td>
<td>{6}</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>{6}</td>
<td>{6}</td>
</tr>
</tbody>
</table>
```

The algorithm works by tracking the index and lowlink of each vertex in the graph, along with the sets of vertices encountered during the depth-first search. The vertices are processed in reverse order of their index, and the sets are updated accordingly.