PARALLEL COMPUTING
OF
FLUID STRUCTURE INTERACTIONS PROBLEMS

Amit S. Baddi
November 28, 2005

Advisor: Prof. Hasan U. Akay

Department of Mechanical Engineering
Computational Fluid Dynamics Laboratory
Indiana University Purdue University Indianapolis
Outline

- Introduction to Fluid-Structure Interactions i.e. Aeroelasticity
- Algorithmic details of Loosely coupled approach for Aeroelastic flutter predictions
- Results Validation and comparisons
- Preconditioning formulation
- Large scale computations on grid computing environments
- Conclusions
- Acknowledgements
"Aeroelasticity is the phenomenon which exhibits appreciable reciprocal interactions (static or dynamic) between aerodynamic forces and the deformations induced in the structure of a flying vehicle, its control mechanisms, or its propulsion system." Bisplinghoff (1975)
Aeroelastic Flutter visual
Work Done in This Thesis

- Mesh based parallel code coupling search and interpolation program was developed coupling the fluid and structure codes.

- Multiple Programs Multiple Data (MPMD) paradigm of parallel computing was implemented.

- Non-conservative local time stepping was replaced by constant time stepping resulting in improved correlation with the results available in the literature.

- Termination criteria in case of mesh movement algorithm was modified resulting in reduced CPU time.

- The present density based flow solver was modified to solve for the low speed flows by preconditioning the time derivative term.

- Grid computing study was carried out on Teragrid in which multi-site execution was demonstrated.
Loosely coupled approach for simulating Aeroelastic flutter

- **CFD Solver (FAPEDA)**
  - Solve the flow field using Implicit Euler equations
  - Extract nodal pressures on the surface mesh
  - Impose the displacements on CFD surface mesh
  - Deform CFD volume grid using Mesh movement algorithm

- **Search & Interpolation (SINEDA)**
  - Interpolate pressures between non-similar surface meshes from fluid to structure domain
  - Interpolate displacements between non-similar surface meshes from structure to fluid domain

- **CSD Solver (SAPEDA)**
  - Calculate dynamic pressure forces on structure
  - Obtain dynamic response of structure using Newmark-Beta algorithm & mode-superposition
  - Extract nodal displacements on surface mesh

- **CSD Solver (SAPEDA)**
  - Calculate dynamic pressure forces on structure
  - Obtain dynamic response of structure using Newmark-Beta algorithm & mode-superposition
  - Extract nodal displacements on surface mesh
Computational Fluid Dynamics Solver - FAPEDA

- Unsteady Euler solver
- Unstructured tetrahedral grids
- Arbitrary Lagrangian Eulerian (ALE) formulation
- Cell centered based Finite Volume spatial discretization
- Backward Euler Implicit time integrations
- Local time stepping for Steady state solutions and Time accurate time stepping for Unsteady solutions
- Moving mesh for solution of moving boundary problems such as Fluid-Structure Interactions, etc.
- MPI for parallel communication
Computational Fluid Dynamics Solver - FAPEDA (Cont.)

Governing Equations for FAPEDA

The integral form of the Arbitrary Lagrangian-Eulerian (ALE) formulation of the 3D time-dependent inviscid fluid-flow equations:

$$\frac{\partial}{\partial t} \iiint_{\Omega} Q dV + \iint_{\partial \Omega} \bar{F} \cdot \hat{n} dS = 0$$

where, $Q = [\rho, \rho u, \rho v, \rho w, E]^T$ is the vector of conserved flow variables.

$$\bar{F} \cdot \hat{n} = F_n = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ e + p \end{bmatrix} \left[ (\bar{u} - \bar{w}) \cdot \hat{n} \right] + P \begin{bmatrix} 0 \\ n_x \\ n_y \\ n_z \\ a_n \end{bmatrix}$$

is the convective flux vector with ALE formulation.
The cell-centered finite volume formulation is employed.

The Backward Euler Implicit time integration scheme is used.

The discretized form is expressed as:

\[
\frac{\{Q\}^{n+1}V^{n+1} - \{Q\}^nV^n}{\Delta t} = -\int\int_{\partial\Omega} \{\bar{F}(Q)\} \cdot \hat{n} dS
\]

Rearranging,

\[
[A]\{\Delta Q\}^n = -\{R\}^n - \{Q\}^n \frac{\Delta V^n}{\Delta t}
\]

where,

\[
[A] = \frac{V^{n+1}}{\Delta t}[I] + \frac{\{\partial R\}}{\{\partial Q\}^n}
\]

\[
\{R\}^n = \int\int_{\partial\Omega} \bar{F}(Q)^n \cdot \hat{n} dS
\]
Mesh-Movement Algorithm

The mechanism of this method is that any two neighboring nodes in the mesh are connected by a spring and the spring stiffness is inversely proportional to the distance of the two nodes.

Predictor corrector scheme is used implementing Jacobi iterative procedure.

**Stiffness K**

\[ k_m = \left[ (x_j - x_i)^2 + (y_j - y_i)^2 + (z_j - z_i)^2 \right]^{-1/2} \]

**Displacements**

\[ \Delta x_i^{n+1} = \frac{\sum k_m \Delta \bar{x}}{\sum k_m}, \quad \Delta y_i^{n+1} = \frac{\sum k_m \Delta \bar{y}}{\sum k_m}, \quad \Delta z_i^{n+1} = \frac{\sum k_m \Delta \bar{z}}{\sum k_m} \]
Mesh-Movement Algorithm (continued)

Termination criteria for Jacobi iterative procedure is modified, reducing the CPU time by over 75%.

Previous termination criteria for Jacobi iterative procedure

\[
error = \sum_{\text{internal nodes}} \| \Delta x_{j}^{n+1} - \Delta x_{j}^{n} \|_2 \leq Tolerance
\]

Modified termination criteria for Jacobi iterative procedure

\[
error = \max_{\text{internal nodes}} \| \Delta x_{j}^{n+1} - \Delta x_{j}^{n} \|_2 \leq Tolerance
\]
Boundary Conditions

The characteristic boundary conditions are applied using Riemann invariants on farfield boundaries.

For the moving boundaries, the velocity and wall acceleration must be taken into account

\[
\vec{V}_w = \vec{V}_c - \vec{n} \cdot \left[ (\vec{V} - \vec{W}) \cdot \vec{n} \right]
\]

\[
\frac{\partial p}{\partial n} = -\rho \vec{n} \cdot \vec{a}_w
\]
Loosely coupled approach for simulating Aeroelastic flutter

CFD Solver (FAPEDA)
- Solve the flow field using Implicit Euler equations
- Extract nodal pressures on the surface mesh
- Impose the displacements on CFD surface mesh
- Deform CFD volume grid using Mesh movement algorithm

Search & Interpolation (SINEDA)
- Interpolate pressures between non-similar surface meshes from fluid to structure domain
- Interpolate displacements between non-similar surface meshes from structure to fluid domain

CSD Solver (SAPEDA)
- Calculate dynamic pressure forces on structure
- Obtain dynamic response of structure using Newmark-Beta algorithm & mode-superposition
- Extract nodal displacements on surface mesh
Finite element based dynamic structural solver

Quadrilateral shell elements used

Mode super-positioning is used to obtain the dynamic response of the structure

Newmark family of time integration used
The finite element discrete aeroelasticity element equation for a structural system can be written as

\[
[M]^{(e)} \dddot{q}^{(e)} + [C]^{(e)} \dot{q}^{(e)} + [K]^{(e)} q^{(e)} = R^{(e)}
\]

[M], [C] and [K] are system mass, damping and stiffness matrices

For static analysis, equation can be rewritten as

\[
[K] q = R
\]

For dynamic analysis, equation can be rewritten as

\[
[M] \dddot{q} + [C] \dot{q} + [K] q = R(t)
\]
Mode superposition method

1. Get the generalized eigenvalue solution

\[ [K]\{\phi\} = \omega^2 [M]\{\phi\} \]

2. Use first n modes to simulate structural response

\[ \{q(t)\} = [\{\phi_1\}, \{\phi_2\}, \ldots, \{\phi_n\}]\{X(t)\} = [A]\{X(t)\} \]

3. Get the generalized displacement solution

\[ \ddot{X}_i + 2\xi_i \omega_i \dot{X}_i + \omega_i^2 X_i = F_i \ast \]

*Remark:*
A Newmark-family of time integration scheme is used to obtain the solution at the \((n+1)\)th time step:

\[
\begin{align*}
\{M\}^{*} \{X\}^{n+1} &= \left[\frac{2}{\Delta t^2} \{M\}^{*} - \frac{(1 - 2\alpha)}{\Delta t} \{C\}^{*} - \left(\frac{1}{2} - 2\beta + \alpha\right) \{K\}^{*}\right] \{X\}^{n} + \\
\left[\frac{-1}{\Delta t^2} \{M\}^{*} + \frac{(1 - 2\alpha)}{\Delta t} \{C\}^{*} - \left(\frac{1}{2} + \beta - \alpha\right) \{K\}^{*}\right] \{X\}^{n-1} + \{F\}^{*}
\end{align*}
\]

<table>
<thead>
<tr>
<th>Method</th>
<th>(\alpha)</th>
<th>(\beta)</th>
<th>Stable Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Galerkin method</td>
<td>3/2</td>
<td>4/5</td>
<td>Always</td>
</tr>
<tr>
<td>The backward difference method</td>
<td>3/2</td>
<td>1</td>
<td>Always</td>
</tr>
<tr>
<td>The constant acceleration method</td>
<td>1/2</td>
<td>1/4</td>
<td>Always</td>
</tr>
<tr>
<td>The linear acceleration method</td>
<td>1/2</td>
<td>1/6</td>
<td>(\Delta t \leq \frac{2\sqrt{3}}{\omega_i})</td>
</tr>
</tbody>
</table>

Initial Condition:
\[
\{X\}_{t=0} = \{X_0\} \\
\{\frac{dX}{dt}\}_{t=0} = \{\dot{X}_0\}
\]

For Flutter Analysis

Either \(\{X_0\} \neq 0\) or \(\{X_0\} = 0\)

\(\{\dot{X}_0\} = 0\) \(\{\dot{X}_0\} \neq 0\)
Loosely coupled approach for simulating Aeroelastic flutter

**CFD Solver (FAPEDA)**
- Solve the flow field using Implicit Euler equations
- Extract nodal pressures on the surface mesh
- Impose the displacements on CFD surface mesh
- Deform CFD volume grid using Mesh movement algorithm

**Search & Interpolation (SINEDA)**
- Interpolate pressures between non-similar surface meshes from fluid to structure domain
- Interpolate displacements between non-similar surface meshes from structure to fluid domain

**CSD Solver (SAPEDA)**
- Calculate dynamic pressure forces on structure
- Obtain dynamic response of structure using Newmark-Beta algorithm & mode-superposition
- Extract nodal displacements on surface mesh
Function of SINEDA: Couple Non-similar meshes

Mesh in coupling region for Simulation Code 1 (FAPEDA)

SINEDA

Alternate Digital Tree (ADT) Geometric Search Algorithm & Bilinear Interpolation Algorithm

Mesh in coupling region for Simulation Code 2 (SAPEDA)

Element Type: Triangle
Usually dense meshes
Partitioned into blocks
Unstructured mesh

Element Type: Quadrilateral
Usually Coarse meshes
Unpartitioned
Structured mesh
Search & Interpolation - SINEDA

- Alternating Digital Tree (ADT) based geometric search algorithm
- Special Geometric selection criteria incorporating minimum distance and barycentric coordinates
- Bilinear interpolations for exchange of variables across non-similar meshes
- MPMD paradigm of parallel computing
- Based on Message Passing Interface (MPI)
Alternating Digital Tree (ADT) generation

A

B

C

<table>
<thead>
<tr>
<th>Left Son</th>
<th>Node</th>
<th>Right Son</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>A</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>B</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>C</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>A</td>
<td>2</td>
</tr>
<tr>
<td>0</td>
<td>B</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>C</td>
<td>0</td>
</tr>
</tbody>
</table>
Alternating Digital Tree (ADT) generation

<table>
<thead>
<tr>
<th>Left Son</th>
<th>Node</th>
<th>Right Son</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>A</td>
<td>2</td>
</tr>
<tr>
<td>0</td>
<td>B</td>
<td>3</td>
</tr>
<tr>
<td>0</td>
<td>C</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>D</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Left Son</th>
<th>Node</th>
<th>Right Son</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>A</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>B</td>
<td>3</td>
</tr>
<tr>
<td>0</td>
<td>C</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>D</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>E</td>
<td>0</td>
</tr>
</tbody>
</table>
Alternating Digital Tree (ADT) – SINEDA (contnd.)

Mesh is stored in a form of Alternating Digital Tree

CFD Mesh Element Binary Tree

CSD Mesh Element Binary Tree
Search & Interpolation - SINEDA (continued)

- Alternating Digital Tree (ADT) search

![Diagram of Alternating Digital Tree (ADT) search]

- Diagram showing nodes A, B, C, D, E, F, G, H, and X1, X2 axes.
Remote node need not lie exactly on the same plane of the local element.
Geometric selection criteria

Generalized model of a Point and Triangle in 3D space is used.

Remote node is projected onto the element plane.
Geometric selection criteria

\[
\text{error}_i = \frac{\text{dist}1^2 + \text{dist}2^2 + A_{\text{diff}}}{A_L} \leq \text{Tolerance}
\]
error_i = \frac{dist_1^2 + dist_2^2 + A_{diff}}{A_L} \leq Tolerance

In case if,

\left[ \frac{dist_1^2}{A_L} \right]_{E_1} \approx \left[ \frac{dist_1^2 + dist_2^2}{A_L} \right]_{E_2} \leq Tolerance

Inclusion of difference of Area term in error criteria is necessary.
But, difference of area is not the sufficient condition.

Inclusion of \( \text{dist1} \) and \( \text{dist2} \) in error criteria is also necessary.

\[
\text{error}_i = \frac{\text{dist1}^2 + \text{dist2}^2 + A_{\text{diff}}}{A_L} \leq \text{Tolerance}
\]
IPMD Paradigm of Parallel Computing – SINEDA (contnd.)

Local Process Rank (Global Process Rank)

MPI INTER-COMMUNICATOR

MPI INTRA-COMMUNICATOR
GARD 445.6 Wing: Pressure Distribution interpolated using SINEDA
GARD 445.6: Displacements interpolated using SINEDA
FSIEDA Testcase

AGARD445.6 Wing
TEST CASE

AGARD Wing 445.6

Tetrahedral CFD Mesh
168604 Nodes
1M cells

Quadrilateral CSD Mesh
840 Nodes
800 Elements
GARD 445.6 Wing

Grid convergence study based on steady state results

Coefficient of pressure distribution at $M = 0.499$ at root chord

Coefficient of pressure distribution at $M = 1.141$ at root chord
Flutter Analysis

Dynamic instability where-by the system extracts energy from the free stream flow producing a divergent response.

The computed flutter characteristics are presented in terms of velocity index $V_f$ which is defined as:

$$V_f = \frac{U_\infty}{b \omega_\alpha \sqrt{\mu}}$$
Flutter Analysis (contd.)

Stable response

Unstable response

AGARD 445.6 wing
$M_\infty = 0.901$ and $V_f = 0.35$
$AF = 0.875$

AGARD 445.6 wing
$M_\infty = 0.901$ and $V_f = 0.4$
$AF = 1.12$

Amplification Factor (AF) = \[
\frac{(2^{nd} \text{ Peak Gen. Disp.})_{\text{Mode1}}}{(1^{st} \text{ Peak Gen. Disp.})_{\text{Mode1}}}
\]
Linear Interpolation of Amplification Factor:

- AF is linearly interpolated with Dynamic pressure to obtain Flutter boundary.

![Graph showing Amplification Factor (AF) vs Dynamic Pressure (DP)]

- Vf = 0.35, DP = 1.396, AF = 0.875
- Vf = 0.38, DP = 1.823, AF = 1.12
- Vf = 0.4, DP = 1.615, AF = 1.12

M_\infty = 0.901
Aeroelastic Flutter Boundary at $M_{\infty} = 0.901$

Neutral response (LCO)

AGARD 445.6 wing
$M_{\infty} = 0.901$ and $V_f = 0.38$
AF = 1.0
Flutter Stability Boundary For AGARD 445.6 Wing

![Graph showing flutter stability boundary for AGARD 445.6 Wing](image-url)
Introduction

Time-marching schemes in the density based solvers provide good stability and convergence characteristics when solving compressible flows at transonic and supersonic Mach numbers.

At low speeds, however, system stiffness resulting from dissimilar particle and acoustic velocities resulting in large condition number causes convergence rates to deteriorate.

Convergence can be made independent of Mach number by altering the acoustic speeds of the system such that all eigenvalues become of the same order and the condition number is made to approach unity.
Reconditioning Formulation

Governing equations

Integral form of 3D Unsteady ALE based Euler equation of Fluid flow

\[
\frac{\partial}{\partial t} \iiint_{\Omega} \{W\} \, d\Omega + \iint_{\partial\Omega} \vec{F} \cdot \hat{n} \, dS = 0
\]

Preconditioned form of the 3D Unsteady ALE based Euler equation of Fluid flow

\[
\Gamma \frac{\partial}{\partial t} \iiint_{\Omega} \{Q\} \, dV + \iint_{\partial\Omega} \vec{F} \cdot \hat{n} \, dS = 0
\]

where, \(\Gamma\) = Preconditioning matrix

\[
\{Q\} = \{p \quad u \quad v \quad w \quad T\}^T = \text{Preconditioned flow variables vector}
\]
reconditioning Formulation

Dual time stepping

To maintain the time accuracy of the equations in case of transient solution dual
time stepping is utilized and preconditioning is applied to pseudotime.

\[
\frac{\partial}{\partial t} \iiint_{\Omega} \{W\} d\mathcal{V} + \Gamma \frac{\partial}{\partial \tau} \iiint_{\Omega} \{Q\} dV = -\iiint_{\partial \Omega} \vec{F} \cdot \hat{n} dS
\]

where, \( \{W\} = \{\rho, \rho u, \rho v, \rho w, e\}^T \) = Conserved flow variable vector

The discretized form is:

\[
\begin{bmatrix}
\Delta t & \frac{\partial W}{\partial Q} \\
\Delta \tau & \frac{\partial R}{\partial W} \\
\end{bmatrix}
\begin{bmatrix}
\{\Delta Q^m\} \\
\end{bmatrix}
= \begin{bmatrix}
\{R^m\} - \frac{\partial R}{\partial W} \{W^m\} + \frac{\partial W}{\partial Q} \{W^n\} \\
\end{bmatrix}
\]

As \( m \to \infty, \) \( \{\Delta Q^m\} \to 0 \) \( \therefore \text{LHS} = 0 \)

\( W^m \to W^{n+1} \)
Application of FSIEDA on TeraGrid (TG)
MSF TeraGrid Extensible Terascale facility (www.teragrid.org)

TeraGrid sites used in this project:
- Computation sites
- Log-in site (also NOC)
Multisite execution of FSIEDA on TG

Local Process Rank (Global Process Rank)
Conclusions

A loosely coupled procedure is developed for prediction of Aeroelastic flutter boundary providing an effective alternative over strongly coupled approach.

ADT based efficient Geometric search and Interpolation program SINEDA is developed for obtaining mesh-based code coupling on non-similar meshes.

Flutter analysis was implemented by choosing initial perturbation of the structural system and examining whether the initial perturbation will decay, grow or maintain neutral conditions to determine the flutter conditions. The results obtained show good correlation with previous works of Batina and experimental results.

The accuracy improvement was obtained especially in the supersonic regime by implementing the constant time stepping over the local time stepping.

The modified termination criteria reduced the CPU time by over 75%.

Preconditioning formulation was developed by modifying the current density based transient Euler solver with ALE formulation to solve for the low speed i.e. incompressible flows.

Explored the potential application of TeraGrid for solving large scale problems.

Multi-site execution was studied for solving potentially large scale problems utilizing multiple clusters on TeraGrid.
Acknowledgements

Prof. Hasan Akay

Prof. Akin Ecer, Prof. Hsu and Dr. Erdal Oktay

Dr. Erdal Yilmaz and Resat Payli

All the Faculty members and staff of the Mechanical Engineering Department

TeraGrid (www.teragrid.org) access to this project was supported by National Science Foundation (NSF) grant, TG-CTS050003T, under the following programs:

Partnerships for Advanced Computational Infrastructure
Distributed Terascale Facility (DTF)
Terascale Extensions: Enhancements to the Extensible Terascale Facility