Project 5.3:
Large-Scale Optimization for Vehicle System Design

QUAD MEMBERS

P. Papalambros, UM
N. Michelena, UM
S. Nelson, UM
J. Reyer, UM
R. Krishnamachari, UM

T. Bailey, TARDEC
K. Ciarelli, TARDEC
M. Salman, GM
T. Kenney, Ford
M. Chiredast, Ford
Overview

- Project Motivation and Objectives
- Difficulties for Systems Design
- Methodology for Systems Design Optimization
- Model-Based Partitioning
  - Hypergraph Approach
  - IP Approach
- Coordination
  - Overlapping
  - Sequentially Decomposed Programming (Sig Nelson)
Motivation and Objectives

- **Motivation:** Complex systems design entails the iterative use of CAE tools originally conceived as stand-alone tools for component or subsystem analysis on a single computer platform.

- **Objective:** Develop and implement a design methodology for complex systems that uses decision-support techniques such as mathematical nonlinear optimization and robust search algorithms.
Vehicle System Design Optimization

- **Design Criteria**
  - Fuel economy
  - Performance metrics
  - Handling metrics
  - Cost metrics

- **Design Variables**
  - Engine parameters
  - Transmission parameters
  - Suspension parameters
  - Track parameters
Difficulties for Systems Design

- Proper system design models are difficult to formulate
- Several models/simulations may be needed to predict a systems’ behavior, performance, reliability, cost, etc.
- Good predictive models are computationally intensive
- Gradient-based optimization techniques cannot deal with noisy or discontinuous responses
- More robust search methods may be too computationally expensive
- ...
Systems Design Optimization

Issues to deal with:
- Design Problem Formulation
- Multiple Simulations
- Computational Cost
- Non-Smooth Models

Techniques:
- Distributed Computation
- Partitioning and Coordination
- Hybrid Models & Algorithms
Methodology for Systems Design Optimization

Simulation or Analysis Models → Functional Dependence Table → Optimal Design Problem → Coordinated Solution

Problem Partitioning → Coordination Strategies → Hybrid Models & Algorithms

Optimal System Design → Distributed Computation
Partitioning and Coordination

Non-Hierarchically Partitioned Problem

- Subproblem local variables
- ... linking variables
- Subproblem local variables

Hierarchically Partitioned Problem

- Master Problem linking var’s
- linking variables
- Subproblem local var’s
- ... Subproblem local var’s

Coordination Procedure
Model-Based Problem Partitioning

- Decomposition Analysis
  - Hypergraph-based problem partitioning
    - Account for computational demands & resources
  - IP-based problem partitioning
    - Hierarchical partitioning

- Decomposition Synthesis
  - Identify design criteria & constraints based on problem structure
Hypergraph-Based Problem Partitioning

- Minimize interconnection among subproblems
- Balance size of partitions

$x_1, x_2, x_3$ are linking variables
Graph-Based Problem Partitioning

http://arc.engin.umich.edu/decomp-docs/decomp.html

Input Data

Browse or specify the path of the file containing the Functional Dependence Table (FDT). The format of this file is available here.

- Path of file containing Functional Dependence Table
- Number of design relations
- Number of design variables
- Design relation weights
- Design variable weights
- Design relation weights equal to one
- Design relation weights equal to the number of variables in relation
- Design relation weights equal to the sum of weights in relation
- Variable weights equal to one
- Variable weights equal to the number of variables
- Variable weights equal to the sum of weights in variable

Enter the number of subproblems you wish the design problem to be divided into.

- Number of subproblems
- Isolate size subproblems
- Allowed deviation of subproblem size (%)
Powertrain Graph-Based Partitioning

http://arc.engin.umich.edu/decomp-docs/decomp.html

SP1: Wheel model, powertrain and vehicle geometry relations; acceleration, starting gradeability, and cruising velocity criteria

SP2: Engine relations

SP3: Torque converter, transmission, and powertrain geometry relations

SP4: Engine relations; anti-lug constraint; emissions and fuel consumption criteria

22 linking variables
IP-Based Problem Partitioning

- Minimize size of master problem (= # variables + # relations)

subject to:

- Constraint on relative size of subproblems
- Relations in master problem only depend on linking variables
- Relations in master problem do not belong to subproblems
- A local variable must be in subproblem if its relations are
- A local variable must not be in subproblem if its relations are not
- Each subproblem must have at least one local variable
- Each relation belong to one and only one cluster
- Each variable belong to one and only one cluster
param K > 0;                # Number of clusters (master + # subproblems)
param J > 0;                # Number of variables
param V > 0;                # Number of relations
param KS > 0;               # Relative size constant (1<= KS <=3)
param a{1..V, 1..J} binary; # Relation dependence
param d{1..V} integer;      # Number of variables in each relation
var e{1..K, 1..J} binary;   # Variables assigned to clusters
var s{1..K, 1..V} binary;   # Relations assigned to clusters

minimize objective:
# Minimize size of master problem
sum{j in 1..J} e[1,j] + sum{v in 1..V} s[1,v];

subject to
# Constraint on relative size of subproblems
g1{k1 in 2..K, k2 in 2..K: k1 <> k2}: KS*(sum{v in 1..V} s[k1,v] + sum{j in 1..J} e[k1,j])
   >= (sum{v in 1..V} s[k2,v] + sum{j in 1..J} e[k2,j]);

# Relations in master problem only depends on linking variables
g2{v in 1..V}: sum{j in 1..J} a[v,j]*e[1,j] >= d[v]*s[1,v];
# Relations in master problem do not belong to subproblems

g3{v in 1..V}: sum{j in 1..J} a[v,j]*e[1,j] <= d[v] - sum{k in 2..K} s[k,v];
# A local variable must be in subproblem if its relations are

g4{k in 2..K, j in 1..J}: e[k,j] = ((sum{v in 1..V} a[v,j]*s[k,v]/(sum{v in 1..V} a[v,j])) - e[1,j]);
# A local variable must not be in subproblem if its relations are not

g5{k in 2..K, j in 1..J}: e[k,j] <= (sum{v in 1..V} a[v,j]*s[k,v]);
# Each subproblem must have at least one local variable

g6{k in 2..K}: sum{j in 1..J} e[k,j] >=1;
# Each relation belong to one and only one cluster

h1{v in 1..V}: sum{k in 1..K} s[k,v] = 1;
# Each variable belong to one and only one cluster

h2{j in 1..J}: sum{k in 1..K} e[k,j] = 1;
HEP IP-Based Partitioning

http://arc.engin.umich.edu/decomp2-docs/decomp.html
Hierarchical Overlapping Coordination

(in collaboration with Oakland University)

- Uses two or more model decompositions, which should have specific characteristics
- Each decomposition “coordinates” the others, i.e., there is no Master Problem
- Convergence for convex programs with linear constraints depends on model decomposition, for example:
  - Reduced number of linking variables
  - Disjoint set of linking variables
- Ideal for QP solution in SQP and for trust region methods
\[ \min f(x) \quad \text{s.t.} \quad Ax = c \quad \text{and} \quad H_\alpha x = y_\alpha \]

\[ \min f(x) \quad \text{s.t.} \quad Ax = c \quad \text{and} \quad H_\beta x = y_\beta \]

**Decomposition of Design Vector**

\[ x = x_l \]

\[ x_{l-A} \quad x_A \]

\[ x_B \quad x_{l-B} \]

\[ y_\alpha \leftrightarrow x_A \]

\[ y_\beta \leftrightarrow x_B \]
Decomposition by Components

MISSION SPECS

POWERTRAIN SYSTEM

DRIVING CYCLE

VEHICLE “PARAMETERS”

ENGINE

TRANSMISSION

FINAL DRIVE

VALVETRAIN

GEARING

WHEELS

MANIFOLD

CLUTCHES

CYLINDER BLOCK

TORQUE CONVERTER

DIFFERENTIAL
If algorithm is started with a feasible point, then at each stage of the process, problems $\alpha$ and $\beta$ will have non-empty feasible domains.

If sequences $\{x_{\alpha n}\}$ and $\{x_{\beta n}\}$ result from solving problems $\alpha$ and $\beta$, and $f^{\text{opt}} = \min f(x) : A x = c$, then:

- $f(x_{\alpha n}) \geq f(x_{\beta n}) \geq f(x_{\alpha n+1})$
- $\lim f(x_{\alpha n}) = \lim f(x_{\beta n}) = f^* \geq f^{\text{opt}}$ as $n \to \infty$

Any accumulation point $x^*$ of either $\{x_{\alpha n}\}$ or $\{x_{\beta n}\}$ solves both problem $\alpha$ and problem $\beta$.

Sufficient condition for HOV convergence:

\[
\begin{bmatrix}
A \\
H_{\alpha} \\
H_{\beta}
\end{bmatrix}
\]

is a full (row) rank matrix.
Distributed Computing in Design

- **Design Analysis**
  - Parallelization at the program level (e.g., Do-loop level)
  - Parallelization at the job level (e.g., by operating system)
  - Coarse grain parallelization of application by domain decomposition or substructuring

- **Design Synthesis**
  - Parallelization of sensitivity calculations (i.e., operator splitting)
  - Parallelization for computation of design criteria & constraints
  - Partitioning and coordination of optimization model
Hybrid Models and Algorithms

- **Surrogate Models:**
  - Obtained from high-fidelity models by varying convergence tolerances
  - Response surface or neural networks

- **Hybrid Search Algorithms:**
  - Combine gradient-based local search algorithm with non-gradient/global algorithms such as:
    - Pattern search, trajectory methods
    - GA, SA