Optimization of the direct fuel injection for the 4-stroke combustion engine

Josef Dubský, 10.10.2011
Institut für Verbrennungskraftmaschinen
Used tools:
- 3D CFD Simulation of the injection, charge motion and wall film deposition performed by StarCD 4.14.
- Automatized simulation process including postprocessing and evaluation allows analysis of multiple designs in parallel.

Objective:
Optimal targeting of individual injector spray mostly with respect to:
- Air/fuel mixture homogenization
- Low wall film deposition
- Charge motion characteristics (e.g. Tumble, etc.)

⇒ Lower fuel consumption
⇒ Lower emissions
Model properties:
- 3D
- Transient
- Compressible
- Turbulent (k-eps)
- Lagrangian multiphase
- Wall film deposition
- Moving geometry

Main analyzed quantities:
- Lambda distribution
- Evaporation rate
- Tumble intensity
- Wall film deposition
Currently each injector model has to be calibrated and validated through experimental data provided by injector supplier. This data include:

- Spray footprint in defined plane
  -> Mass flow distribution

- Spray images sequence
  -> Velocity scaling

- Injector geometry
  -> Coordinates of each single spray in the measuring plane
Optimization

Model parametrization

- Necessary data for the injector modelling are predicted based on the available data.
- No experimental measurements of every single injector necessary.
Optimization

Simulation process definition

- Esice model update - import of modified vertices positions (Locally)
- Complete Prostar model update with morphed mesh (Locally)
- Estimation of mass flow distribution for the injector based on empirical data and approximation

Task 1

- Do the morphing and export .vrt file for Star (Locally)
- Esice transient
- Prostar transient
- MassFlow Distrib
- Star transient

- Static
- Prostar Static
- Star Static
- Evaluate

- Import the updated positions of vertices (.vrt file) from the morpher to Prostar and re-smooth the mesh (Locally)
- Run Star Static (Locally or HWW)
- Evaluate the Alfa_K and Tumble (Locally)
- Run Star Transient (Locally or HWW)
Optimization

Selection of optimization strategy

Important aspects:

• Each design takes ~12 hours to evaluate (scalability will be discussed further).
• Optimization strategy allowing parallel designs evaluation is desired.
• The number of runs has to be kept low.

Selected strategy: DOE / Approximation / Genetic algorithm / Refinement

• **DOE** analyzes the design space with “probes” and helps to understand the effects of the parameters.
• When sufficient number of results from analyzed cases is available, **Approximation** model can be created to predict the results without running the CFD code.
• **Genetic algorithm** which requires high number or runs can be quickly executed on the approximation model.
• Predicted optimum can be verified by **refined** DOE in some constrained part of the design space.
• Whole process can be repeated until desired accuracy is reached.

Optimum found
Simulation performance

Mesh movement in StarCD

- Dedicated process (or even multiple processes) are used to calculate the mesh movement and mesh topology changes during the simulation.
- Mesh movement includes cell layers **addition/removal**, **moving** of mesh nodes, **sliding interfaces** in the valve curtain region and also **topology changes** (e.g. after valve closing).
- Multiple mesh movement events may be pre-calculated in advance: **events-ahead**, **events-cpus**
- Eventually all the necessary meshes can be prepared and written out before the simulation: **WriteAhead**
- It’s crucial to use such settings, that the mesh mover has **always enough events ready for solver**!

Mesh mover using N CPUs

Solver using N CPUs

Simulation performance

Mesh movement in StarCD

- Dedicated process (or even multiple processes) are used to calculate the mesh movement and mesh topology changes during the simulation.
- Mesh movement includes cell layers **addition/removal**, **moving** of mesh nodes, **sliding interfaces** in the valve curtain region and also **topology changes** (e.g. after valve closing).
- Multiple mesh movement events may be pre-calculated in advance: **events-ahead**, **events-cpus**
- Eventually all the necessary meshes can be prepared and written out before the simulation: **WriteAhead**
- It’s crucial to use such settings, that the mesh mover has **always enough events ready for solver**!
Simulation performance

Domain decomposition

- Domain decomposition must be optimal during whole simulation.
- Mesh movement and topology changes during simulation must be considered.
- Decomposition with “smallest“ interfaces doesn’t have to be optimal.
Simulation performance
Identification of bottlenecks

Performance increase measures

Interconnect traffic
Achieved through efficient domain decomposition
- $2^n$ sets work best
- all CPUs on the node should be used
- domain decomposition must be optimal during whole simulation

Binding the solver processes to the CPUs
- through instructions to MPI
- Reduces runtime by $\sim 10\%$ and effect increases with more CPUs

I/O operations
- Limiting the frequency of periodic outputs only to the necessary minimum.
  - E.g. on 32CPUs writing the output file every 2 Deg CA increases overall runtime by $+20\%$
- Running Mesh movement processes on a dedicated node
  - avoids I/O sharing between solver master process and mesh mover
  - Can bring as much as $-40\%$

够 enough capacity for mesh movement
- Using the WriteAhead capability of the Mesh mover is not recommended. CPU demands of Mesh mover are lowered, but I/O demands do increase. (23GB for given model)
- Good practice is to set events-ahead and events-cpus, both to 3.
  - This is still sufficient for the 2mil cells model running at 48CPUs and keeps the whole mesh mover node busy at $\sim 80\%$.

Notes: Benchmarking performed on a model with 1.8 mil cells at the HLRS Ahuja cluster (Nehalem Xeon CPUs), HP MPI
Simulation performance

Scaling analysis

- After applying all the measures a model which previously didn’t scale well above 16 solver CPUs scales efficiently up to 32 solver CPUs.

- Wall clock time of the simulation was reduced to ~50% compared to initial setup.

- Even 48 solver CPUs can bring some performance increase, though with lower efficiency.

Notes: Benchmarking performed on a model with 1.8 mil cells at the HLRS Ahuja cluster (Nehalem Xeon CPUs), HP MPI
Final conclusions

**Project summary:**
- Standardized simulation process including automatic postprocessing was established.
- Thanks to the proposed optimization strategy the number of designs needed to find the optimum is kept as low as possible.
- Through improvement in the performance scaling and the fact, that during the optimization the designs can be evaluated in parallel, reasonable time for the whole optimization can be achieved.

**Further development:**
- Feasibility study for LES turbulence modeling.
- Including the simple combustion modeling.
- Scaling up to ~100 CPUs