Entwicklung einer automatisierten und parallelen Test-Suite für MaMiCo

Projekt Parallelrechnerevaluation

Leonard Hannen, Felix Maurer

Fachbereich Informatik
Universität Hamburg

04.03.2021

Universität Hamburg

Gliederung

Zielsetzung: Compilation Tests

- Automatisches Kompilieren
- Verschiedene MaMiCo-Konfigurationen
- Vergleich versch. Compiler(-Parametrisierungen)

Zielsetzung: Unit Tests

- Vereinheitlichtes Unit-Testing-System für MaMiCo
- Testinterface zur Handhabung von
  - Testfunktionen
  - Dummy-Testobjekten (Mocks)
- Automatisierte Verarbeitung von
  - verwendeten Mocks
  - gefundenen Fehlern

Zielsetzung: Simulation Tests

- Mehrere komplexe Testszenarien
- Abgleich mit Kontrolldatensatz
- Regelmäßige, automatisierte Ausführung auf HSU-Cluster

Aktueller Zwischenstand: Compilation Tests

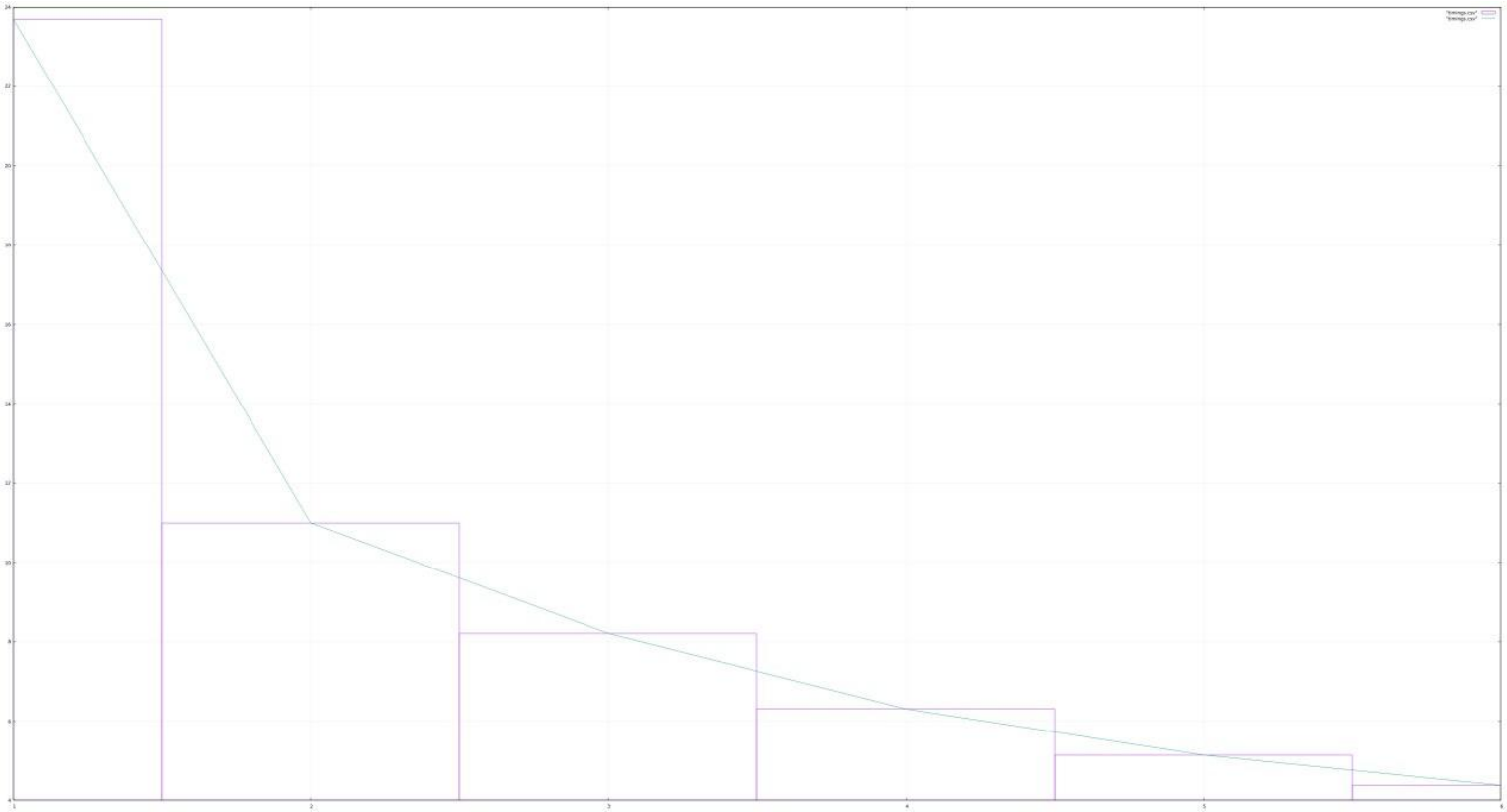
- CMake
  - Konfiguration von MaMiCo- und Compilerflags
  - Automatisierte Einbindung von
    - Include-Pfaden
    - SCons-Skript zur Kompilation von SimpleMD
  - Opional: Integration von pybind11
- CompileTest.sh
  - Fehlerausgabe
  - Iteriert über Liste von Compilern:
    - `cmake`, dann `make`
    - Simple (Konsolen-)Fehlerbehandlung
  - Prüft sowohl Couette-Szenario als auch pybind11-Library
  - Prüft parallelen und sequentiellen Build von Couette

Aktueller Zwischenstand: Unit Tests

- Programmbestandteile:
  - UnitTestingService
  - MockService
  - UnitTestInterface und UnitTestImpl
- Einfache Schablone zum Erstellen eigener Unit Tests
- Examplarische Unit Tests für versch. MaMiCo-Klassen
- Parallelisiert (über Mocks der getesteten Klasse)

Aktueller Zwischenstand Unit Tests (2)

- Momentaner Speedup-Graph



Aktueller Zwischenstand: Simulation Tests

- Abgleich von Test- und Referenzwerten
  - Datensätze liegen als CSV-Dateien vor
  - Vergleich unter Zuhilfenahme eines Pythonskripts mit Pandas

Nächste Schritte

- Compilation Tests
  - Spack für Compilerversionsverwaltung
  - evtl. weitere konfigurierbare Parameter einführen: Dimension, MD Solver, ...
- Unit Tests
  - Unit Tests komplexerer Klassen implementieren: CellMappings, CellTraversal (, ParticleInsertion)
  - MockService: Selektierbarkeit von Mock-Subsets
- Simulation Tests
  - Abgleich mit analytischen Daten aus Kontinuumslöser
  - Qualitätsevaluation der Daten mit Hilfe von durchschnittlichem Fehlerwert
  - Behandlung von Simulationsabstürzen
  - Einrichten eines cronjobs auf dem HSU-Cluster