Optimizing a Superscalar System using Multi-objective Design Space Exploration

Horia Calborean

Prof. Lucian Vinţan

Ralf Jahr

Prof. Theo Ungerer

Universität Augsburg University

Advanced Computer Architecture & Processing Systems Research Lab

http://acaps.ulbsibiu.ro/research.php

Outline

- Design space exploration
  - Multi-objective optimization
  - Algorithm comparison
  - Metrics used
- Methodology and tools
  - FADSE
  - GAP and GAPtimize
- Results
  - With GAP simulator
  - With GAP and GAPtimize
- Conclusions

Design space exploration (DSE)

- Number of architectural parameters has risen
- Huge number of possible configurations
  - 50 parameters with 8 values =>
    - 2^{150} possible configurations
    - Exhaustive evaluation impossible
- Manual design space exploration infeasible

- Solution: **heuristic search algorithms**

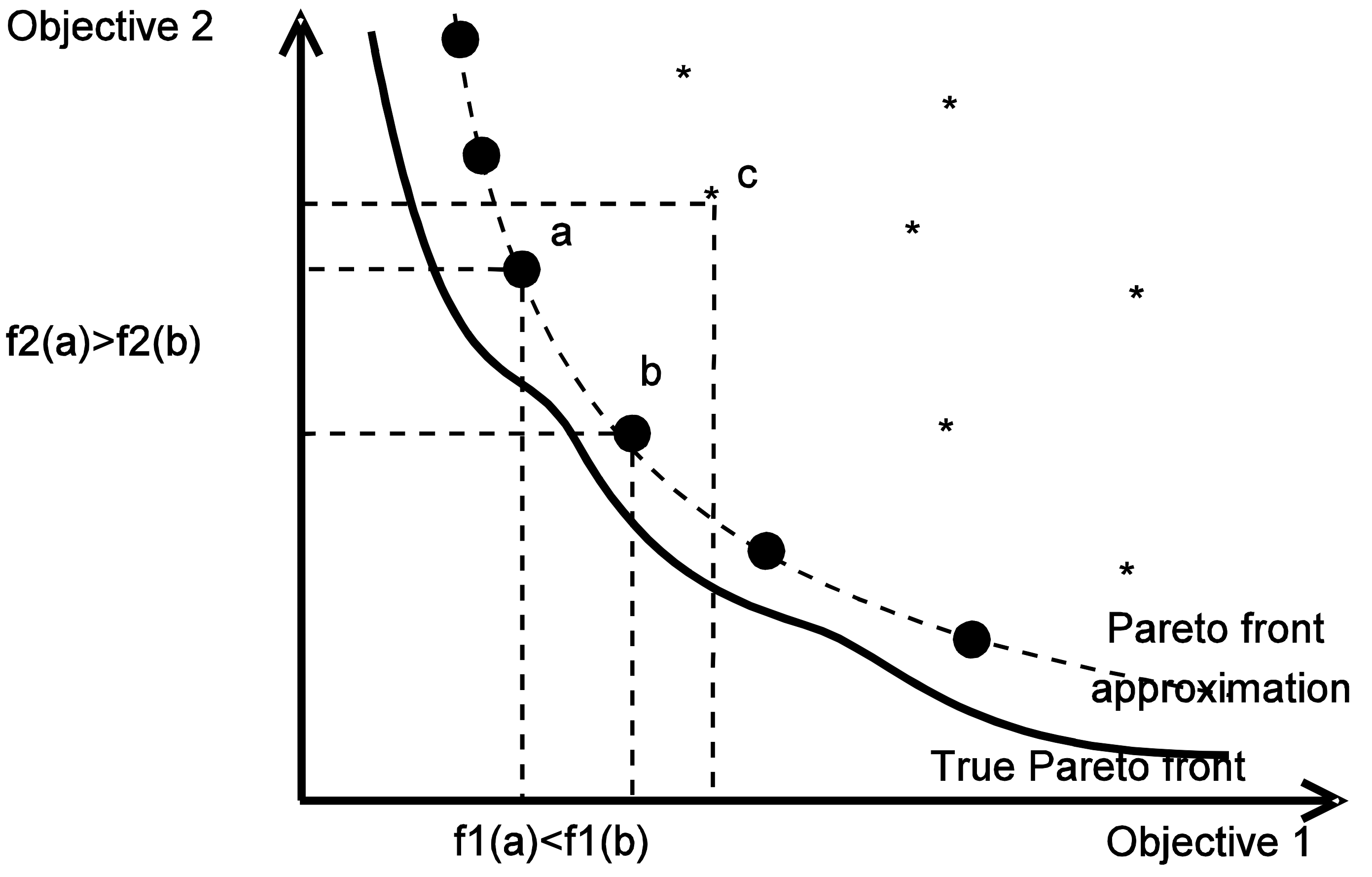
Multi-objective DSE

- Performance evaluation has become a complex **Multi-objective** evaluation (speed, power consumption, area integration, etc.)
- Multi-objective search algorithms are used

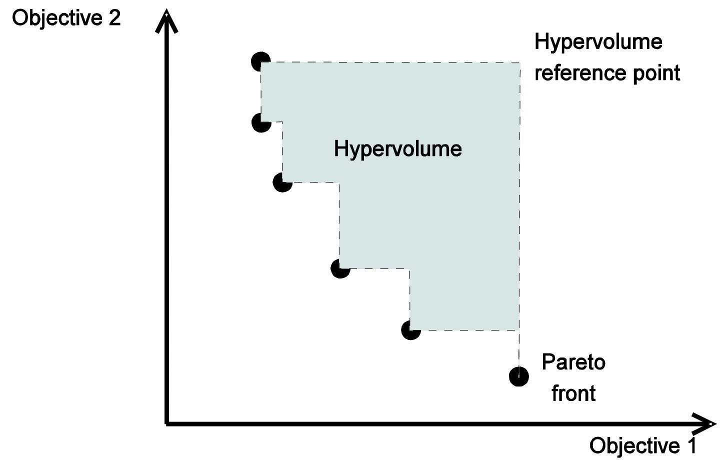
This work

- We perform a comparison between three well known search algorithms:
  - NSGA-II – genetic algorithm
  - SPEA2 – genetic algorithm
  - SMPSO – particle swarm optimization (PSO)
    - inspired by the flight of birds

Basic notions about Pareto front



Metrics used: hypervolume

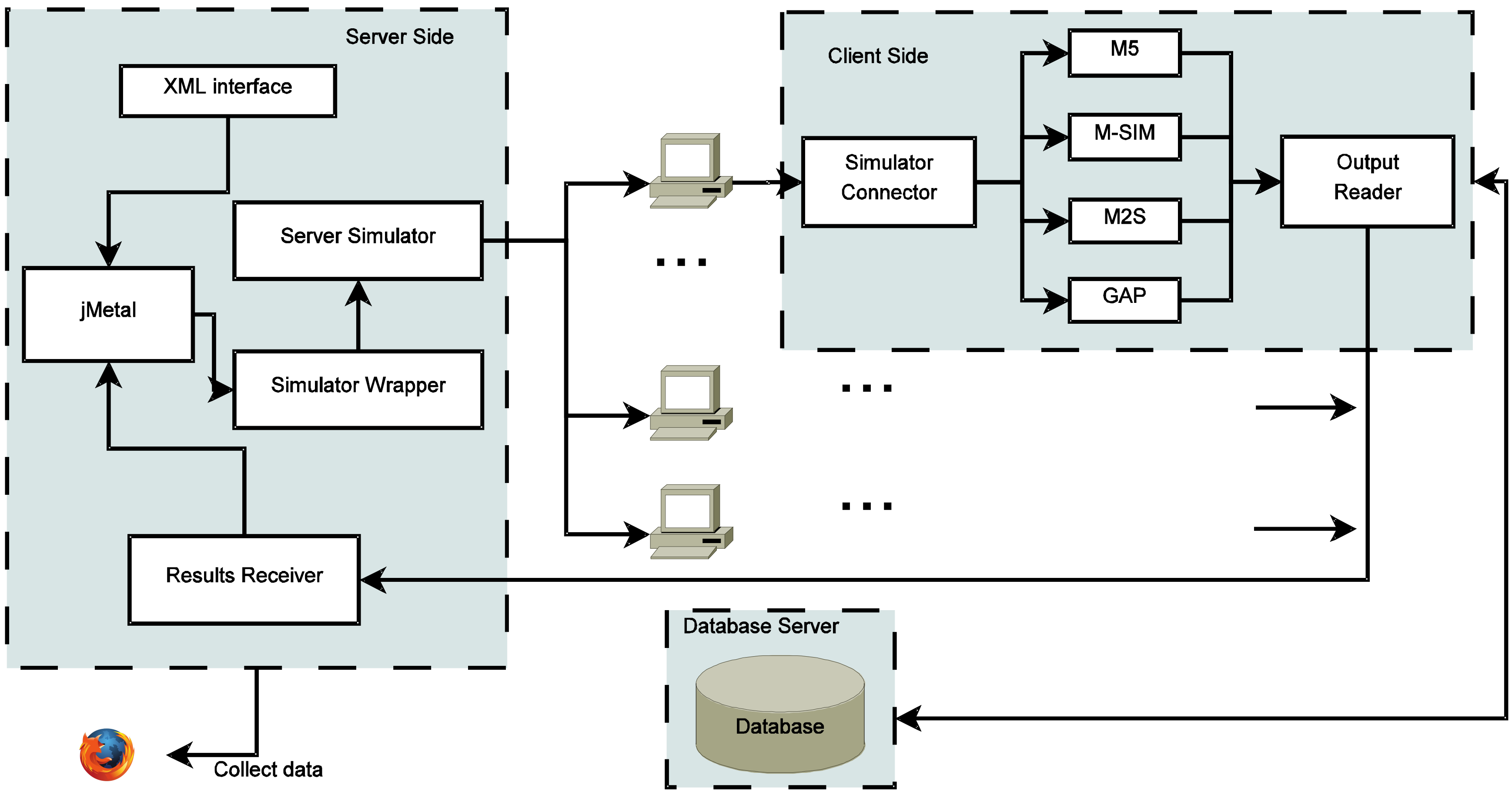


- Does not require the true Pareto front to be known
- Volume enclosed by:
  - the current Pareto front approximation and
  - the hypervolume reference point

FADSE

- Integrates many DSE algorithms (through jMetal library)
- Can connect to many simulators: M5, MSIM2, MSIM3, Multi2Sim, GAP, GAPtimize, UniMap
- Accelerates DSE process:
  - Parallel evaluation of individuals
  - Results reuse
- Runs on commodity LAN, HPC systems
- Reliable
- Introduction of domain knowledge
  - Constraints
  - Fuzzy rules
  - Linked parameters

FADSE

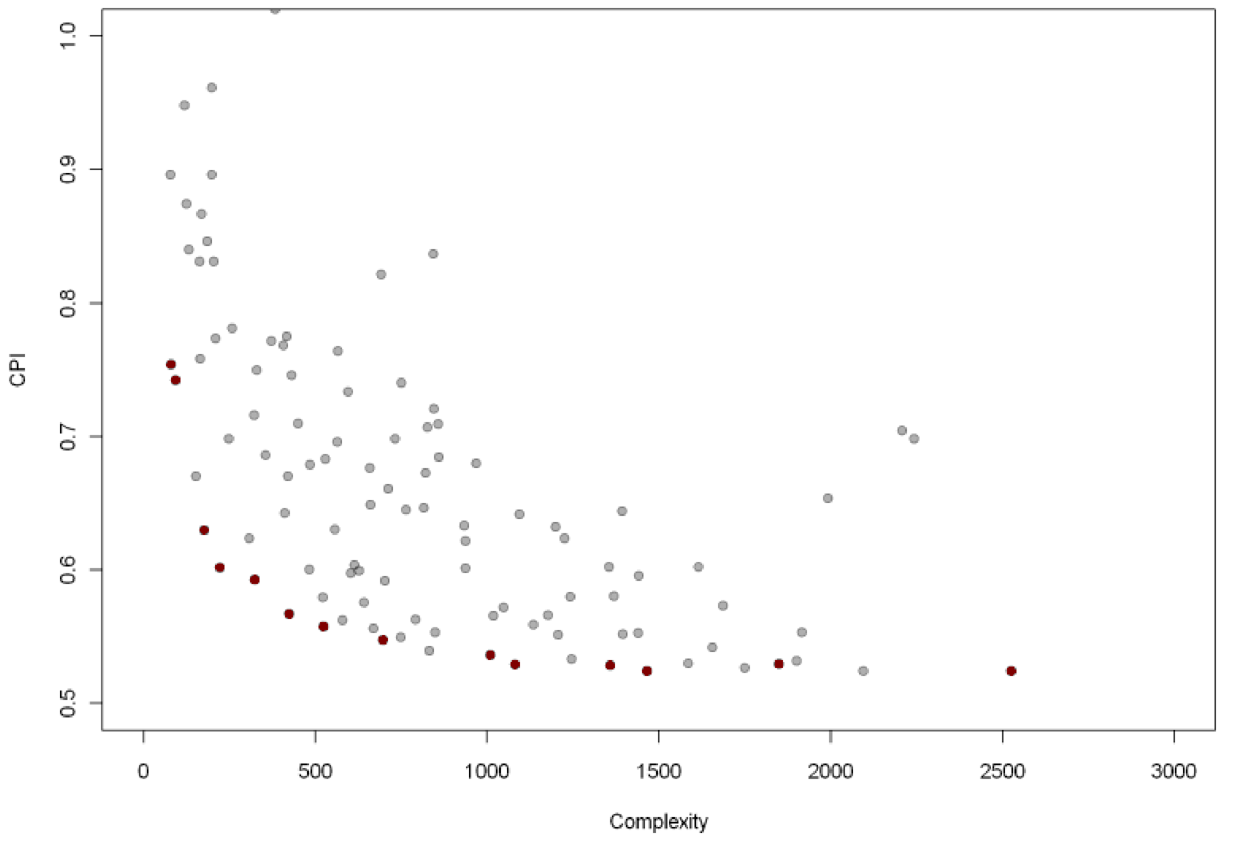


Server Side
XML interface
Server Simulator
jMetal
Simulator Wrapper
Results Receiver
Collect data
. . .
Client Side
M5
M-SIM
M2S
GAP
Simulator Connector
Output Reader
. . .
. . .
Database Server
Database

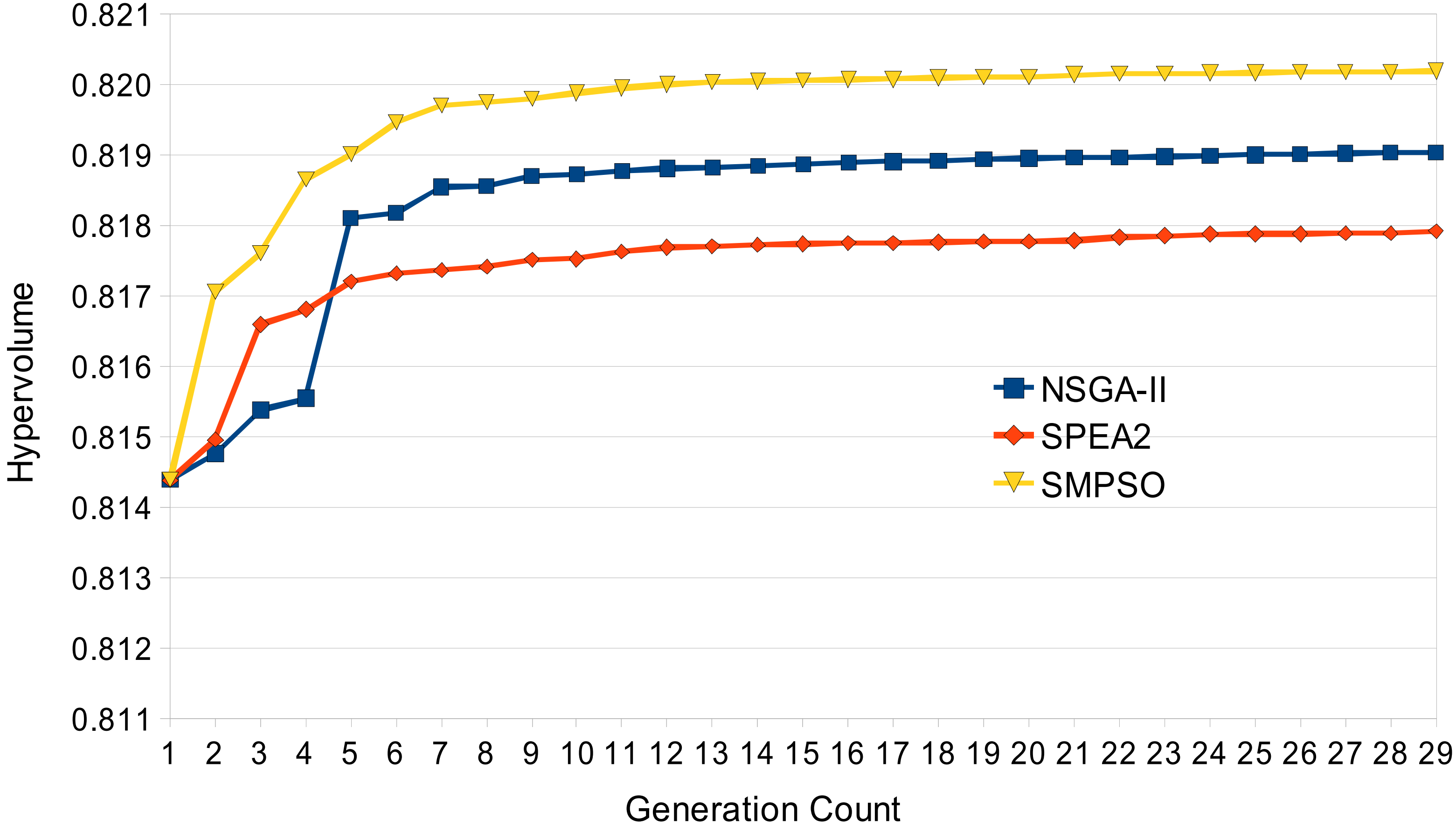
GAP and GAPtimize

- Grid Alu Processor (GAP)
  - Novel processor architecture from the University of Augsburg, combines coarse-grained reconfigurable array of functional units with superscalar-like frontend
  - Design space of over $\mathbf{1.1*10^6}$
- GAPtimize: Code optimizer for the GAP
  - Runs feedback-directed post-link optimizations
  - Supports whole program optimizations, here: inlining
  - GAP+GAPtimize: design space over $\mathbf{1.6*10^{13}}$
- Two objectives to be minimized: speed (CPI) and complexity (Jahr et al.(2011))

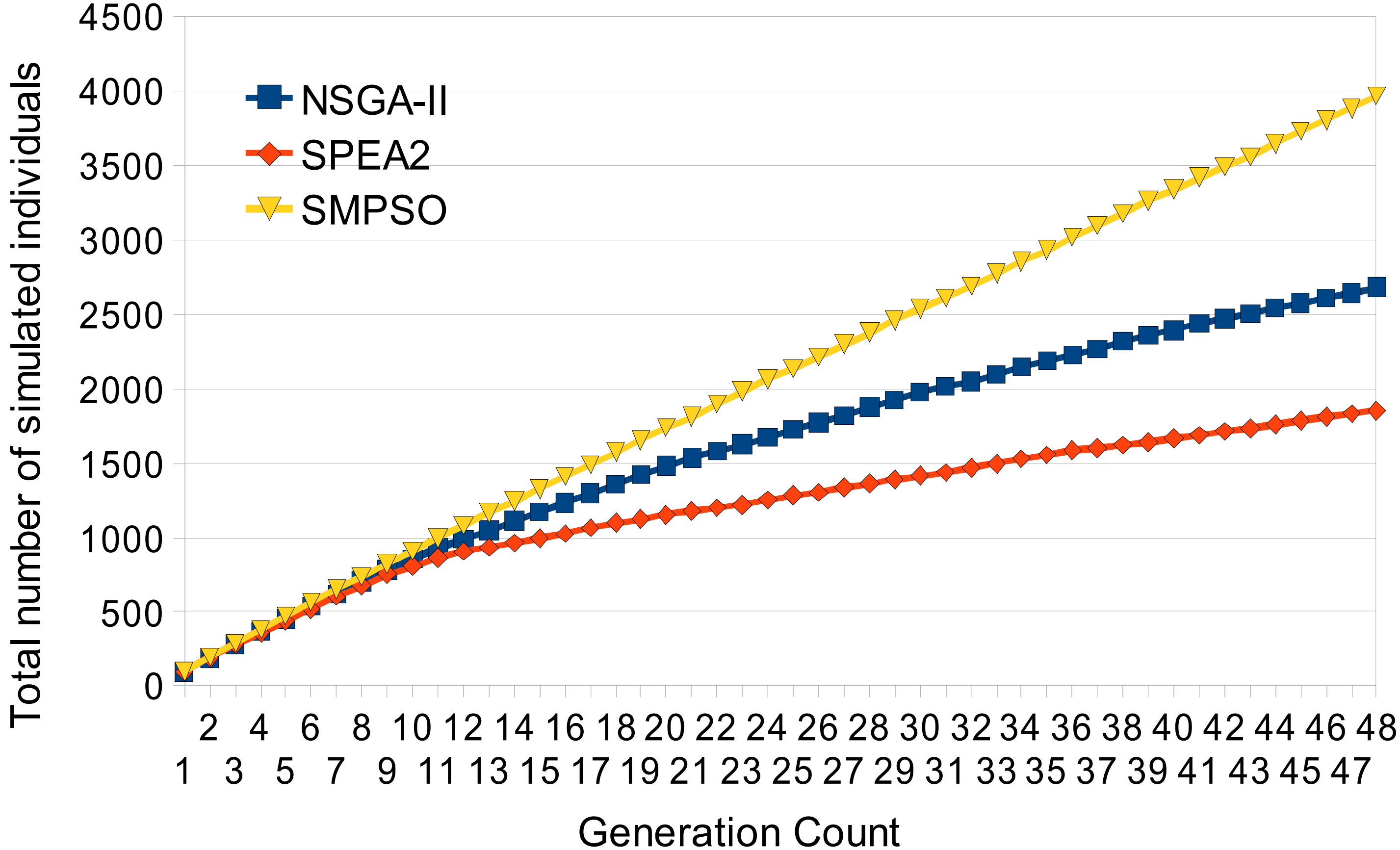
Pareto front approximation over the generations



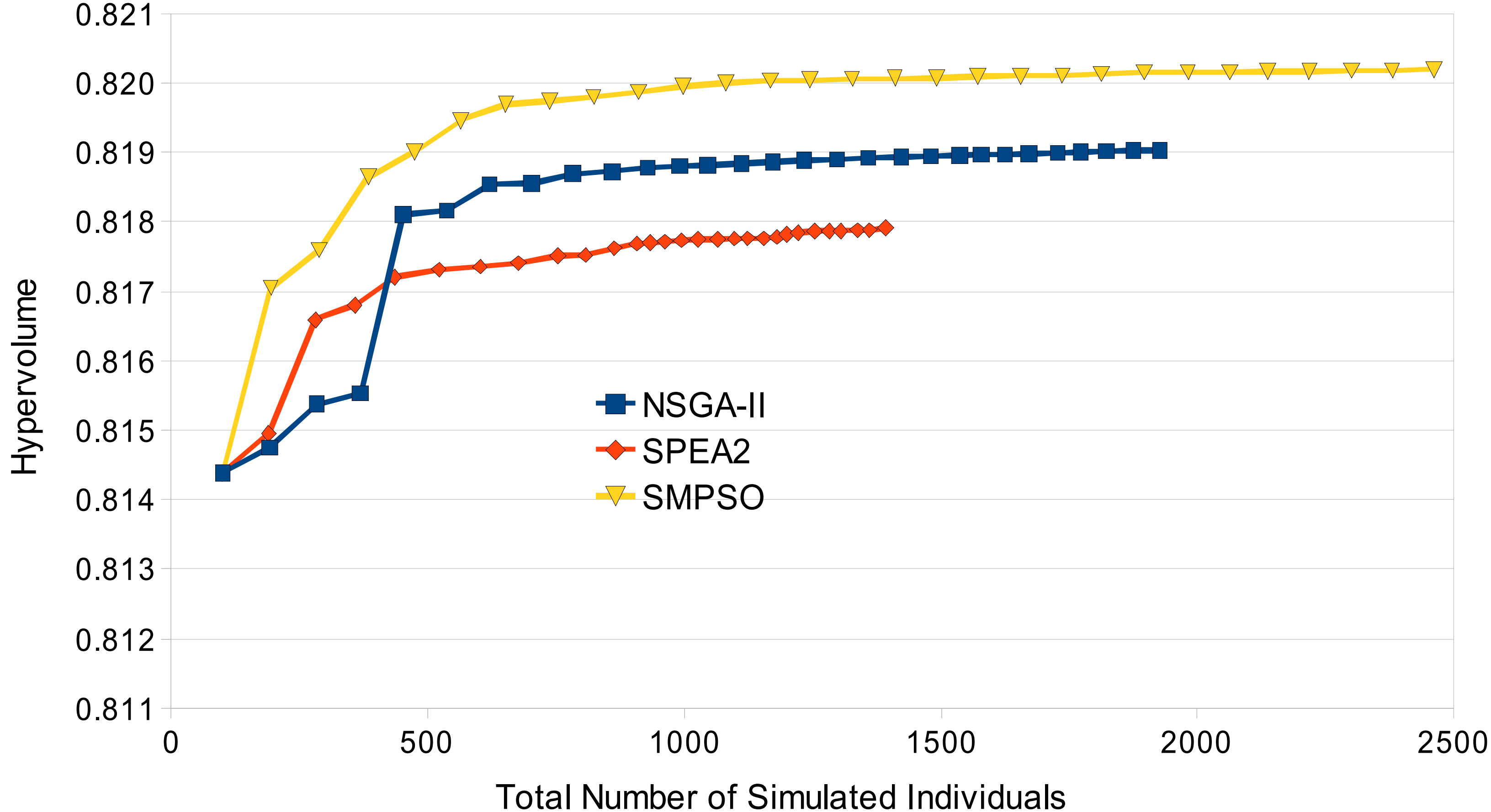
Hypervolume on GAP



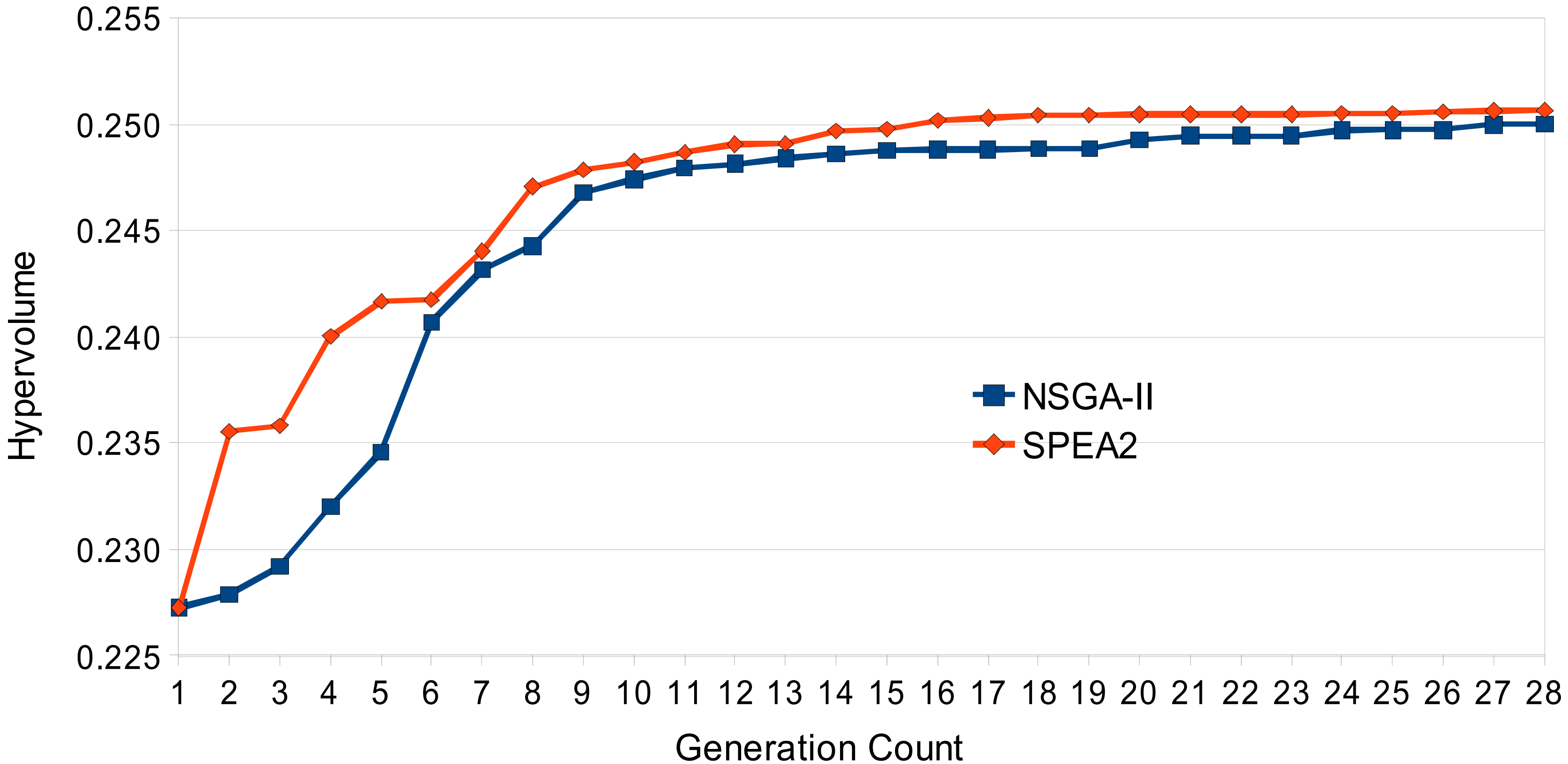
Number of simulated individuals



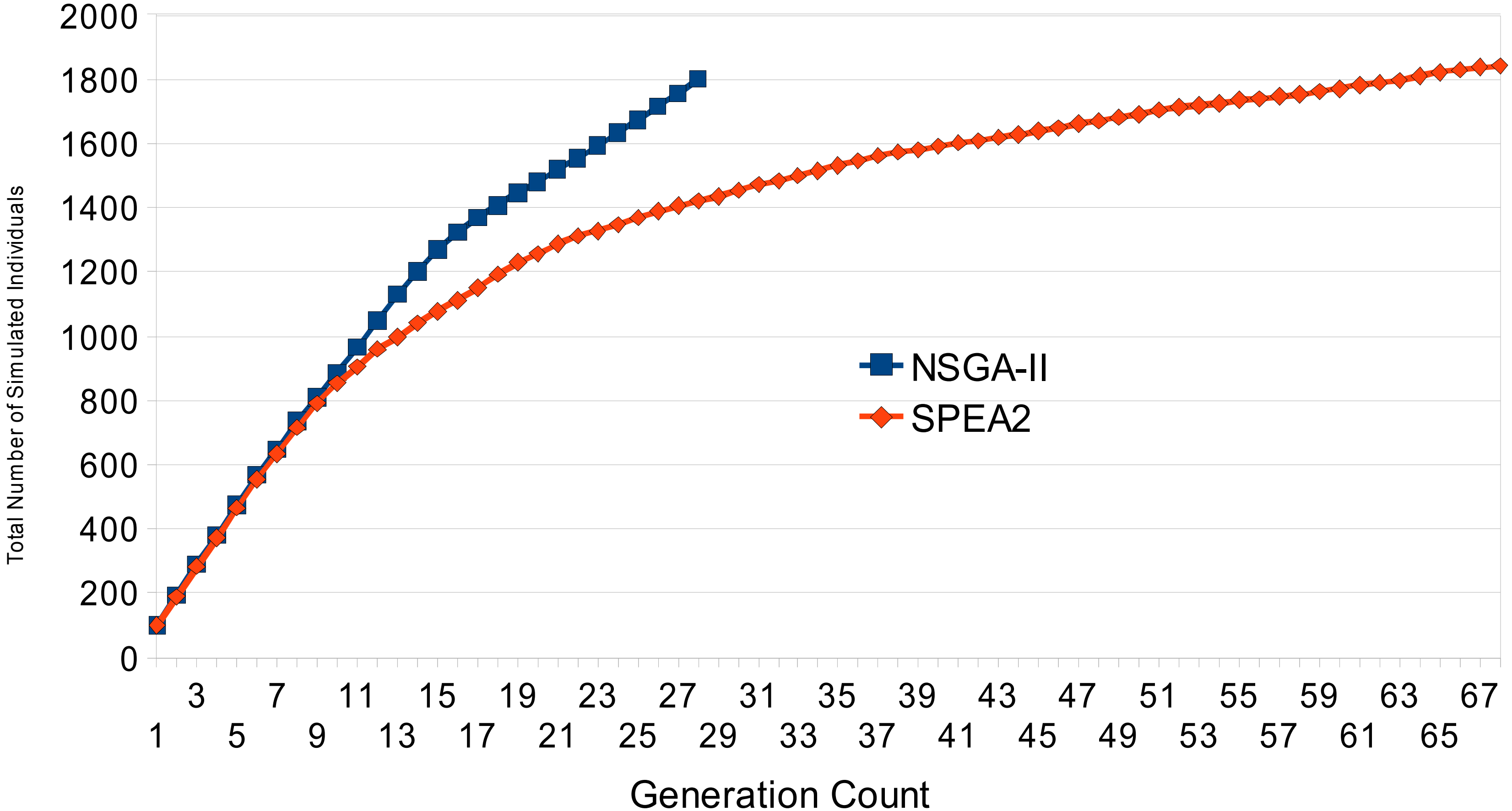
Hypervolume VS number of simulated individuals



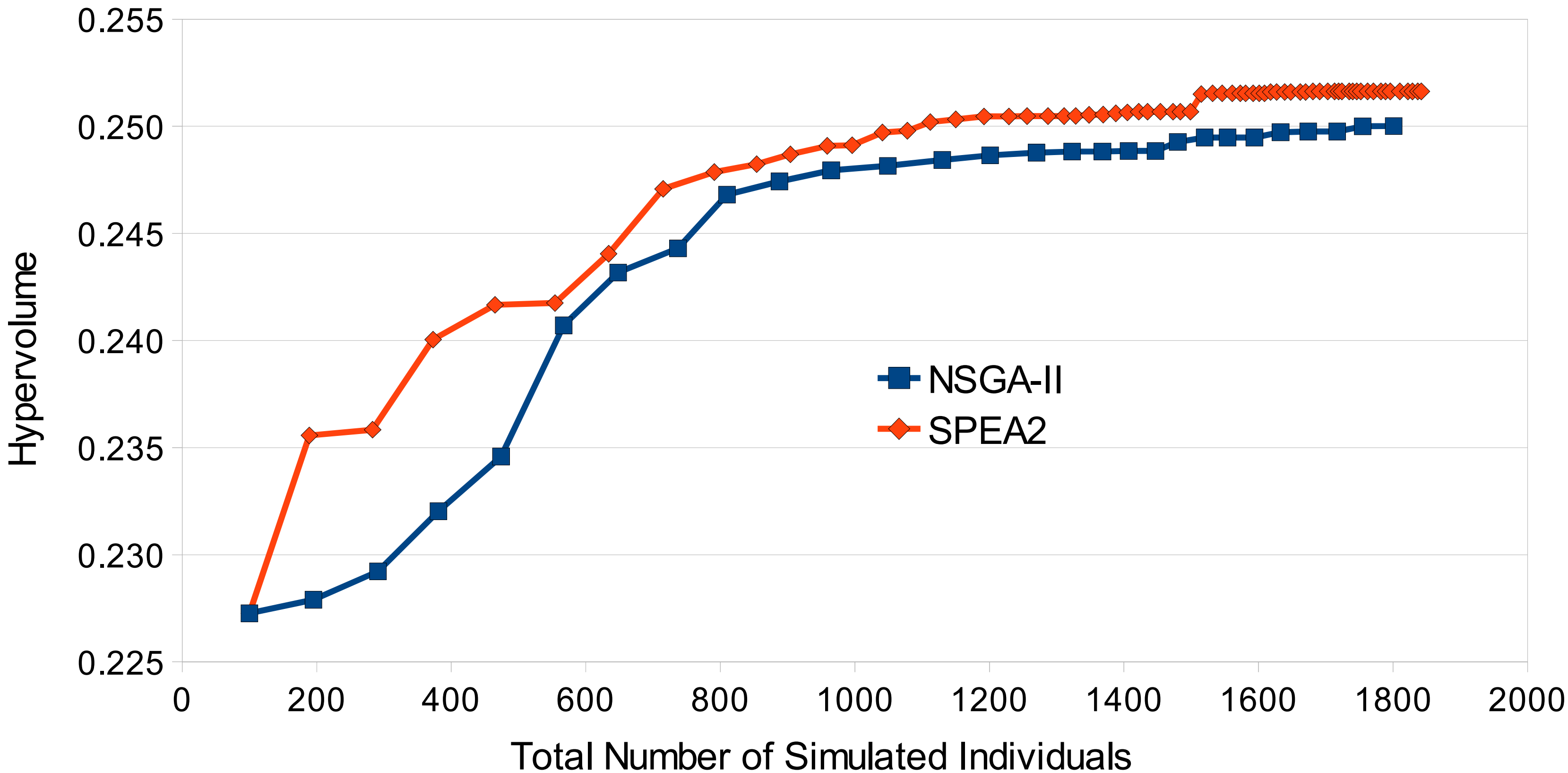
Hypervolume comparison NSGA-II, SPEA2



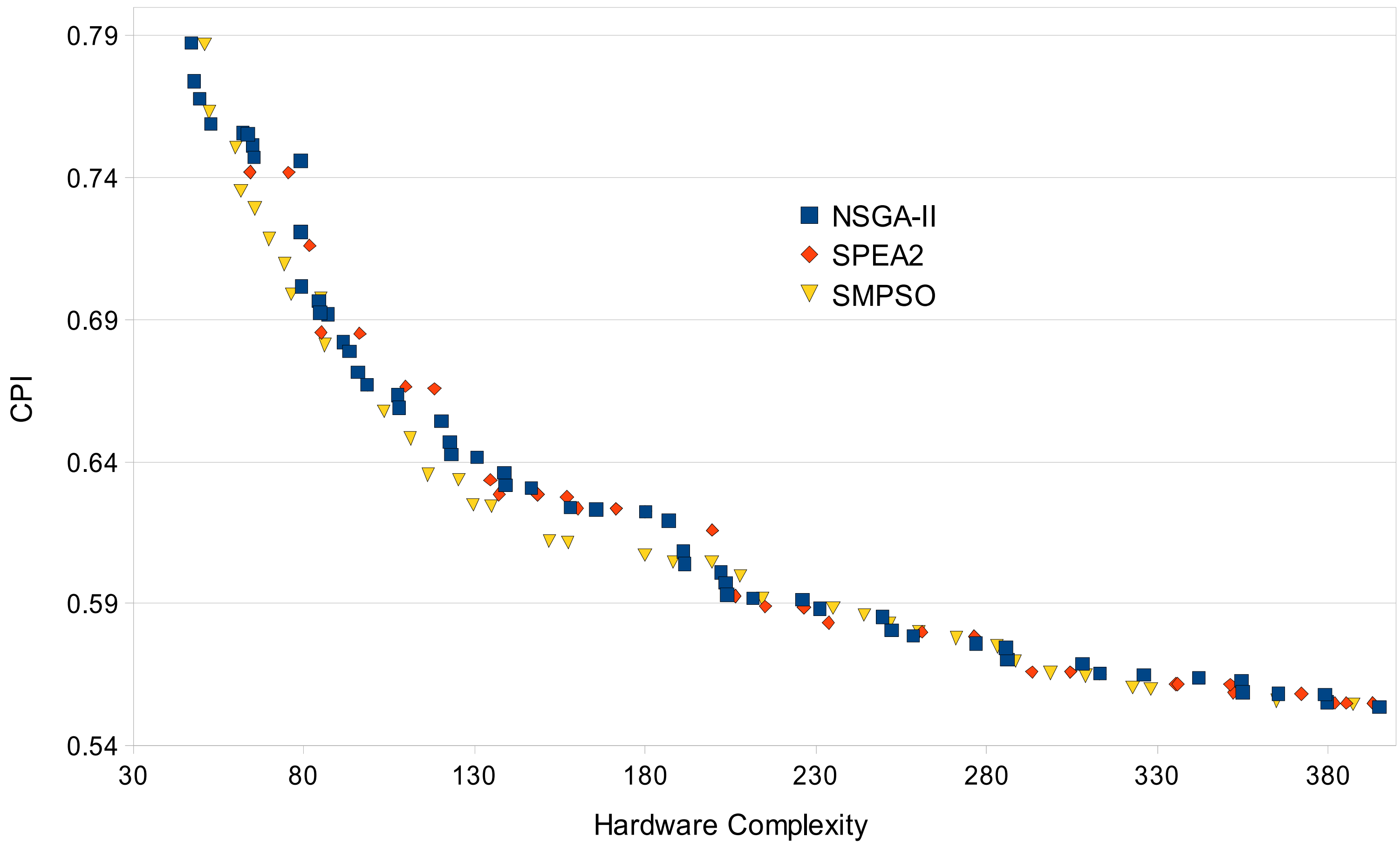
Number of simulations



Hypervolume VS total number of simulations



Pareto fronts approximation comparison



Conclusions

- SMPSO converges faster. No clear winner between NSGA-II and SPEA2
- After some generations (depending on the problem size) the algorithms tend to have similar results
- FADSE is able to find good results in large design spaces

Further work

- Insert known good configurations at the beginning of the search
- Domain knowledge using fuzzy rules, constraints (“hard” rules), “soft” rules
- Compare other algorithms on different simulators

Thank you

Questions