Automatic, Model-Based Software Performance Improvement for Component-Based Software Designs

Anne Martens, Heiko Koziolbek
FESCA 2009
Current Software Performance Engineering

Try a change

Prediction Response in 5 sec

Requirement Response in 3 sec

Try another one

Prediction Response in 5.2 sec

Requirement Response in 3 sec

What to change?
Automated Improvement

1. Generate new candidate
2. Evaluate candidate
3. Search the design space
Overview

Motivation

Design Options

Search Process

Current Prototype

Summary
Degrees of Freedom

- Considered design options
  - No functional impact
  - Potential extra-functional impact

- Degrees of freedom encoded in architectural model
  - Resource environment
    - Replication, sizing, hardware choice
  - Allocation
  - Configuration as feature configs
    - Of communication, middleware, app servers,...
      (whatever is available in the models)
  - Selection of components

- Design options are instantiations for a particular system
Design Options form the Design Space

Designs options are dimensions.

Many more choices in resource environment alone.

Response Time in sec

Sun Glassfish 1.0
IBM Websphere 2.0
JBoss 3.0

Speed of CPU 1

Application Server
Search the Space

- No full search
  - Performance evaluation costly
  - Space too large

- Pragmatic approach: Good solution is enough

- Use metaheuristics: Intelligently search
- Use performance domain knowledge
**PEROPTeryx Prototype**

- Starting with a given PCM instance
- Limited number of operators:
  - Increase processing speed of most utilised (>70%) resource
  - Exchange functionally equivalent components
- Steepest ascent hill climbing
- Mean value of response time distribution
- Stops if
  - Requirements are met or
  - All “neighbours” are worse
- Gets stuck in local optima
Current and Future Work

- Good representation of architectural decisions in the genome
- Multicriteria optimisation
- Use of more domain knowledge
- Learning during the search
- Constraints and requirements
### Problem
- No support for software architects to improve their design based on prediction results

### Idea
- Exploit explicit architecture model and components
- Automatically improve the design by applying metaheuristics
- Use performance domain knowledge to direct this search

### Benefit
- Faster improvement of the design by automation
- Insight for software architects
- Improves applicability of performance prediction approaches