ANGEL: A Hierarchical Approach to Online Auto-Tuning

Ray S. Chen
Jeffrey K. Hollingsworth
Motivation

• HPC systems will require online auto-tuning
  – Managing billion-way parallelism is non-trivial
• Cannot myopically focus on wall-time
  – 20MW power goal represents additional hurdle
• Need an auto-tuner that is:
  – Coordinated (Managed by the runtime OS)
  – Online (Optimization occurs without training runs)
  – Multi-objective (Handle power as well as wall-time)
Dealing with Multiple Objectives

- Multi-objective problems have a set of solutions
  - Each solution in set is equivalent
- Optimal solution is subjective
  - Tuner cannot choose for the user
- Online tuning even harder
  - Cannot pause for user input
  - Must limit overhead of testing
  - Use as few evaluations as possible
ANGEL Inputs

- Two values per objective collected from user apriori
  - Priority Rank
    - Orders each objective from highest to lowest
    - Each rank must be unique
  - Leeway Percentage
    - Amount ANGEL may stray from this objective’s best
    - Used to find improvements in other objectives
ANGEL Algorithm

• Begin with highest priority objective
  – Use single-objective algorithm for this objective alone
  – Record all value ranges (min, max) during sub-search
  – Repeat with next highest objective until all are searched

• Penalize sub-searches to maintain leeway preference
  – Applied when higher priority objective exceeds leeway
  – Allows upper level sub-searches to guide lower levels

• Result of final sub-search is the overall solution
ANGEL Penalty Function

• One-dimensional example with two objectives
Numerical Testsuite Experiments

• Tests from multi-objective optimization literature
  – Designed to be difficult, but not pathological

• Compared against ParEGO
  – Represents best evolutionary algorithm for our case
  – Strives to use very few function evaluations
  – Geared towards (relatively) low-dimensional objectives

• Compared against random
  – Must ensure our algorithm does something intelligent
Testsuite Results – Quality

• Quality is a measure of the converged solution.
  – Distance from the best solution discovered by hand.

• ANGEL wins on two-thirds of testsuite.

Converged Distance from Optimal (Normalized)
Testsuite Results – Efficiency

- Efficiency is a measure of search overhead.
  - Critically important to keep low for online auto-tuning.
- ANGEL wins on all but one test.

Distance from Optimal per Evaluation (Normalized)

- KNO1
- OKA1
- OKA2
- VLMOP2
- VLMOP3
- DTLZ1a
- DTLZ2a
- DTLZ4a
- DTLZ7a
LULESH Experiments

• Lawrence Livermore’s LULESH proxy application
  – Unstructured hex mesh problem

• Tuning two input variables:
  – OpenACC loop vector length
  – GPU clock frequency

• Two objectives:
  – Minimize running time
  – Minimize energy consumption
LULESH Objective Landscapes

Energy Search Space

Runtime Search Space
Changing the Threshold

- ANGEL behaves properly for changing leeways
  - Energy usage declines along with leeway
  - Shows proper behavior for real HPC data
Conclusion and Future Work

• ANGEL is a step towards runtime system auto-tuning
  – Uses an iterative and hierarchical approach
  – Controlled by simple user inputs provided apriori
  – Performs well on numerical testsuite
  – Shown to work correctly on real HPC data

• Future work
  – Power (rather than energy) studies
  – Alternate underlying single-objective algorithms
  – Explore avenues for parallelism