Soft Performance Analysis for Parallel and Distributed Programs

Hong-Linh Truong, Thomas Fahringer
Distributed and Parallel Systems Group
Institute of Computer Science
University of Innsbruck

{truong,tf}@dps.uibk.ac.at
http://dps.uibk.ac.at/projects/pma
Talk Outline

- Motivation
- Outline of soft performance analysis approach
- Performance score and similarity measure
- Some soft analysis techniques
- Conclusion and future work
Motivation

- Lack of the specification and control of inexact parameters, commands and requests in existing performance analysis tools.
- Performance tools do not interact with the user through high-level notation (e.g., words).
- Graphics techniques are very useful, but not suitable for performance analysis of large-scale and complex applications.

Our approach: apply soft computing, similarity measure, machine learning in performance analysis.

Picture taken from a talk of D. Kranzlmüller (Uni. Linz)
Simple Example: Soft vs Hard Analysis

- **Hard computing**
  - Apply exact methods
  - Binary logic, crisp system, numerical analysis
  
  If \( \frac{T_{comm}}{T_{comp}} > 0.7 \) then r have high communication to computation ratio

- **Soft Computing**
  - Support imprecision and uncertainty
  - Computing with words
  
  If (\( T_{comm}/T_{comp} \) is high) then r have high communication to computation ratio with x degree
Existing works

- Fuzzy logic for performance monitoring, e.g. performance contracts (Pablo)

- Using classification techniques based on machine learning, multivariate statistical techniques (e.g., done by Vetter and colleagues)

- APART performance property characterizes specific negative performance behavior of code regions

- Recent work applying data clustering in TAU (Uni. Oregon, to appear in SC05)

  - Fuzzy logic has not been exploited in data analysis techniques, e.g., performance classification

  - Not interact with the end user through high-level notation, e.g. linguistic query
Outline of Soft Performance Analysis

Approach

- Performance values are mapped into performance scores
- Performance characteristic terms are represented by a fuzzy set
  - A set of perf. characteristic terms describes possibilities of a metric
- To analyze the performance and interpret performance results with linguistic terms
- Similarity theory and machine learning: similarities and differences among performance data items
- Focuses of this talk
  - Conceptual framework: How can we apply soft computing into performance analysis
  - Interaction between performance tools and the user: Through high level notions and concepts expressed in linguistic expressions
  - Potential applications of soft performance analysis
Preliminaries

- Performance data
  - A program consists of a set of (instrumented) code regions
  - Each code region is measured with a set of \( n \) metrics

- Performance experiment data used obtained from
  - 3DPIC, an MPI program, simulates the interaction of high intensity ultrashort laser pulses with plasma in three dimensional geometry
  - LAPW0 calculates the effective potential of the Kohn-Sham Eigen-Value problem, implemented in Fortran MPI
  - Stommel, OpenMP/MPI program, solves the 2d Stommel Model of Ocean Circulation using a Five-point stencil and Jacobi iteration.
Performance Score

Performance score concept

- Map a value of metric $m$, $v$, into $[0, 1]$. Performance score, $s$, of $v$ is defined by
  
  $$ s = \mathcal{O}(v), \quad \mathcal{O}(v) : [0, V] \mapsto [0, 1] $$

- $\mathcal{O}(v)$ is the membership function, $V$ is the maximum value of $m$ obtained from the base.

- Each code region is represented by a vector of scores

- Overall weighted average (OWA) for performance scores

  $$ \text{OWA}(\vec{s}) = \frac{\sum_{i=1}^{n} (s_i \cdot w_i)}{\sum_{i=1}^{n} w_i} $$
Performance Score (cont.)

- The base is dependent on the scope of the analysis
  - Analysis can be done within a code region, a thread or the entire program

- \([0,1]\): 0 means lowest score, 1 means highest score
  - Semantics is defined by specific implementations

- Membership functions are also analysis-dependent
  - Examples: linear, S-function, etc.

- Performance score concept allows to normalize performance metrics but considering
  - The dynamics and flexibility
  - The uncertainty and imprecision

- Used in dynamic tuning, ranking, clustering, etc.
Ranking Analysis

- Widely used in distinguishing significant and insignificant components
  - Which child code regions of a code region have strong impact on the performance of the parent?

- Ranking based on raw measurement value is difficult to interpret and compare the significance of the performance

<table>
<thead>
<tr>
<th>Performance Rank based on</th>
<th>Performance Rank based on Fuzzy</th>
</tr>
</thead>
<tbody>
<tr>
<td>File View Code Region Analysis</td>
<td>File View Code Region Analysis</td>
</tr>
<tr>
<td>Experiment</td>
<td>Thread 0</td>
</tr>
<tr>
<td>gsr403</td>
<td>Region 1:RELATIVISTIC_PLASMA_PROPAGATION[CR_P:1.186][rank=1.0]</td>
</tr>
<tr>
<td>Process 0</td>
<td>Region 57:MPI_SEND[CR_MPISEND:1770:1772][rank=0.07648119182319221]</td>
</tr>
<tr>
<td></td>
<td>Region 69:MPI_SEND[CR_MPISEND:1904:1908][rank=0.06450781781946]</td>
</tr>
<tr>
<td></td>
<td>Region 8:MPI_SEND[CR_MPISEND:1077:1079][rank=0.04915781679412661]</td>
</tr>
<tr>
<td></td>
<td>Region 26:SR_E_FIELD[CR_A:0.653.723][rank=0.04736557687277515]</td>
</tr>
<tr>
<td></td>
<td>Region 7:MAIN[CR_S:173.183][rank=0.999993531881636]</td>
</tr>
<tr>
<td></td>
<td>Region 48:CAL_POWER[CR_A:2244:2323][rank=0.00196]</td>
</tr>
<tr>
<td></td>
<td>Region 13:MPI_RECV[CR_MPI_RECV:1132:1134][rank=6.0]</td>
</tr>
<tr>
<td></td>
<td>Region 52:MPI_RECV[CR_MPI_RECV:1714:1716][rank=6.0]</td>
</tr>
<tr>
<td></td>
<td>Region 15:MPI_RECV[CR_MPI_RECV:1155:1157][rank=5.0]</td>
</tr>
<tr>
<td></td>
<td>Region 64:MPI_RECV[CR_MPI_RECV:1846:1848][rank=4.0]</td>
</tr>
<tr>
<td></td>
<td>Region 17:MPI_RECV[CR_MPI_RECV:1179:1180][rank=1.0]</td>
</tr>
</tbody>
</table>
Fuzzy-based Performance Classification

1. Define a set of *performance characteristic terms* $T$ for a given metric

   $$T = \{t_1, t_2, ..., t_n\}$$

2. A term is represented by a fuzzy set

3. Performance data are classified according to terms
Fuzzy-based Performance Classification (cont.)

<table>
<thead>
<tr>
<th>Code Region</th>
<th>Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Region 18: PARTICLE_LOAD [CR_A:191:328]</td>
<td>high (degree = 0.937) /</td>
</tr>
<tr>
<td>Region 26: SR_E_FIELD [CR_A:653:723]</td>
<td>medium (degree = 1) /</td>
</tr>
<tr>
<td>Region 47: SET_FIELD_PAR_BACK [CR_A:1794:1928]</td>
<td>medium (degree = 0.549) /</td>
</tr>
<tr>
<td>Region 48: CAL_POWER [CR_A:2244:2323]</td>
<td>low (degree = 0.007) / medium (degree = 0.28) /</td>
</tr>
</tbody>
</table>
Fuzzy-based Performance Search

- Existing performance tools
  - Do not offer the possibility of search performance data with linguistic query

- PERFORMANCE Query Language based on fuzzy logic (PERFQL)
  - Performance search based on linguistic expressions
  - Easily to define/understand queries

```plaintext
<PERFQL_Statement> ::= <PERFQL_Expr> | <PERFQL_Statement>
    OR <PERFQL_Expr>

<PERFQL_Expr> ::= <PERFQL_Term> | <PERFQL_Expr>
    AND <PERFQL_Term>

<PERFQL_Term> ::= (<METRIC_Expr> is <F_Expr>)
```

- Metric or Metric Expression
  - `wtime`
  - `L2_TCM/L2_TCA`
  - `odata_send/wtime`

- Fuzzy Expression
  - `HIGH_EXECUTION_TIME`
  - `very HIGH_EXECUTION_TIME`
  - `slightly POOR_SEND_OVERHEAD`
Fuzzy-based Performance Search (cont.)

Assume any code region takes more than 20% total execution is `HIGH_EXECUTION_TIME`

New query with cache misses condition
**Fuzzy Approach to Bottleneck Search**

1. Using fuzzy sets to represent *bottleneck conditions*

2. Using fuzzy sets to represent *negligible bottlenecks*

- **Search results**
  - Indicate the *degree of bottleneck*
    - We can use the degree of bottleneck for further tasks
  - Locate *negligible bottlenecks*
    - We may not find any bottlenecks because the condition is not exact
Bottleneck Search: Simple Example

- Search for low, medium and high degree of bottleneck

- Search also negligible bottlenecks
Performance Similarity Measure

- Problems:
  - Difficult to observe and perceive the performance similarity and difference through complex visualization
  - Performance similarity measure indicates the performance similarity among code regions and among experiment factors

\[ \text{sim}(o_i, o_j) \to [0, 1] \]

- 0 denotes complete dissimilarity and 1 denotes complete similarity
Performance Similarity Measure

- Performance similarity measure for code regions
  1. Using performance score concept to determine performance scores of region summaries \( r_{si} \) and \( r_{sj} \). Each \( rs \) is represented as a vector of \( n \) performance scores
  2. Determining distance measure between \( r_{si} \) and \( r_{sj} \). For example,

\[
d_{ij} = \sqrt{\sum_{l=1}^{n} (s_{il} - s_{jl})^2}
\]

3. Determining performance similarity between two code regions

\[
sim_{ij} (r_{si}, r_{sj}) = 1 - d_{ij}
\]
Performance Similarity Analysis (cont.)

- **Stommel:**
  - Similarity measure for cache accesses of Stommel application

```
<table>
<thead>
<tr>
<th>ProcessingUnit</th>
<th>gsr415-&gt;0-&gt;0</th>
<th>gsr415-&gt;0-&gt;1</th>
<th>gsr415-&gt;0-&gt;2</th>
<th>gsr415-&gt;0-&gt;3</th>
<th>gsr411-&gt;1-&gt;0</th>
<th>gsr411-&gt;1-&gt;1</th>
<th>gsr411-&gt;1-&gt;2</th>
<th>gsr411-&gt;1-&gt;3</th>
</tr>
</thead>
<tbody>
<tr>
<td>gsr415-&gt;0-&gt;0</td>
<td>1</td>
<td>0.944</td>
<td>0.659</td>
<td>0.659</td>
<td>0.893</td>
<td>0.893</td>
<td>0.659</td>
<td>0.659</td>
</tr>
<tr>
<td>gsr415-&gt;0-&gt;1</td>
<td>0.944</td>
<td>1</td>
<td>0.672</td>
<td>0.672</td>
<td>0.837</td>
<td>0.949</td>
<td>0.672</td>
<td>0.672</td>
</tr>
<tr>
<td>gsr415-&gt;0-&gt;2</td>
<td>0.659</td>
<td>0.672</td>
<td>1</td>
<td>1</td>
<td>0.602</td>
<td>0.683</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>gsr415-&gt;0-&gt;3</td>
<td>0.659</td>
<td>0.672</td>
<td>1</td>
<td>1</td>
<td>0.602</td>
<td>0.683</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>gsr411-&gt;1-&gt;0</td>
<td>0.893</td>
<td>0.837</td>
<td>0.602</td>
<td>0.602</td>
<td>1</td>
<td>0.766</td>
<td>0.602</td>
<td>0.602</td>
</tr>
<tr>
<td>gsr411-&gt;1-&gt;1</td>
<td>0.893</td>
<td>0.949</td>
<td>0.683</td>
<td>0.683</td>
<td>0.766</td>
<td>1</td>
<td>0.683</td>
<td>0.683</td>
</tr>
<tr>
<td>gsr411-&gt;1-&gt;2</td>
<td>0.659</td>
<td>0.672</td>
<td>1</td>
<td>1</td>
<td>0.602</td>
<td>0.683</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>gsr411-&gt;1-&gt;3</td>
<td>0.659</td>
<td>0.672</td>
<td>1</td>
<td>1</td>
<td>0.602</td>
<td>0.683</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
```

- **LAPW0:**
  - Similarity measure based on wallclock time

```
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Region 2:CA_MULTIPOLMENTS[CR_A.256.508]</td>
<td>1</td>
<td>0.996</td>
<td>0.638</td>
<td>0.635</td>
<td>0.625</td>
<td>0.625</td>
</tr>
<tr>
<td>Region 3:CA_COULOMB_INTERSTITIAL_POTENTIAL[CR_A.536.565]</td>
<td>1</td>
<td>0.986</td>
<td>0.629</td>
<td>0.636</td>
<td>0.597</td>
<td>0.597</td>
</tr>
<tr>
<td>Region 4:CAL_COULOMBE_RMT[CR_A.635.668]</td>
<td>1</td>
<td>0.999</td>
<td>0.63</td>
<td>0.631</td>
<td>0.597</td>
<td>0.597</td>
</tr>
<tr>
<td>Region 5:CAL_CP_INSIDE_SPHERES[CR_A.678.772]</td>
<td>1</td>
<td>0.962</td>
<td>0.632</td>
<td>0.639</td>
<td>0.598</td>
<td>0.598</td>
</tr>
<tr>
<td>Region 6:FFT_REAL0[CR_OTHERSEQ.881.883]</td>
<td>1</td>
<td>0.997</td>
<td>1</td>
<td>0.997</td>
<td>0.981</td>
<td>0.981</td>
</tr>
<tr>
<td>Region 7:FFT_REAL3[CR_OTHERSEQ.889.991]</td>
<td>1</td>
<td>0.999</td>
<td>1</td>
<td>1</td>
<td>0.536</td>
<td>0.756</td>
</tr>
<tr>
<td>Region 9:FFT_REAL4_CR[CR_OTHERSEQ.915.917]</td>
<td>1</td>
<td>1</td>
<td>0.492</td>
<td>0.479</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Performance Similarity Analysis (cont.)

- Performance similarity measure for experiment factors
  Given a set of controllable factors $F = \{f_1, f_2, \ldots, f_n\}$ and given experiments $e_i$ and $e_j$
  1. Factor $f$ is described by a membership function
  2. Determine similarity measure between $f$ of $e_i$ and $e_j$, $\text{sim}_f (e_i, e_j)$
  3. Analyze relations among similarity measures for code regions and experiment factors

LAPW0

<table>
<thead>
<tr>
<th>Factor</th>
<th>Fuzzy Set</th>
<th>Range</th>
<th>Factor Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>atoms</td>
<td>linear</td>
<td>[0,0.72]</td>
<td>problem size</td>
</tr>
<tr>
<td>CPU</td>
<td>S-function</td>
<td>[0,0.64]</td>
<td>machine</td>
</tr>
<tr>
<td>network</td>
<td>S-function</td>
<td>[0,158.20]</td>
<td>communication</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{sim}_{\text{atoms}} ({\text{atoms}, 1})$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.5</td>
<td>0.5</td>
</tr>
<tr>
<td>$\text{sim}_{\text{CPU}} ({\text{CPU}, 1})$</td>
<td>1</td>
<td>1</td>
<td>0.9531</td>
<td>0.9531</td>
<td>0.9531</td>
<td>0.9531</td>
</tr>
<tr>
<td>$\text{sim}_{\text{network}} ({\text{network}, 1})$</td>
<td>1</td>
<td>0.1519</td>
<td>1</td>
<td>0.1519</td>
<td>1</td>
<td>0.1519</td>
</tr>
<tr>
<td>$\text{sim}_{\omega} ({\text{wtime}, 1})$</td>
<td>1</td>
<td>0.996</td>
<td>0.638</td>
<td>0.635</td>
<td>0.625</td>
<td>0.625</td>
</tr>
</tbody>
</table>

Similarity analysis for CA_MUTIPOLMENTS region
Fuzzy C-Means Clustering

- 3D PIC executed with 4 processes
Other potential applications of soft performance analysis techniques

- Decision making in dynamic performance tuning
  - Dynamic performance tuning tools: MATE (UAB), Active Harmony (J. Hollingsworth)
  - Automatically replacing components, selecting different implementations based on performance scores and performance similarity measures
- Performance data collection/reduction
  - Rules based on crisp-condition can be replaced by fuzzy rules based on performance scores
- etc.
Conclusion and Future Work

Contributions: we proposed the soft performance analysis approach

- Provide flexible, scalable techniques for analyzing and comparing the performance of parallel and distributed applications
- Interact with the user through high-level notation
- Aim to support automatic performance analysis

However, soft performance analysis is just at an early stage

- Not everything discussed has been fully implemented

What should be done next

- So far, we have just focused on conceptual framework, not on how to select membership and distance functions
  - Study the selection of membership and distance functions
- Apply soft performance analysis for dynamic performance tuning, autonomic computing
- Linguistic variables and fuzzy rules for SLAs (service level agreements) in the Grid