Data Mining Analysis to Validate Performance Tuning Practices for HPL

Tuan Zea Tan¹, Rick Siow Mong Goh¹, Verdi March²,³, Simion See³,⁴

¹A*STAR Institute of High Performance Computing (IHPC), Singapore
²Dept. of Computer Science, NUS, Singapore
³Dept. of Mechanical & Aerospace Engineering, NTU, Singapore
⁴APSTC, Sun Microsystems Inc., Singapore
Data Mining Analysis to Validate Performance Tuning Practices for HPL

Tuan Zea Tan¹, Rick Siow Mong Goh¹, Verdi March²,₄, Simon See³,₄

¹A*STAR Institute of High Performance Computing (IHPC), Singapore
²Dept. of Computer Science, NUS, Singapore
³Dept. of Mechanical & Aerospace Engineering, NTU, Singapore
⁴APSTC, Sun Microsystems Inc., Singapore
Outline

Introduction
Methodology
Experimental Results
Conclusion
• Benchmarking is a key aspect to assess the performance of a system
• High Performance Linpack (HPL): a case study
  – A benchmark application to evaluate the floating point performance of HPC systems by solving dense linear algebra equations
  – Used to rank systems in Top500
  – Tries to answer qns such as “which system should I buy?”
• Hardware & Software approaches to achieve performance
  – HW: System architects design the system to meet the resource demands of targeted applications [interconnect, I/O, employ accelerators (e.g. Roadrunner), etc]
  – SW: Tune application parameters that maximize performance/throughput. Flexible since able to conduct after system is operational. Our work is on this.
• Large number of application parameters
  – Large number of application configurations need to be tested to obtain optimum configuration
  – Large search space

• Testing each application configuration using production data set requires extensive amount of time on performance tuning:
  – Per experiment: >2 hrs for matrix dim 268k, 64 nodes, 26% of max memory
  – HPL has many performance-related parameters; will take a long time if all configurations are explored
• **Analytical performance models**
  
  • Derive relationship between parameters and performance based on complexity analysis performed by domain experts on algorithm implemented
  
  • Based on subset of benchmark parameters, else impractical to derive
  
  • Models may perform poorly if underlying assumptions are violated
  
  • Different models may have to be developed for different architectures
• Rules of Thumb
  – Or heuristic-based guidelines, are developed based on benchmarking experience over time across various system architectures

• Example of HPL guidelines [Bozzo-Rey et al., HPL website]
  – Select $N \approx 0.8 \sqrt{\text{total memory size} \times \frac{\text{sizeof(double)}}{8}}$
  – Use row-major mapping for MPI processes ($PMAP = 0$)
  – Tuning $NB$, $P$ and $Q$ is sufficient
Our Approach to Speed Up Performance Tuning

- Use Data Mining (DM)
- DM techniques model the relationship/association between application parameters and performance \textit{autonomously}
- DM models capable of predicting the performance of a set of parameters configuration after it has learnt the relationship between parameters and performance based on a set of training data
- Training data refers to benchmark results based on a number of sample configurations which is a subset of the whole parameter space
- Relationships found are represented as equations or association rules
Data Mining to Speed Up Performance Tuning

Application Parameters (e.g. HPL) → Form configurations: Subset of design space → Use Data Mining to learn/train Predictive Model → Predict performance (e.g. FLOPs) of new configurations

DM models are capable of predicting the performance of a set of parameters configuration after it has learnt the relationship between parameters and performance based on a set of training data.
Objectives / Outcome of our work

- To demonstrate the accuracy of data mining (DM) as compared to the analytical approach
- To present that the proposed DM approach can be used to validate different rules of thumb
- To propose a DM approach for speeding up performance tuning by predicting *relative performance* (instead of absolute perf)
Outline

Introduction

Methodology

Experimental Results

Conclusion
High Performance Linpack: our case study

### Parameters of HPL

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Range</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N$</td>
<td>89,600-268,800</td>
<td>Problem size (i.e., matrix size = $N^2$)</td>
</tr>
<tr>
<td>$NB$</td>
<td>32, 40, 50, 64, 80, 100, 256</td>
<td>Block size (matrix block size)</td>
</tr>
<tr>
<td>$P \times Q$</td>
<td>$8 \times 32, 16 \times 16$, $16 \times 64, 32 \times 8$, $32 \times 32, 64 \times 16$</td>
<td>The processor topology in two dimensions</td>
</tr>
<tr>
<td>$BCAST$</td>
<td>0-5</td>
<td>Broadcast algorithm specified</td>
</tr>
<tr>
<td>$PFACT$</td>
<td>0-2</td>
<td>Panel factorization</td>
</tr>
<tr>
<td>$RFACT$</td>
<td>0-2</td>
<td>Recursive factorization</td>
</tr>
<tr>
<td>$PMAP$</td>
<td>0-1</td>
<td>MPI process mapping</td>
</tr>
<tr>
<td>$SWAP$</td>
<td>0-2</td>
<td>Swapping algorithm</td>
</tr>
</tbody>
</table>
Methodology: Analytical Model

\[ R_{est} = \frac{1}{PQ\gamma_3} + \frac{3\alpha[(NB + 1)\ln(P) + P]}{2N^2NB} + \frac{3\beta(3P + Q)}{4NPQ} \]

<table>
<thead>
<tr>
<th>Variable</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>Problem size (i.e. matrix size = (N^2))</td>
</tr>
<tr>
<td>NB</td>
<td>Block size</td>
</tr>
<tr>
<td>P x Q</td>
<td>The processor topology in two dimensions</td>
</tr>
<tr>
<td>(\alpha)</td>
<td>Latency of point-to-point message passing, a constant</td>
</tr>
<tr>
<td>(\beta)</td>
<td>Reciprocal of throughput of point-to-point message passing, a constant</td>
</tr>
<tr>
<td>(\gamma_3)</td>
<td>Approximate floating point operations per second when the processor is performing matrix-matrix operations</td>
</tr>
<tr>
<td>(R_{est})</td>
<td>Estimated maximal HPL performance</td>
</tr>
</tbody>
</table>

[Ref: Chou et al. 2007, Peng Wang et al. 2004]
• For performance modeling using Data Mining approach, given a dataset $D = \bigcup_{i=1}^{I} \{x, y\}_i$
  – $x$ is the input {HPL parameters}, $y$ is the targeted output (FLOPs), $I$ is the total no. of samples
  – $D$ is divided into training and testing dataset. Training dataset is used to build the model, while test dataset measures the quality of the model
• DM finds the relationship $Y = f(X)$
Four common data mining techniques were used

- Linear Regression (LR): curve fitting
- M5P: regression tree, LR on subsets
- Multilayer Perceptron (MLP): artificial neural network
- Support Vector Machine (SVM): regression

[Ref: Witten & Frank 2005]
• Feature selection is a Data Mining process that determines which features (i.e. application parameters) are important for estimating the relationship.

• Three techniques were used
  – Relieff: single feature filtering method based on information gain
  – Wrapper: incorporate learning algorithms such as SVM or M5P to assess the quality of features
  – Correlation-based feature selection (Cfs): perform pair-wise correlation analysis of features to determine whether feature is redundant
TACC Ranger System

- 3,936 Sun Blade X6420 nodes with 16 cores per node (4xQuadcore) giving a total of 62,976 cores, 123TB memory, 579.4 TFLOPs (theoretical peak), Infiniband (Sun Magnum), CentOS 4.4, MVAPICH 1.0, PGI 7.1, ACML 4.1
- Our experiments use 64 nodes with 1,024 cores in total
Objectives / Outcome of our work

- To demonstrate the accuracy of data mining (DM) as compared to the analytical approach
  - Performance Modeling
- To present that the proposed DM approach can be used to validate different rules of thumb
  - Concept Validation
- To propose a DM approach for speeding up performance tuning by predicting relative performance (instead of absolute perf)
  - Performance Tuning
### Results: Performance Modeling

<table>
<thead>
<tr>
<th>Method</th>
<th>Pearson</th>
<th>Err %</th>
</tr>
</thead>
<tbody>
<tr>
<td>M5P</td>
<td>0.73</td>
<td>6.17</td>
</tr>
<tr>
<td>LR</td>
<td>0.59</td>
<td>8.16</td>
</tr>
<tr>
<td>MLP</td>
<td>0.63</td>
<td>8.98</td>
</tr>
<tr>
<td>SVM</td>
<td>0.63</td>
<td>7.78</td>
</tr>
<tr>
<td>Analytical Model</td>
<td>0.81</td>
<td>13.24</td>
</tr>
</tbody>
</table>

- **Pearson correlation** indicates how close is the *trend* of the actual and predicted FLOPs
  - 1.0 means perfect correlation, 0.0 means no correlation
- **Err**: average relative error measures the difference (in value) between the actual and predicted FLOPs
Higher the Pearson and lower the Err%, the more reliable is the model.

M5P offers most accurate estimation of FLOPs among DM techniques.

Comparing Analytical Model vs M5P:
- Analytical Model has higher Pearson but higher Err% of 13.24 compared to M5P of 6.17.

M5P has an added feature: model is described by a set of rules, which can help gain insights into the relationship between application parameters & performance.
Feature selection methods applied to full dataset

- Important features selected are listed in order
  - ReliefF: NB, N, P, Q, PMAP
  - SVM-Wrapper: N, NB, P, Q, PFACT, BCAST
  - M5P-Wrapper: N, NB, P, Q, PMAP, BCAST, SWAP
  - Cfs: N, NB, P, Q, PMAP

- N, NB, P, and Q are chosen by all the methods except PMAP which is chosen by 3 of 4.
  - Indicates that these are important parameters/features
  - Consistent with the rules of thumb that NB, P, and Q are critical for determining the system performance
  - PMAP needs to be examined further
• Difference between 4-input and 5-input is small (< 1%)
  – Means that PMAP is relatively less important than N, NB, P, Q when estimating the FLOPs
If $NB \leq 45$, smaller $NB$ is better (Rule2). But if $NB > 45$, larger $NB$ is better.

- Coefficients of $P$ in rules are negative, implying that for $PxQ$ configurations tested \{8x32, 16x16, 32x8\}, \{8x32\} gives highest FLOPs, i.e. $P < Q$ is better.
  - In contrast to conventional wisdom that $P$ should be as close as possible to $Q$. 

---

**Results: Concept Validation – Rules generated by M5P**

**Rule 1:**

\[
\text{IF } P \leq 24 \text{ AND } NB > 90 \text{ THEN}
\]

\[ GFLOPS = 0.11NB - 11.16P - 63.74PMAP + 1214.23 \]

**Rule 2:**

\[
\text{ELSEIF } NB \leq 45 \text{ AND } P \leq 24 \text{ THEN}
\]

\[ GFLOPS = -2.91NB - 0.77P - 39.15PMAP + 940.64 \]

**Rule 3:**

\[
\text{ELSEIF } NB > 45 \text{ THEN}
\]

\[ GFLOPS = 0.56NB - 0.32P + 33.2PMAP + 861.88 \]
- Fix PxQ: Affirms that if NB > 45, higher NB gives higher FLOPs: NB=256 is better than NB=100
- Fix NB: Affirms that P < Q gives higher FLOPs than P=Q

Rules by M5P:
- If NB ≤ 45, smaller NB is better. But if NB > 45, larger NB is better.
- P < Q is better
8 categories of HPL experiments

<table>
<thead>
<tr>
<th>Category (Size_Nodes)</th>
<th>$N$ (Problem Size)</th>
<th>Memory %</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_{16}$</td>
<td>89,600</td>
<td>12%</td>
</tr>
<tr>
<td>$M_{16}$</td>
<td>102,400</td>
<td>15%</td>
</tr>
<tr>
<td>$L_{16}$</td>
<td>134,400</td>
<td>26%</td>
</tr>
<tr>
<td>$XL_{16}$</td>
<td>179,200</td>
<td>47%</td>
</tr>
<tr>
<td>$XS_{64}$</td>
<td>102,400</td>
<td>4%</td>
</tr>
<tr>
<td>$S_{64}$</td>
<td>179,200</td>
<td>12%</td>
</tr>
<tr>
<td>$M_{64}$</td>
<td>204,800</td>
<td>15%</td>
</tr>
<tr>
<td>$L_{64}$</td>
<td>268,800</td>
<td>26%</td>
</tr>
</tbody>
</table>

- For each category, 100 configurations are randomly selected
• Significant difference in the time needed to carry out a benchmark in different categories: e.g. S_16 (500 sec), L_64 (8000 sec); i.e. potential for speed-up.
• Top-10 – the percentage of our predicted configurations that are in actual top ten configurations.

• IsRank$_1$ – a Boolean metric to indicate whether the Rank$_1$ configuration is in the top ten of the predicted configurations. If “Yes”, then the Rank$_1$ configuration is among the top ten of the predicted configurations.

• PRank$_1$ – the predicted rank of the Rank$_1$ configuration. This metric quantify the disparity in predicting Rank$_1$. The ideal value of PRank$_1$ is one. PRank$_1$ is closely related to IsRank$_1$. If PRank$_1$ is smaller than 10, then IsRank$_1$ is definitely a “Yes”.

Rank$_1$ is the configuration which gives the highest actual GFLOPs.
### Illustration to determine performance metrics

<table>
<thead>
<tr>
<th>Actual Rank</th>
<th>Actual GFLOPS</th>
<th>Predicted Rank</th>
<th>Predicted GFLOPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 (Rank_1)</td>
<td>3609</td>
<td>3</td>
<td>2998.821</td>
</tr>
<tr>
<td>2</td>
<td>3458</td>
<td>4</td>
<td>2987.405</td>
</tr>
<tr>
<td>3</td>
<td>3428</td>
<td>2</td>
<td>3004.251</td>
</tr>
<tr>
<td>4</td>
<td>3360</td>
<td>1</td>
<td>3206.309</td>
</tr>
<tr>
<td>5</td>
<td>3252</td>
<td>13</td>
<td>2871.513</td>
</tr>
<tr>
<td>6</td>
<td>3106</td>
<td>24</td>
<td>2843.592</td>
</tr>
<tr>
<td>7</td>
<td>3062</td>
<td>18</td>
<td>2853.277</td>
</tr>
<tr>
<td>8</td>
<td>2993</td>
<td>12</td>
<td>2872.949</td>
</tr>
<tr>
<td>9</td>
<td>2988</td>
<td>56</td>
<td>2760.943</td>
</tr>
<tr>
<td>10</td>
<td>2985</td>
<td>16</td>
<td>2858.989</td>
</tr>
<tr>
<td>11</td>
<td>2838</td>
<td>8</td>
<td>2938.181</td>
</tr>
</tbody>
</table>

\(PRank_1 = 3\)

\(IsRank_1 = Yes\)

\(Top-10 = 40\%\)

Each row is a configuration with different application input parameters.
1. Fixed number of nodes and varying problem size $N$
   – E.g. build model based on $S_{16}$ to predict $M_{16}$ and $L_{16}$.

2. Varying number of nodes and fixed Memory% 
   – E.g. build $S_{16}$ to predict $S_{64}$

3. Varying number of nodes and fixed $N$
   – E.g. build $M_{16}$ to predict $XS_{64}$

4. Varying number of nodes, $N$, and Memory% 
   – E.g. Build $S_{16}$ to predict $L_{64}$

Investigated 4 types of experiments and 20 scenarios

<table>
<thead>
<tr>
<th>Category (Size_Nodes)</th>
<th>$N$ (Problem Size)</th>
<th>Memory %</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_{16}$</td>
<td>89,600</td>
<td>12%</td>
</tr>
<tr>
<td>$M_{16}$</td>
<td>102,400</td>
<td>15%</td>
</tr>
<tr>
<td>$L_{16}$</td>
<td>134,400</td>
<td>26%</td>
</tr>
<tr>
<td>$XL_{16}$</td>
<td>179,200</td>
<td>47%</td>
</tr>
<tr>
<td>$XS_{64}$</td>
<td>102,400</td>
<td>4%</td>
</tr>
<tr>
<td>$S_{64}$</td>
<td>179,200</td>
<td>12%</td>
</tr>
<tr>
<td>$M_{64}$</td>
<td>204,800</td>
<td>15%</td>
</tr>
<tr>
<td>$L_{64}$</td>
<td>268,800</td>
<td>26%</td>
</tr>
</tbody>
</table>
• For models built, 94% correctly predict the configuration with best actual FLOPs
  – Exclude models developed using XS_64 (4% Memory%), which is too small
Results: On obtaining speed-up

<table>
<thead>
<tr>
<th>Training</th>
<th>Prediction</th>
<th>Total Required Time (sec)</th>
<th>Tuning Speed Up</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Data Mining</td>
<td>Direct Tuning</td>
</tr>
<tr>
<td>S_16</td>
<td>M_64</td>
<td>107,000</td>
<td>570,000</td>
</tr>
<tr>
<td>S_16</td>
<td>L_64</td>
<td>130,000</td>
<td>800,000</td>
</tr>
<tr>
<td>M_16</td>
<td>L_64</td>
<td>157,000</td>
<td>800,000</td>
</tr>
</tbody>
</table>

With data mining:

- \(100 \times 500\) (collect 100 samples to train models)
- \(+ 10 \times 5,700\) (actual running of the configurations predicted)
- 107,000 (seconds).

Direct tuning:

- \(100 \times 5,700\) (actual running of the 100 configurations)
- 570,000 (seconds)

Speed up:

\[= 570,000/107,000 \approx 5.3x.\]
Outline

Introduction
Methodology
Experimental Results
Conclusion
• For performance modeling, data mining techniques offer comparable accuracy compared to analytical model
  – Obviate the need of manual construction of the model.
  – Of the four data mining techniques evaluated, M5P is found to offer the most accurate prediction of the ranking.
• Data mining analysis can help in concept validation of tuning guidelines
  – Validated that N, NB, P, Q significantly impact HPL performance whereas PMAP parameter plays a relatively minor role
  – Higher performance is achieved when $P<Q$ compared to when $P>Q$ and $P=Q$ (which is the suggested configuration)
• Our data mining approach for performance tuning yields a potential speed up of 1.7x-6.2x for the performance tuning process, compared to the direct tuning approach.

• M5P offers the most accurate prediction – for 94% of all scenarios, the optimum configuration is correctly identified among the predicted top ten configurations
  – Use relative performance (i.e. rank) of application configurations as the prediction target, rather than the absolute performance (e.g. FLOPs)
Thank you

Rick Goh

rickgoh@ieee.org
gohsm@ihpc.a-star.edu.sg

http://ihpc.a-star.edu.sg