Performance Modeling of Constrained Producer-Consumer Algorithms

Tariq Kamal
Overview

• What is Performance Modeling?
• Why are they important?
• What are CPC algorithms?
• The cost metric
• Extraction of Constants
• Conclusion
Performance Modeling and Analysis

- Estimate the scalability of codes and algorithms
- Estimate the cost of a job
  - CPU hours, job completion time and many more
- Analyze program/algorithm – A Sanity Check
- Identify performance bottlenecks
- Predict performance and capacity
- Optimization of application design
- Improving data and task distribution
  - Load balancing, Scaling
A simple example -- Execution Time per core

PE0
PE1
PE2
PE3
PE4
PE5
PE6

Step1  Step2  Step3
A top level view

Platform or System Model (Hardware, Middleware)

Application Model (Algorithm)

Performance Model

Application Expertise

System Expertise
Performance Model

Platform or System Model (Hardware, Middleware)

Application Model (Algorithm)

Application Expertise

System Expertise
Cost Analysis

• Quantitatively measure the cost of running an algorithm
• Typical way to measure cost of an algorithm is time complexity
  – n, nlog(n), n², n³, exponential
• Does it work when performing cost analysis of parallel algorithms?
  – Portions of algorithm run by different tasks
  – Heterogeneous systems
  – Synchronization
  – Communication overhead
Challenges in Cost Analysis

Cost = Computation + Communication

• Not so Simple, I guess??
  – Portions of algorithm run by different tasks
  – Synchronizations
  – Heterogeneous systems
  – Communication overhead
  – Relative cost of communication and computations
  – Communication/computation overlap
Constrained Producer-Consumer Algorithms

• Computation is done by classes of tasks in two phases
  – Two classes, $A$ and $B$

• Phase 1
  – tasks in class $A$ produce messages and send it to $B$
  – tasks in class $B$ consume messages received from $A$

• Phase 2
  – tasks in class $B$ produce outcome messages and send it to $A$
  – tasks in class $A$ consume messages received from $B$

Note: The consumer will not start processing the messages until all the messages from the producers are received
Why CPC algorithms?

• Many important applications naturally fall in this category
• Genetic Algorithms (GA)
  – Mimics the process of natural selection
  – Optimization and search problems
  – Two steps: fitness and combination
  – Gene mutation
  – Population breeding
• Agent-based contagion models
  – Agents interact at locations
  – Two steps: Agents processing, location processing
  – Spread of fear, disease, and opinion in population
  – Spread of malware, spyware
• Molecular Dynamics
General Simulation Algorithm
(Producer-consumer relationship)

For each simulated day
a) Class A compute messages
b) Class A send messages
c) Class B receive messages
d) Class B compute interactions
e) Class B send outcomes
f) Class A receive outcome
g) Class A process outcomes
end

Class A and Class B objects are mapped onto PEs and Message brokers handles communication
General Simulation Algorithm
(Producer-consumer relationship)

For each simulated day
a) Class A compute messages
b) Class A send messages
c) Class B receive messages
d) Class B compute interactions
e) Class B send outcomes
f) Class A receive outcome
g) Class A process outcomes
end
General Simulation Algorithm
(Producer-consumer relationship)

For each simulated day
a) Class A compute messages
b) Class A send messages
c) Class B receive messages
d) Class B compute interactions
e) Class B send outcomes
f) Class A receive outcome
g) Class A process outcomes
end

Message contains information about the visit (e.g., Agent id, fitness level etc)
General Simulation Algorithm
(Producer-consumer relationship)

For each simulated day
a) Class A compute messages
b) Class A send messages
c) Class B receive messages
d) Class B compute interactions
e) Class B send outcomes
f) Class A receive outcome
g) Class A process outcomes
end

Class B block until all messages have been received
General Simulation Algorithm
(Producer-consumer relationship)

For each simulated day
a) Class A compute messages
b) Class A send messages
c) Class B receive messages
d) Class B compute interactions
e) Class B send outcomes
f) Class A receive outcome
g) Class A process outcomes
end

\[ p_{i \rightarrow j} = 1 - e^{\tau \ln(1 - r_i s_j \rho)} \]
General Simulation Algorithm
(Producer-consumer relationship)

For each simulated day
a) Class A compute messages
b) Class A send messages
c) Class B receive messages
d) Class B compute interactions
e) Class B send outcomes
f) Class A receive outcome
g) Class A process outcomes
end
General Simulation Algorithm
(Producer-consumer relationship)

For each simulated day
a) Class A compute messages
b) Class A send messages
c) Class B receive messages
d) Class B compute interactions
e) Class B send outcomes
f) Class A receive outcome
g) Class A process outcomes
end
General Simulation Algorithm
(Producer-consumer relationship)

For each simulated day
a) Class A compute messages
b) Class A send messages
c) Class B receive messages
d) Class B compute interactions
e) Class B send outcomes
f) Class A receive outcome
g) Class A process outcomes
end
Revisiting Cost Analysis

• Entities in a CPC algorithm can be characterized by a bipartite interaction graph
  – Two computational components
  – One communication Component

• Total simulation cost is the cost of computation and communication

\[ TCost = \text{compCost} + \text{CommCost} \]

For CPC algorithms,

\[ TCost = K_a CA + K_b CB + K_c CC \]

where CA is cost of A, CB is cost of B and CC is cost of communication

• Why Constants?
  – To show the relative contribution of components in the cost metric
Cost Analysis

• The computation in A and B is a function of the number of messages exchanged between A and B

\[
\text{Comp}_A = f(\text{msgsSent}_A, \text{msgsRecv}_A) \\
\text{Comp}_B = f(\text{msgsSent}_B, \text{msgsRecv}_B)
\]

• For each generated message the producer performs some computations
• Similarly, for each received message the reducer performs some processing
Class A
(Generate msgs)

Class B
(process received msgs)

PE0
PE1
PE2
PE3
PE4
PE5
PE6

0

\( t \)
Cost Analysis (cont.)

• Computations happen in steps – separated by synchronization
  – The most loaded processor usually finishes last
  – Other processor wait for most loaded processor to reach synchronization
  – This way the most loaded processor determines the cost of computation
  – In CPC two computational entities, AI – imbalance in A, BI – imbalance in B

• Communication Cost – measured in number of remote messages exchanged in all processors
  – Local messages have small overhead (negligible)
  – Remote messages carry overhead
The Cost Metric

• Quantitatively measure the cost of simulation
• Cost metric is summation of computational imbalances and communication load
  \[ M_{\text{cost}} = K_a A_I + K_b B_I + K_c C_I \]
  *Where \( A_I \) is load imbalance in A type of computations, \( B_I \) is load imbalance in B type of computations, and \( C_I \) is communication load*
• The components shows the dynamics of simulation – very application dependent
  – How do we quantify \( A_I \) and \( B_I \)?
  – Linear, Quadratic, logarithmic or exponential
  – Identify the computational components
  – Compute the number of messages that it send out and receives
  – The function of its summation is the cost of that computational component
• Regression Analysis
  – To determine the quantification function for A and B
  – To compute the constants
  – Validate the model
Regression Analysis

• Performed to estimate cost metric constants
  – running performance experiments to collect real date on real codes on real machines
  – then we use statistical techniques to calculate estimates for these constants
• How good these estimates are?
  – by using the resulting model to see how well it predicts another set of experimental runs
• We performed Regression to estimate constants for EpiSimdemics using IBM SPSS
• Data for regression analysis
  – 732 runs of EpiSimdemics
  – Independent variables: AI (person imbalance), BI (location imbalance) and CL (communication load)
  – Dependent Variable: T (simulation time)
  – Data normally distributed in range for all four variables
• Regression Modeling
  – Data divided into two samples (SampleA and SampleB)
    • SampleA for model extraction
    • SampleB for model verification using cross validation technique
    • R-square value of 0.96 shows that the model is a good choice
Correlations and scaler plots

- Strong linear correlation of T with BI and CL
- Weak or weak negative correlation with AI
- Person shows moderately strong linear relationship

<table>
<thead>
<tr>
<th></th>
<th>T</th>
<th>BI</th>
<th>AI</th>
<th>CL</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>1</td>
<td>.941</td>
<td>-.122</td>
<td>.632</td>
</tr>
<tr>
<td>BI</td>
<td>.941</td>
<td>1</td>
<td>-.233</td>
<td>.479</td>
</tr>
<tr>
<td>AI</td>
<td>-.122</td>
<td>-.233</td>
<td>1</td>
<td>.173</td>
</tr>
<tr>
<td>CL</td>
<td>.632</td>
<td>.479</td>
<td>.173</td>
<td>1</td>
</tr>
</tbody>
</table>
Descriptive Statistics and Normality

- Close to perfect cosine shape means that data is normally distributed in the range for all variables

<table>
<thead>
<tr>
<th></th>
<th>N</th>
<th>Min</th>
<th>Max</th>
<th>Mean</th>
<th>Std. Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>CL</td>
<td>324</td>
<td>19.4</td>
<td>97.8</td>
<td>61.806</td>
<td>19.3687</td>
</tr>
<tr>
<td>AI</td>
<td>324</td>
<td>6.5</td>
<td>109.3</td>
<td>57.942</td>
<td>25.5226</td>
</tr>
<tr>
<td>BI</td>
<td>324</td>
<td>3.9</td>
<td>108.1</td>
<td>53.002</td>
<td>23.8416</td>
</tr>
<tr>
<td>T</td>
<td>324</td>
<td>.8</td>
<td>95.1</td>
<td>40.951</td>
<td>20.9703</td>
</tr>
<tr>
<td>Valid N (listwise)</td>
<td>324</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The model revealed a squared parameter ($B_1^2$)

Rsquare of 0.96 shows that the choice of our components is very good and explains the model

Model equation

\[ M_{\text{cost}} = 0.041 A_1 + 0.733 B_1 + 0.05 B_1^2 + 0.245 C_L \]

<table>
<thead>
<tr>
<th>Model</th>
<th>Coefficientsa</th>
<th>Unstandardized Coefficients</th>
<th>Std. Error</th>
<th>Sig.</th>
<th>Collinearity Statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(Constant)</td>
<td>20.553</td>
<td>1.041</td>
<td>.000</td>
<td></td>
</tr>
<tr>
<td></td>
<td>l_center</td>
<td>.733</td>
<td>.013</td>
<td>.000</td>
<td>.667</td>
</tr>
<tr>
<td></td>
<td>l_center_sq</td>
<td>.005</td>
<td>.000</td>
<td>.000</td>
<td>.985</td>
</tr>
<tr>
<td></td>
<td>p</td>
<td>.041</td>
<td>.011</td>
<td>.000</td>
<td>.834</td>
</tr>
<tr>
<td></td>
<td>c</td>
<td>.245</td>
<td>.016</td>
<td>.000</td>
<td>.685</td>
</tr>
</tbody>
</table>
Histogram and Model Validation

- The error is normally distributed across the zero mean line
- This means that the model is valid and explains the cost of simulation

Scatterplot
Questions, Discussion and Conclusion

- Models greatly benefit application design
- Models are useful to check Performance consistency
- Computational load in CPC type simulations is related to communication load
- Regression Analysis helped quantify the computational components and extract constants