A Prediction-based Approach to Distributed Interactive Applications

Peter A. Dinda
Jason Skicewicz  Dong Lu
Prescience Lab
Department of Computer Science
Northwestern University

http://www.cs.northwestern.edu/~pdinda
Context and Question

How an distributed interactive application running on shared, unreserved computing environment provide consistent responsiveness?
Why Is This Interesting?

• Interactive resource demands set to explode
  • Tools and toys increasingly are physical simulations
  • High-performance computing for everyone

• People provision according to peak demand
  • Responsiveness tied to peak demand
  • 90% of the time CPU or network link is unused

• Opportunity to use the resources smarter
  • New kinds of applications
  • Shared resource pools, resource markets, Grid…
Interactivity Demands Responsiveness

But…

- Dynamically shared resources
  - Commodity environments
- Resource reservations unlikely
  - History
  - End-to-end requirements
- User-level operation
  - Difficult to change OS
  - Want to deploy anywhere

Supporting interactive apps under such constraints is not well understood
Approach

• Soft real-time model
  • Responsiveness requirement -> deadline
  • Advisory, no guarantees

• Adaptation mechanisms
  • Exploit DOF available in environment

• Prediction of resource supply and demand
  • Control the mechanisms to benefit the application
  • Computers as natural systems

Rigorous statistical and systems approach to prediction
Outline

• The story
• Interactive applications
  • Virtualized Audio
• Advisors and resource signals
• The RPS system
  • Intermixed discussion and performance results
• Current work
  • Wavelet-based techniques

All Software and Data publicly available
Application Characteristics

- **Interactivity**
  - Users initiate aperiodic tasks with deadlines
  - Timely, consistent, and predictable feedback needed before next task can be initiated

- **Resilience**
  - Missed deadlines are acceptable

- **Distributability**
  - Tasks can be initiated on any host

- **Adaptability**
  - Task computation and communication can be adjusted

Shared, unreserved computing environments
Applications

• Virtualized Audio
  • Dong Lu

• Image Editing

• Games

• Visualization of massive datasets
  – Interactivity Environment at Northwestern
    • With Watson, Dennis
  – Dv project at CMU
Microphone signals are a result of sound source signals, positions, microphone positions, and the geometry and material properties of the room.

We seek to recover these underlying producers of the microphone signals.
VA: The Forward Problem

Auralization

- sound source positions
- sound source signals
- room geometry/properties
- Listener positions
- Listener wearing headphones

- In general, all inputs are a function of time
- Auralization must proceed in real-time  
  (AccessGrid 2001)
Forward Problem App Structure

Input Audio Streams

RT Task: recompute impulse responses using room model

Physical Simulation Running on Cluster or Grid

Output Audio Streams

User Placed In Room

Impulse Response
Forward Problem App Structure

- Room Model
  - Source and Listener Positions
    - Stream from Source 1
    - Stream from Source 2
      - Room Filter (source 1)
      - Room Filter (source 2)
        - \( \Sigma \)
          - HRTF
            - Headphones
              - Client Workstation

- How complex?
  - Finite Difference Simulation of Wave Equation
    - Impulse Response
      - FIR/IIR Filter Estimation
        - Little Comm.
          - Remote Supercomputer or the Grid

- Which one?
  - Soft Real-time Constraint

- Little Comm.
A Universal Problem

Which host should the application send the task to so that its running time is appropriate?

Known resource requirements

What will the running time be if I...
Advisors

• Adaptation Advisors
  – Real-time Scheduling Advisor
    • Which host should I use?
    • Task assumptions appropriate to interactive applications
    • Soft real-time
    • Known resource demand
    • Best-effort semantics

• Application-level Performance Advisors
  – Running Time Advisor
    • What would running time of task on host x be?
    • Confidence intervals
    • Can build different adaptation advisors
  – Message Transfer Time Advisor
    • How long to transfer N bytes from A to B?

Current focus
Resource Signals

• Characteristics
  • Easily measured, time-varying scalar quantities
  • Strongly correlated with resource supply
  • Periodically sampled (discrete-time signal)

• Examples
  • Host load (Digital Unix 5 second load average)
  • Network flow bandwidth and latency

Leverage existing statistical signal analysis and prediction techniques
Currently: Linear Time Series Analysis and Wavelets
RPS Toolkit

• Extensible toolkit for implementing resource signal prediction systems [CMU-CS-99-138]
  • Growing: RTA, RTSA, Wavelets, GUI, etc

• Easy “buy-in” for users
  • C++ and sockets (no threads)
  • Prebuilt prediction components
  • Libraries (sensors, time series, communication)

• Users have bought in
  • Incorporated in CMU Remos, BBN QuO
  • A number of research users

• RELEASED

http://www.cs.northwestern.edu/~RPS
Example RPS System

RPS components can be composed in other ways
Example RPS System

Application
Nominal time, slack, confidence, host list
Host, running time estimate

Real-time Scheduling Advisor
Nominal time confidence, host
Running time estimate (confidence interval)

Running Time Advisor
Load Prediction Request
Load Prediction Response

Host Load Prediction System
Measurement Stream

Host Load Measurement System

Daemon (one per host)

Library
Measurement and Prediction
Measurement and Prediction Overhead

<1% of CPU At Appropriate Rate

1-2 ms latency from measurement to prediction
2KB/sec transfer rate
# Host Load Traces

- DEC Unix 5 second exponential average
  - 1 Hz
  - Payload tool

<table>
<thead>
<tr>
<th>Machines</th>
<th>Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>August 1997</strong></td>
<td></td>
</tr>
<tr>
<td>13 production cluster</td>
<td>~ one week</td>
</tr>
<tr>
<td>8 research cluster</td>
<td>(over one million</td>
</tr>
<tr>
<td>2 compute servers</td>
<td>samples)</td>
</tr>
<tr>
<td>15 desktops</td>
<td></td>
</tr>
<tr>
<td><strong>March 1998</strong></td>
<td></td>
</tr>
<tr>
<td>13 production cluster</td>
<td>~ one week</td>
</tr>
<tr>
<td>8 research cluster</td>
<td>(over one million</td>
</tr>
<tr>
<td>2 compute servers</td>
<td>million samples)</td>
</tr>
<tr>
<td>11 desktops</td>
<td></td>
</tr>
</tbody>
</table>

[http://www.cs.northwestern.edu/~pdinda/LoadTraces](http://www.cs.northwestern.edu/~pdinda/LoadTraces)
[http://www/cs.northwestern.edu/~pdinda/LoadTraces/playload](http://www/cs.northwestern.edu/~pdinda/LoadTraces/playload)
Salient Properties of Host Load

+/- Extreme variation

+ Significant autocorrelation
  Suggests appropriateness of linear models

+ Significant average mutual information

- Self-similarity / long range dependence

+/- Epochal behavior
  + Stable spectrum during an epoch
  - Abrupt transitions between epochs

+ encouraging for prediction
- discouraging for prediction

(Detailed study in LCR98, SciProg99)
## Linear Time Series Models

<table>
<thead>
<tr>
<th>Model Class</th>
<th>Fit time (ms)</th>
<th>Step time (ms)</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>MEAN</td>
<td>0.03</td>
<td>0.003</td>
<td>Error is signal variance</td>
</tr>
<tr>
<td>LAST</td>
<td>0.75</td>
<td>0.001</td>
<td>Last value is prediction</td>
</tr>
<tr>
<td>BM(p)</td>
<td>46.26</td>
<td>0.001</td>
<td>Average over best window</td>
</tr>
<tr>
<td>AR(p)</td>
<td>4.20</td>
<td>0.149</td>
<td>Deterministic algorithm</td>
</tr>
<tr>
<td>MA(q)</td>
<td>6501.72</td>
<td>0.015</td>
<td>Function Optimization</td>
</tr>
<tr>
<td>ARMA(p,q)</td>
<td>77046.22</td>
<td>0.034</td>
<td>Function Optimization</td>
</tr>
<tr>
<td>ARIMA(p,d,q)</td>
<td>53016.77</td>
<td>0.045</td>
<td>Non-stationarity, FO</td>
</tr>
<tr>
<td>ARFIMA(p,d,q)</td>
<td>3692.63</td>
<td>9.485</td>
<td>Long range dependence, MLE</td>
</tr>
</tbody>
</table>

Pole-zero / state-space models capture autocorrelation parsimoniously

(2000 sample fits, largest models in study, 30 secs ahead)
AR(p) Models

\[ z_t = \phi_1 z_{t-1} + \phi_2 z_{t-2} + \ldots + \phi_p z_{t-p} + \alpha_t \]

- Fast to fit (4.2 ms, AR(32), 2000 points)
- Fast to use (<0.15 ms, AR(32), 30 steps ahead)
- Potentially less parsimonious than other models
AR(16) vs. LAST

Lead Time (seconds)

Average Percent Reduction in Variance

AR(16) models

LAST models
Host Load Prediction Results

• Host load exhibits complex behavior
  • Strong autocorrelation, self-similarity, epochal behavior
• Host load is predictable
  • 1 to 30 second timeframe
• Simple linear models are sufficient
  • Recommend AR(16) or better
• Low overhead

Extensive statistically rigorous randomized study

(Detailed study in HPDC99, Cluster Computing 2000)
Example RPS System

- **Application**
  - Nominal time, slack, confidence, host list
  - Host, running time estimate

- **Real-time Scheduling Advisor**
  - Nominal time confidence, host
  - Running time estimate (confidence interval)

- **Running Time Advisor**
  - Running time estimate

- **Host Load Prediction System**
  - Measurement Stream

- **Host Load Measurement System**

- **Daemon** (one per host)

- **Library**
Running Time Advisor
Example Performance

Near-perfect CIs on typical hosts

Target 95% level

AR(16) predictor

3000 randomized tasks
Running Time Advisor Results

• Predict running time of task
  • Application supplies task size and confidence level
  • Task is compute-bound (current limit)

• Prediction is a confidence interval
  • Expresses prediction error
  • Statistically valid decision-making

• Maps host load predictions and task size through simple model of scheduler
  • Rigorous underlying prediction system essential

• Effective
  • Statistically rigorous randomized evaluation

(Study in HPDC 2001, SIGMETRICS 2001)
Example RPS System

Application

Nominal time, slack, confidence, host list

Host, running time estimate

Real-time Scheduling Advisor

Nominal time, confidence, host

Running time estimate (confidence interval)

Running Time Advisor

Load Prediction Request

Load Prediction Response

Host Load Prediction System

Measurement Stream

Host Load Measurement System

Daemon (one per host)
Real-time Scheduling Advisor
RTSA Results – Probability of Meeting Deadline

Target 95% Level

Deadline / Nominal Time

Fraction of Deadlines Met

Predicted CI < Deadline

Host With Lowest Load

Random Host

16000 tasks
RTSA Results – Probability of Meeting Deadline When Predicted

16000 tasks
RTSA Results

• Application supplies scheduling problem
  • Task size, deadline, and confidence level
  • Task is compute-bound (*current limit*)

• RTSA returns solution
  • Host where task is likely to meet deadline
  • Prediction of running time on that task

• Based on running-time advisor predictions

• Effective
  • Statistically rigorous randomized evaluation

(Study in review)
The Holy Grail

Shared resources scalably provide appropriate measurements and predictions of supply to all comers.

Individual applications measure and predict their resource demands.

Advisors help applications pursue high-level goals, competing with others.
Current work

• Virtualized Audio (with Dong Lu)
• Wavelet-based techniques (with Jason Skicewicz) [HPDC 01]
  • Scalable information dissemination, compression, analysis, prediction
• Network prediction
  • Sampling theory and non-periodic sampling
  • Nonlinear predictive models
  • Minet user-level network stack
• Relational approaches (with Beth Plale and Dong Lu)
  • Grid Forum Grid Information Services RFC [GWD-GIS-012-1]
• Better scheduler models (with Jason Skicewicz)
• Windows monitoring and data reduction (with Praveen Paritosh, Michael Knop, and Jennifer Schopf)
• Application prediction
  • Activation trees
• Clusters for Interactive Applications (with Ben Watson and Brian Dennis)
The Tension

Sensor

Resource-appropriate measurement

Resource Signal (periodic sampling)
Example: host load

Network

Video App

Fine-grain measurement

Grid App

Course-grain measurement
Multi-resolution Views Using 14 Levels

Periodic Resource Measurements

Host Load vs. Time

Periodic Resource Measurements

Host Load vs. Time
Proposed System

Application receives levels based on its needs

Application

Sensor

Wavelet Transform

Level 0

Level M-1

Level M

Network

Level M

Level L

Inverse Wavelet Transform

Stream Interval
Wavelet Compression Gains, 14 Levels

Typical appropriate number of levels for host load, error < 20%
For More Information

- http://www.cs.northwestern.edu/~pdinda
- Resource Prediction System (RPS) Toolkit
  - http://www.cs.northwestern.edu/~RPS
- Prescience Lab
  - http://www.cs.northwestern.edu/~plab

- Load Traces and Payload
  - http://www.cs.northwestern.edu/~pdinda/LoadTraces
  - http://www.cs.northwestern.edu/~pdinda/LoadTraces/playload