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

- Interactive resource demands exploding
  - Tools and toys increasingly are physical simulations
  - New kinds of resource-intensive applications
- Responsiveness tied to **peak demand**
- People provision according to peak demand
  - 90% of the time CPU or network link is unused
- Opportunity to use the resources smarter
  - Build more powerful, more interesting apps
  - Shared resource pools, resource markets, The Grid…
- Resource reservations unlikely
  - History argues against it, partial reservation, …
Approach

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

• Adaptation mechanisms
  • Exploit DOF available in environment

• **Prediction of resource supply and demand**
  • Control the mechanisms to benefit the application
  • Avoid synchronization

Rigorous statistical and systems approach to prediction
Applications

- Virtualized Audio
- Image Editing
- Games
- Scientific visualization of massive datasets
  - Interactivity Environment here
    - With Watson, Dennis
  - Dv project at CMU

\{ Funding difficulties \}
Virtualized Audio (VA) System

Performance Room
- Performer
- Sound Field 1
- Microphones

Amp

Separation

Auralization

HRTF

Real Listening Room
- Virtual Performer (virtual speaker)
- Virtual Sound Field 3
- Listener

Virtual Listening Room
- Virtual Performer
- Sound Field 2
- Listener at Virtual Location

Listener

Headphones

Dong Lu
Curtis Barrett
Virtualized Audio (VA) System

Performance Room
- Performer
- Microphones
  - Sound Field 1

Real Listening Room
- Listener
- Virtual Performer (virtual speaker)
  - Sound Field 3

Auralization
- Listener
- HRTF

HSS

Amp

Separation

Virtual Listening Room
- Virtual Performer
  - Sound Field 2
- Listener at Virtual Location

Dong Lu
Curtis Barrett
Application Structure

Input Audio Streams

Parallel Simulation Running on PLAB Cluster or Grid

RT Task: recompute impulse responses using room model

Output Audio Streams

User Placed In Room

Impulse Response
Application Structure

Room Model

Source and Listener Positions

Stream from Source 1

Room Filter (source 1)

\( \Sigma \)

HRTF

Headphones

Client Workstation

Stream from Source 2

Room Filter (source 2)

Finite Difference Simulation of Wave Equation

Impulse Response

FIR/IIR Filter Estimation

Soft Real-time Constraint

Which one?

Remote Supercomputer or the Grid

How complex?

Which one?
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

Spend a lot of my research time here
Resource Signals

• Characteristics
  • Easily measured, time-varying scalar quantities
  • Strongly correlated with resource availability
  • 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]
- 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
Demo of Measurement+Prediction

RPS components can be composed in other ways.
Demo of RTA

Application

Host Load Measurement System

Host Load Prediction System

Running Time Advisor

Real-time Scheduling Advisor

Nominal time, slack, confidence, host list

Host, running time estimate

Nominal time, confidence, host

Running time estimate (confidence interval)

Request

Response

Measurement Stream

 Daemon
(one per host)

Library

RPS components can be composed in other ways
Demo of RTSA

Application

Real-time Scheduling Advisor

Running Time Advisor

Host Load Prediction System

Host Load Measurement System

Daemon (one per host)

Library

Nominal time, slack, confidence, host list

Host, running time estimate

Running time estimate (confidence interval)

Load Prediction Request

Load Prediction Response

Nominal time, confidence, host list

Measurement Stream

RPS components can be composed in other ways
Current Results

• Apps
  • Virtualized audio: 1st prototype [AGrid 01]
  • Now on 2nd prototype

• Host load
  • Self-similarity, epochs, LRD [LCR 98, SciProg 99]
  • Predictable with AR models [HPDC 99, Cluster 00]
  • Archives + playback [LCR 00]
    http://www.cs.northwestern.edu/~pdinda/LoadTraces

• Running Time Advisor
  • Compute-bound tasks [SIGMETRICS 01, HPDC 01]

• Real-time Scheduling Advisor
  • Compute-bound tasks (In Press)

All Software and Data publicly available
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
  • NSF CAREER grant
• Relational approaches (with Beth Plale and Dong Lu)
  • Grid Forum Grid Information Services RFC [GWD-GIS-012-1]
  • NSF ITR grant
• Better scheduler models (with Jason Skicewicz)
• Windows monitoring and data reduction (with Praveen Paritosh, Michael Knop, and Jennifer Schopf)
• Application prediction
  • Activation trees
• Clusters (with Ben Watson and Brian Dennis)
The Holy Grail

Shared resources scalably provide appropriate measurements and predictions to all comers

Individual applications measure and predict their resource demands

Advisors help applications pursue high-level goals, competing with others
Demo Backup

Application

Nominal time, slack, confidence, host list

Host, running time

Load Prediction

 Daemon (one per host)

RPS components can be composed in other ways
Host Load Measurement System

RPS components can be composed in other ways
Demo Backup

Application

RPS components can be composed in other ways
Outline

• Distributed interactive applications
• Real-time scheduling advisors
• Running time advisor
• Resource signals
• RPS system
• Current work
DV Framework For Distributed Interactive Visualization

• Large datasets (e.g., earthquake simulations)
• Distributed VTK visualization pipelines
• Active frames
  • Encapsulate data, computation, path through pipeline
  • Launched from server by user interaction
  • Annotated with deadline
  • Dynamically chose on which host each pipeline stage will execute and what quality settings to use

http://www.cs.cmu.edu/~dv
Example DV Pipeline for QuakeViz

Logical View

- Simulation Output
- Reading
- Interpolation
- Interpolation
- Isosurface Extraction
- Morphology Reconstruction
- Scene Synthesis
- Rendering

ROI, resolution, contours

Physical View

- Active Frame n+2
- Interpolation
- Active Frame n+1
- Isosurface Extraction
- Active Frame n

Deadline

Which one? How complex?
The Forward Problem - Auralization

- sound source positions
- sound source signals
- room geometry/properties

Auralization Algorithms

Listener positions

Listener wearing Headphones (or HSS scheme)

- In general, all inputs are a function of time
- Auralization filtering must proceed in real-time
- Changes require that the filters be recomputed quickly
Running Time Advisor

Application notifies advisor of task’s computational requirements (nominal time)

Advisor predicts running time on each host

Application assigns task to most appropriate host

Predicted Running Time

Task

nominal time
Real-time Scheduling Advisor

Application notifies advisor of task’s computational requirements (nominal time) and its deadline.

Advisor acquires predicted task running times for all hosts.

Advisor recommends one of the hosts where the deadline can be met.
Variability and Prediction

High Resource Availability Variability

Low Prediction Error Variability

Characterization of variability

Exchange high resource availability variability for low prediction error variability and a characterization of that variability.
Confidence Intervals to Characterize Variability

Application specifies confidence level (e.g., 95%)

Running time advisor predicts running times as a confidence interval (CI)

Real-time scheduling advisor chooses host where CI is less than deadline

CI captures variability to the extent the application is interested in it

“3 to 5 seconds with 95% confidence”
Confidence Intervals And Predictor Quality

**Bad Predictor**
No obvious choice

**Good Predictor**
Two good choices

Good predictors provide smaller CIs
Smaller CIs simplify scheduling decisions
Overview of Research Results

• Predicting CIs is **feasible**
  • Host load prediction using AR(16) models
  • Running time estimation using host load predictions

• Predicting CIs is **practical**
  • RPS Toolkit (inc. in CMU Remos, BBN QuO)
  • Extremely low-overhead online system

• Predicting CIs is **useful**
  • Performance of real-time scheduling advisor

Measured performance of real system
Statistically rigorous analysis and evaluation
Experimental Setup

- **Environment**
  - Alphastation 255s, Digital Unix 4.0
  - Workload: host load trace playback
  - Prediction system on each host

- **Tasks**
  - Nominal time $\sim U(0.1,10)$ seconds
  - Interarrival time $\sim U(5,15)$ seconds

- **Methodology**
  - Predict CIs / Host recommendations
  - Run task and measure
Predicting CIs is Feasible

Near-perfect CIs on typical hosts

AR(16) predictor

3000 randomized tasks

Nominal Time (seconds)
Predicting CIs is Practical - RPS System

1-2 ms latency from measurement to prediction
2KB/sec transfer rate

<2% of CPU At Appropriate Rate

1-2 ms latency from measurement to prediction
2KB/sec transfer rate
Predicting CIs is Useful - Real-time Scheduling Advisor

16000 tasks
Predicting CIs is Useful - Real-time Scheduling Advisor

16000 tasks
Limitations

• Compute-intensive apps only
  • Host load based

• Network prediction not solved
  • Not even limits are known

• Poor scheduler models

• Poor integration of resource supply predictions

• Programmer supplies resource demand
  • Application resource demand prediction is nascent and needed
Conclusion

• Prediction-based approach to responsive distributed interactive applications

Peter Dinda, Jason Skicewicz, Dong Lu

http://www.cs.nwu.edu/~pdinda
http://www.cs.nwu.edu/~pdinda/RPS.html
Wave Propagation Approach

\[ \frac{\partial^2 p}{\partial^2 t} = \frac{\partial^2 p}{\partial^2 x} + \frac{\partial^2 p}{\partial^2 y} + \frac{\partial^2 p}{\partial^2 z} \]

- Captures all properties except absorption
- absorption adds 1st partial terms
- LTI simplification
LTI Simplification

• Consider the system as LTI - Linear and Time-Invariant
• We can characterize an LTI system by its impulse response h(t)
• In particular, for this system there is an impulse response from each sound source i to each listener j: h(i,j,t)
• Then for sound sources s_i(t), the output m_j(t) listener j hears is m_j(t) = Σ_i h(i,j,t) * s_i(t), where * is the convolution operator
## Design Space

<table>
<thead>
<tr>
<th>Measurement</th>
<th>Application-oriented</th>
<th>Resource-oriented</th>
</tr>
</thead>
<tbody>
<tr>
<td>Measurement</td>
<td>Task executions</td>
<td>Resource-specific</td>
</tr>
<tr>
<td>Advantages</td>
<td>Close to application</td>
<td>Periodic measurements</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Scales with resources</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Easy sharing</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Easy exploration</td>
</tr>
<tr>
<td>Disadvantages</td>
<td>Aperiodic measurements</td>
<td>Distant from application</td>
</tr>
<tr>
<td></td>
<td>Difficult to make scale</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Limited sharing</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Limited exploration</td>
<td></td>
</tr>
</tbody>
</table>

Can the gap between the resources and the application can be spanned? **yes!**
### 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><strong>MEAN</strong></td>
<td>0.03</td>
<td>0.003</td>
<td>Error is signal variance</td>
</tr>
<tr>
<td><strong>LAST</strong></td>
<td>0.75</td>
<td>0.001</td>
<td>Last value is prediction</td>
</tr>
<tr>
<td><strong>BM(p)</strong></td>
<td>46.26</td>
<td>0.001</td>
<td>Average over best window</td>
</tr>
<tr>
<td><strong>AR(p)</strong></td>
<td>4.20</td>
<td>0.149</td>
<td>Deterministic algorithm</td>
</tr>
<tr>
<td><strong>MA(q)</strong></td>
<td>6501.72</td>
<td>0.015</td>
<td>Function Optimization</td>
</tr>
<tr>
<td><strong>ARMA(p,q)</strong></td>
<td>77046.22</td>
<td>0.034</td>
<td>Function Optimization</td>
</tr>
<tr>
<td><strong>ARIMA(p,d,q)</strong></td>
<td>53016.77</td>
<td>0.045</td>
<td>Non-stationarity, FO</td>
</tr>
<tr>
<td><strong>ARFIMA(p,d,q)</strong></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)
Host Load Traces

- DEC Unix 5 second exponential average
  - Full bandwidth captured (1 Hz sample rate)
  - Long durations

<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 samples)</td>
</tr>
<tr>
<td>2 compute servers</td>
<td></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 samples)</td>
</tr>
<tr>
<td>2 compute servers</td>
<td></td>
</tr>
<tr>
<td>11 desktops</td>
<td></td>
</tr>
</tbody>
</table>
Results for Host Load

- 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
- Predictions are useful
  - Can compute effective CIs from them

Extensive statistically rigorous randomized study