Resource Demand Estimation in Distributed, Service-Oriented Applications using LibReDE

Simon Spinner

University of Würzburg – Chair of Software-Engineering

Symposium on Software Performance, Munich, Germany
05/11/2015
Context

Service-oriented applications:

- Integration of different applications (→ SOA)
- Architecture of one complex application (→ Microservices)

Edge Services  Business Services  Data Services

[Diagram showing network of services]

 NETFLIX  facebook  LinkedIn
What are resource demands?

Example SEFF in PCM:

A resource demand is the time a unit of work (e.g., request or internal action) spends obtaining service from a resource (e.g., CPU or hard disk) in a system.
Resource Demand Estimation

Direct Measurement
Requires specialized infrastructure to monitor low-level statistics.

Examples:
- TimerMeter + ByCounter
- PMWT
- Dynatrace

Statistical Estimation
Use of statistical techniques on high-level monitoring statistics.

Examples:
- Linear regression
- Kalman filtering
- Nonlinear optimization
- Etc.
Residence times may be missing or inaccurate
→ Use end-to-end response times instead?
→ Existing work limited to 3-tier applications
Approach Overview

Descartes Modeling Language (DML)

1. Workload Description

2. Estimation Problem(s)

3. Estimation
1. DERIVE WORKLOAD DESCRIPTION
Workload Description

calledService

Service

Task

ExternalCall

ResourceDemand

Resource

WorkloadDescription

calledService

Service

Task

ExternalCall

ResourceDemand

Resource

WorkloadDescription
Assumptions

- Any parameter dependencies are solved
- Coarse-grained internal actions
  - Not more than one internal action per resource type in RDSEFF
  - Internal actions in top-level component internal behavior of RDSEFF
- Arbitrary control flow for external calls
  - Loops, branches, forks, etc.
- Product-form workload description
Mapping to DML (1/2)

- Component instance reference
  - Path of assembly contexts
  - Unique within system
- Service in workload description maps to
  - component service
  - of provided interface role
  - of a component instance reference
Mapping to DML (2/2)

- Further mappings
  - Internal action ↔ Resource demand
  - External call ↔ External call
  - Processing resource ↔ Resource

- Visit counts of external calls are derived from DML
  - Loops: average iteration count
  - Branches: weights based on branching probabilities

- Fork actions
  - Without synchronization → Ignore fork
  - With synchronization → Future work
2. DERIVE ESTIMATION PROBLEM
Estimation Problem

- **State model**
  - Definition of state variables (i.e., resource demands)
  - Constraints on state variables
  - Initial values of state variables

- **Observation model**
  - Analytical function $\hat{y} = h(\hat{x})$
  - $\hat{y}$: vector of observations
  - $\hat{x}$: vector of state variables

- **Estimation algorithm**
  - Mathematical solution algorithm
  - E.g., non-linear constrained optimization
Strategies

- **Resource level**
  - Use only utilization and throughput measurements

- **Tier level**
  - Use residence times

- **System level**
  - Use end-to-end response times

Mathematical formula:

\[ T_c = \sum_{M \in S} V_{M,c} \cdot R_{M,c} + D_{0,c} \]

- Number of visits at tier M of service c
- Residence time at tier M of service c
- Constant delay of service c

For each tier M in system S
3. ESTIMATION
Optimization

- Non-linear, constrained optimization
  - Interior-point solver (→ Ipopt library\(^1\))
  - Integrated in LibReDE

- Minimize:
  - Relative difference between
    - Observed and calculated response times
    - Observed and calculated utilization
  - Constant delays

- Equal weights for all parts of the objective function

\(^1\) https://projects.coin-or.org/Ipopt
State Space

- Ipopt requires
  - Jacobi matrix
  - Hessian matrix for Lagrange multipliers
- Use Rall’s system for automatic differentiation
  - Automatic calculation of all partial derivatives
  - Memory and computational complexity may be limiting
- See DerivativeStructure in Apache Commons Math
CASE STUDY
Experiment Setup
Results: Transaction Rate 60 (1/2)

Prediction Error Response Time

Relative Error (%)

<table>
<thead>
<tr>
<th>Activity</th>
<th>System-level</th>
<th>Tier-level</th>
<th>Resource-level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Purchase</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Manage</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Browse</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mfg EJB</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mfg WS</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Results: Transaction Rate 60 (2/2)

Prediction Error Utilization

- VM 2
- VM 3
- VM 4
- VM 5
- VM 6
- VM 7
- VM 9

Relative Error (%)

Approach

Case Study
Results: Transaction Rate 100 (1/2)

Prediction Error Response Time

<table>
<thead>
<tr>
<th>Activity</th>
<th>System-level</th>
<th>Resource-level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Purchase</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Manage</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Browse</td>
<td>40%</td>
<td>60%</td>
</tr>
<tr>
<td>Mfg EJB</td>
<td>60%</td>
<td>60%</td>
</tr>
<tr>
<td>Mfg WS</td>
<td>60%</td>
<td>60%</td>
</tr>
</tbody>
</table>
Results: Transaction Rate 100 (2/2)

Prediction Error Utilization

Relative Error (%)

VM 2  VM 3  VM 4  VM 5  VM 6  VM 7  VM 9

- System-level
- Resource-level
Summary

- Extended LibReDE to support service-oriented applications
  - Control flow awareness
  - Based on end-to-end response times
- Identified different strategies for resource demand estimation
  - Resource-level
  - Tier-level
  - System-level
- Experimental results show
  - System-level is a feasible alternative
  - Tier-level highly depends on accuracy of residence times
http://descartes.tools/librede
Eclipse Public License (EPL)