Timing Behavior Anomaly Detection
for Automatic Failure Detection and Diagnosis
Research visit at Charles University Prague

Matthias Rohr
matthias.rohr@informatik.uni-oldenburg.de

Graduate School TrustSoft, Software Engineering Group
Department of Computing Science, University of Oldenburg

10th of April 2007
Motivation

Failure diagnosis in business-critical software systems

Manual failure diagnosis is time-consuming and error-prone

Runtime behavior observations are indicative for failure diagnosis

Matthias Rohr (TrustSoft)

Timing Behavior Anomaly Detection

10th of April 2007
Motivation

Complex Software System

Users

Administrators

Failure diagnosis in business-critical software systems is manual and time-consuming due to limited knowledge of users and administrators. Runtime behavior observations are indicative for failure diagnosis.

Matthias Rohr (TrustSoft)
Motivation

Failure diagnosis in business-critical software systems

- Manual failure diagnosis is time-consuming and error-prone
Motivation

Complex Software System with Monitoring

Failure diagnosis in business-critical software systems

- Manual failure diagnosis is time-consuming and error-prone
- Runtime behavior observations are indicative for failure diagnosis
Motivation

Vision

- Automatic localization of faults through runtime behavior evaluation
Approach

- Automatic localization of faults through runtime behavior evaluation
- Automatic detection of timing behavior anomalies in software systems
Approach

- Automatic localization of faults through runtime behavior evaluation
- Automatic detection of timing behavior anomalies in software systems

Research questions:
- How can anomalies be detected in timing behavior?
- How can system usage variations be addressed in timing behavior evaluation?
- What is the relation between software faults and runtime timing behavior?
Outline

1. Foundations
   - Dependability
   - Anomaly Detection
   - Software Performance

2. Creation of the timing behavior profile

3. Fault Localization

4. Evaluation

5. Related work

6. Conclusions
Dependability Terminology [Avižienis et al., 2004]

Threats to dependability

**Fault**  Root-cause of a failure

**Error**  Incorrect system state

**Failure**  Deviation from correct system behavior visible to the user
Dependability Terminology [Avižienis et al., 2004]

**Threats to dependability**
- **Fault**: Root-cause of a failure
- **Error**: Incorrect system state
- **Failure**: Deviation from correct system behavior visible to the user

**Failure Diagnosis:**
- Failure detection
- Identification of faults
- **Fault localization**
Availability

Availability: Common definition (e.g., [Musa et al., 1987])

\[
\text{Availability} = \frac{\text{MTTF}}{\text{MTTF} + \text{MTTR}}
\]

**MTTF**  Mean Time to Failure

**MTTR**  Mean Time to Repair
Availability

Availability: Common definition (e.g., [Musa et al., 1987])

\[
\text{Availability} = \frac{\text{MTTF}}{\text{MTTF} + \text{MTTR}}
\]

- **MTTF**: Mean Time to Failure
- **MTTR**: Mean Time to Repair

Two alternative strategies to increase availability

- Increase of mean time to failure (reliability)
- Decrease of mean time to repair
  - **Failure diagnosis support**
An anomaly is a deviation from “normal” system behavior.
An anomaly is a deviation from “normal” system behavior

Normal system behavior:

- Static reference values (e.g., mean response time over a day ≤ T )
- Analytical or statistical models in dependence to system influences and historical system behavior
Anomaly Detection (2/2)

Methods to create normal behavior profiles

- Manual specification
- **Automatic profile learning from observations**
Methods to create normal behavior profiles
- Manual specification
- **Automatic profile learning from observations**

Challenges of anomaly detection:
- False alarms
- System usage
- Nonlinear system behavior, modeling uncertainties
Methods to create normal behavior profiles
- Manual specification
- **Automatic profile learning from observations**

Challenges of anomaly detection:
- False alarms
- System usage
- Nonlinear system behavior, modeling uncertainties

Typical application domains:
- Industrial manufacturing, large-scale control systems [Palade et al., 2006]
- Network management [Maxion, 1990]
- Intrusion detection (Security) [Denning, 1987]
Software Timing Behavior

Influences to software timing behavior:

- **System architecture:**
  - Hardware resource capacity
  - Software design

- **System usage:** [cp. Sabetta and Koziolek, 2007]:
  - Workload intensity (e.g., number of active users)
  - Service demand characteristics (e.g., individual request parameters)

- **System state**
  - Performance tuning (e.g., caching, load balancing), ...
  - Server virtualization
Outline

1. Foundations

2. Creation of the timing behavior profile
   - Instrumentation
   - Monitoring
   - Analysis of Execution Sequences
   - Analysis of Workload Intensity

3. Fault Localization

4. Evaluation

5. Related work

6. Conclusions
Timing behavior anomalies:

- Deviations from normal timing behavior (here: response times) of operations of a software system
- e.g., exceptional high or low response times
Failure diagnosis through online timing behavior evaluation

Timing behavior anomalies:
- Deviations from normal timing behavior (here: response times) of operations of a software system
- e.g., exceptional high or low response times

Relation between software faults and timing behavior anomalies
Failure diagnosis through online timing behavior evaluation

Timing behavior anomalies:
- Deviations from normal timing behavior (here: response times) of operations of a software system
- e.g., exceptional high or low response times

Relation between software faults and timing behavior anomalies:
- Software faults tend to cause timing behavior anomalies [Kao et al., 1993]
- Successful fault localization based on timing behavior anomalies [Agarwal et al., 2004]
- Response times in enterprise resource planning systems (ERP) are often log-normally distributed [Mielke, 2006]
Overview

Timing behavior anomaly detection for failure diagnosis

Initial activities
- Instrumentation for Monitoring
- Monitoring
- Creation of the timing behavior profile
- Timing behavior profile

Continuous activities
- Monitoring
- Update of timing behavior profile
- Timing behavior profile

Activities during failure diagnosis
- Log:
  - response times
  - execution sequences
- Timing behavior profile
- Anomaly detection
- Anomaly analysis
- Diagnosis report

Matthias Rohr (TrustSoft)
Overview

Timing behavior anomaly detection for failure diagnosis

Initial activities
- Instrumentation for Monitoring
- Monitoring
- Creation of the timing behavior profile
  - Timing behavior profile

Continuous activities
- Monitoring
- Update of timing behavior profile
  - Timing behavior profile

Activities during failure diagnosis
- Log:
  - response times
  - execution sequences
- Timing behavior profile
- Anomaly detection
- Anomaly analysis
  - Diagnosis report
Overview

Timing behavior anomaly detection for failure diagnosis

Initial activities

Instrumentation for Monitoring → Monitoring → Creation of the timing behavior profile → Timing behavior profile
Overview

Timing behavior anomaly detection for failure diagnosis

Initial activities

- Instrumentation for Monitoring
- Monitoring
- Creation of the timing behavior profile

Execution sequence analysis

- Operation analysis
- Execution sequence analysis
- Workload intensity analysis

Timing behavior profile
Instrumentation for Monitoring

Execution sequence analysis

Worload intensity analysis

Monitoring of Response times (Start and end of an operation execution)

Execution sequences of operations for each thread

Instrumentation challenges:
- Measurement metrics
- Number and position of measurement points [Focke et al., 2007a]
- Maintainable integration of measurement logic [Focke et al., 2007b]
Monitoring of
- **Response times** (Start and end of an operation execution)
- **Execution sequences** of operations for each thread
Monitoring of
  - Response times (Start and end of an operation execution)
  - Execution sequences of operations for each thread

Instrumentation challenges:
  - (Measurement metrics)
  - Number and position of measurement points [Focke et al., 2007a]
  - Maintainable integration of measurement logic [Focke et al., 2007b]
Creation of the timing behavior profile

Monitoring

Instrumentation
for Monitoring

Operation
analysis

Execution
sequence
analysis

Worload
intensity
analysis

- Operation analysis
- Execution sequence analysis
- Workload intensity analysis

Trace reconstruction of execution sequences from monitoring log:

Operations: \( O = \{a, b, c\} \)

Executions with TraceID 1:

\( E_1 = \{a, b, c_1, c_2\} \)

Execution sequence:

\[
t_1 = (a, ac_1, c_1a, ab, bc_2, c_2b, ba, a)
\]

<table>
<thead>
<tr>
<th>Operation</th>
<th>TraceID</th>
<th>( t_{in} )</th>
<th>( t_{out} )</th>
<th>( t_{out} - t_{in} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>1</td>
<td>0000</td>
<td>0150</td>
<td>150</td>
</tr>
<tr>
<td>c</td>
<td>1</td>
<td>0030</td>
<td>0050</td>
<td>20</td>
</tr>
<tr>
<td>b</td>
<td>1</td>
<td>0060</td>
<td>0140</td>
<td>80</td>
</tr>
<tr>
<td>c</td>
<td>1</td>
<td>0090</td>
<td>0130</td>
<td>40</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>c</td>
<td>2</td>
<td>0340</td>
<td>0358</td>
<td>18</td>
</tr>
<tr>
<td>c</td>
<td>2</td>
<td>0400</td>
<td>0437</td>
<td>37</td>
</tr>
</tbody>
</table>

Matthias Rohr (TrustSoft)
Trace reconstruction of execution sequences from monitoring log:

- Operations: \( O = \{ a, b, c \} \)
- Executions with TraceID 1: \( E^1 = \{ a, b, c_1, c_2 \} \)

<table>
<thead>
<tr>
<th>Operation</th>
<th>TraceID</th>
<th>( t_{in} )</th>
<th>( t_{out} )</th>
<th>( t_{out} - t_{in} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>1</td>
<td>0000</td>
<td>0150</td>
<td>150</td>
</tr>
<tr>
<td>c</td>
<td>1</td>
<td>0030</td>
<td>0050</td>
<td>20</td>
</tr>
<tr>
<td>b</td>
<td>1</td>
<td>0060</td>
<td>0140</td>
<td>80</td>
</tr>
<tr>
<td>c</td>
<td>1</td>
<td>0090</td>
<td>0130</td>
<td>40</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>c</td>
<td>2</td>
<td>0340</td>
<td>0358</td>
<td>18</td>
</tr>
<tr>
<td>c</td>
<td>2</td>
<td>0400</td>
<td>0437</td>
<td>37</td>
</tr>
</tbody>
</table>
Trace reconstruction of execution sequences from monitoring log:

- Operations: \( O = \{a, b, c\} \)
- Executions with TraceID 1: \( E^1 = \{a, b, c_1, c_2\} \)
- Execution sequence \( t_1 = (\$a, ac_1, c_1 a, ab, bc_2, c_2 b, ba, a\$) \)
Trace reconstruction of execution sequences from monitoring log:

- Operations: $O = \{a, b, c\}$
- Executions with TraceID 1: $E^1 = \{a, b, c_1, c_2\}$
- Execution sequence $t_1 = (\$a, ac_1, c_1 a, ab, bc_2, c_2 b, ba, a\$)
Associate response times to operations:

- \( RT := \) all response times
- \( RT(o) := \) response times of one operation \( o \)

\[
\begin{align*}
\text{All response times} & \quad \text{Response time per operation} \\
RT = (150, 80, \ldots) & \quad RT(a) = (150, \ldots) \\
RT(b) = (80, \ldots) & \\
RT(c) = (40, 20, 37, 18, \ldots) &
\end{align*}
\]
Associate response times to operations:

- $RT :=$ all response times
- $RT(o) :=$ response times of one operation $o$

All response times:

$$RT = (150, 80, \ldots)$$

Response time per operation:

- $RT(a) = (150, \ldots)$
- $RT(b) = (80, \ldots)$
- $RT(c) = (40, 20, 37, 18, \ldots)$

Statistical description of $RT(o) = (rt_1, \ldots, rt_n)$

- Probability density functions, histograms
- Location parameters: Mean, Median, Mode
Creation of the timing behavior profile

Analysis of Execution Sequences

Instrumentation for Monitoring

Monitoring

Operation analysis

Execution sequence analysis

Workload intensity analysis

Instrumentation

for Monitoring

Execution sequence analysis

Workload intensity analysis

Prob. density

Response time

:Bookshop

:CRM

:Catalog

Matthias Rohr (TrustSoft)
Creation of the timing behavior profile

Analysis of Execution Sequences

Instrumentation for Monitoring

Monitoring

Operation sequence analysis

Workload intensity analysis

Matthias Rohr (TrustSoft)

Response time

Prob. density

:Bookshop :CRM :Catalog

Response time

Bookshop

CRM

Catalog
Creation of the timing behavior profile

Instrumentation for Monitoring → Monitoring → Operation analysis → Execution sequence analysis → Worload intensity analysis

Analysis of Execution Sequences

Instrumentation

Execution sequence analysis

Worload intensity analysis

Response time (ms):

:Bookshop
:CRM
:Catalog

Prob. density

Separation to achieve "trace-aware" timing behavior evaluation

Prob. density

Prob. density

Response time

Response time (ms)
Prefix of an execution sequence \( t \in T \) of an execution \( e \in E \): 

\[
p : T \times E \rightarrow T; (t, e) \mapsto (m_j^i = 1 \text{ with } m_j^i \text{ as pair } e', e)
\]

Example: Prefix of an execution sequence 

\[
(a, ac_1, c_1a, ab, bc_2, c_2b)
\]
Example: Prefix of an execution sequence

\[ p(t_1, c_1) \]

\[ t_1 = (a, ac_1, c_1a, ab, bc_2, c_2b, ba, a$) \]
Example: Prefix of an execution sequence

\[
p(t_1, c_1)
\]

\[
t_1 = (a, ac_1, c_1a, ab, bc_2, c_2b, ba, a$)
\]

\[
p(t_1, c_2)
\]
Prefix of an execution sequence $t \in T$ of an execution $e \in E$:

$$p : T \times E \rightarrow T; (t, e) \mapsto (m_i)_{i=1}^j$$

with $m_j$ as pair $(e', e)$.

Example: Prefix of an execution sequence

$$t_1 = (a, ac_1, c_1a, ab, bc_2, c_2b, ba, a\$)$$
Distinction of **response times** based on prefixes

The timing behavior observations of an operation \( o \) are distinguished based on their prefix \( p \), denoted \( RT_p = (rt_1, \ldots, rt_n) \).
Distinction of \textit{response times} based on prefixes

The timing behavior observations of an operation $o$ are distinguished based on their prefix $p$, denoted $RT_p = (rt_1, \ldots, rt_n)$.

Example:

$p(t_1, c_1)$

$t_1 = t_2 = (\{a, ac_1, c_1a, ab, bc_2, c_2b, ba, a\}$

$p(t_1, c_2)$

$RT_{(a, ac_1)} = (40, 37, \ldots)$

$RT_{(a, ac_1, c_1a, ab, bc_2)} = (20, 18, \ldots)$
Response times per operation:
- RT(a) = (150, ... )
- RT(b) = (80, ... )
- RT(c) = (20, 40, 18, 37, ... )

Distinction based on prefix:
- RT_{p1} = (150, ... )
- RT_{p2} = (80, ... )
- RT_{p3} = (20, 18, ... )
- RT_{p4} = (40, 37, ... )

- p3 = ($a, ac_1$)
- p4 = ($a, ac_1, c_1 a, ab, bc_2$)
The workload intensity during an execution influences the response times.

![Graph showing the relationship between workload intensity and response time. The graph displays two lines: a dashed blue line representing the average response time in milliseconds, and a dotted red line representing the median response time in milliseconds. The x-axis represents workload intensity, and the y-axis represents response time. The graph illustrates an increasing trend as workload intensity increases.]
The workload intensity during an execution influences the response times.

What is the expected response time distribution of an operation for a particular workload intensity?
The workload intensity during an execution influences the response times.

What is the expected response time distribution of an operation for a particular workload intensity?

Metric for workload intensity $w(e)$:

- Average number of active application threads during the operation execution $e$
Process:

- Determine the workload intensity for each execution monitored
Process:

- Determine the workload intensity for each execution monitored.

\[ RT_p = (rt_1, ..., rt_n) \] is extended to

\[ RT_p' = ((rt_1, w_1), \ldots, (rt_n, w_n)) \]
Process:

- Determine the workload intensity for each execution monitored

\[ RT_p = (rt_1, \ldots, rt_n) \] is extended to

\[ RT'_p = ((rt_1, w_1), \ldots, (rt_n, w_n)) \]

- Approximation of normalized probability density functions

\[ f^{w}_{RT_p} : \mathbb{R} \rightarrow [0, 1]; \quad rt \mapsto f^{w}_{RT_p}(rt) \]
Process:

- Determine the workload intensity for each execution monitored

\[ RT_p = (rt_1, \ldots, rt_n) \] is extended to \[ RT'_p = ((rt_1, w_1), \ldots, (rt_n, w_n)) \]

- Approximation of normalized probability density functions
  \[ f^{w}_{RT_p} : \mathbb{R} \rightarrow [0, 1]; rt \mapsto f^{w}_{RT_p}(rt) \]

- Example: Approximated normal distributions for response times in dependence to the workload intensity \( w \) (normalized to \([0, 1]\))
Process:

- Determine the workload intensity for each execution monitored
- $RT_p = (rt_1, ..., rt_n)$ is extended to $RT_p' = ((rt_1, w_1), ..., (rt_n, w_n))$
- Approximation of normalized probability density functions $f_{RT_p}^w : \mathbb{R} \rightarrow [0, 1]; rt \mapsto f_{RT_p}^w (rt)$
- Example: Approximated normal distributions for response times in dependence to the workload intensity $w$ (normalized to $[0, 1]$)
Process:

- Determine the workload intensity for each execution monitored

\[ RT_p = (rt_1, \ldots, rt_n) \] is extended to \[ RT'_p = ((rt_1, w_1), \ldots, (rt_n, w_n)) \]

- Approximation of normalized probability density functions

\[ f^w_{RT_p} : \mathbb{R} \rightarrow [0, 1]; rt \mapsto f^w_{RT_p}(rt) \]

- Example: Approximated normal distributions for response times in dependence to the workload intensity \( w \) (normalized to \([0, 1])

\[ w = \text{workload intensity} \]

- \( w=1 \)

- \( w=5 \)

- \( w=10 \)

\[ \text{response time in ms} \]

\[ \text{normalized prob. density (pdf(rt))} \]
Process:

- Determine the workload intensity for each execution monitored
  \[ RT_p = (rt_1, ..., rt_n) \]
- \( RT'_p = ((rt_1, w_1), \ldots, (rt_n, w_n)) \)
- Approximation of normalized probability density functions
  \[ f_{RT_p}^w : \mathbb{R} \to [0, 1]; rt \mapsto f_{RT_p}^w (rt) \]
- Example: Approximated normal distributions for response times in dependence to the workload intensity \( w \) (normalized to \([0, 1]\))
All monitored response times | Response times per operation | Distinction based on prefix | Modeling of workload intensity
---|---|---|---
RT | RT(a) = (150, ... ) | RT\textsubscript{p1} = (150, ... ) | \( f_{RT\textsubscript{p1}}^w (rt) \)
| RT(b) = (80, ... ) | RT\textsubscript{p2} = (80, ... ) | \( f_{RT\textsubscript{p2}}^w (rt) \)
| RT(c) = (20, 40, 18, 37, ... ) | RT\textsubscript{p3} = (20, 18, ... ) | \( f_{RT\textsubscript{p3}}^w (rt) \)
|  | RT\textsubscript{p4} = (40, 37, ... ) | \( f_{RT\textsubscript{p4}}^w (rt) \)

p1 = (a) | p3 = (a, ac) | p2 = (a, ac, c, a, ab) | p4 = (a, ac, c, a, ab, bc)

### Timing behavior profile

The timing behavior profile consists of a function \( f_{RT_p}^w \) for each prefix of the monitoring data. The values \( f_{RT_p}^w (rt) \in [0, 1] \) describe how “normal” a response time \( rt \) is under consideration of a workload intensity \( w \) and a prefix \( p \).
Outline

1. Foundations
2. Creation of the timing behavior profile
3. Fault Localization
4. Evaluation
5. Related work
6. Conclusions
Overview Fault Localization

Initial Activities
- Instrumentation for Monitoring
- Monitoring
- Creation of Timing behavior profile
- Timing behavior profile

Continuous activities
- Monitoring
- Update of Timing behavior profile
- Timing behavior profile

Activities during diagnosis after detection of a failure
- Monitoring Log of some time period before the failure:
  - Response times
  - Execution sequences
- Timing behavior profile
- Anomaly detection
- Anomaly analysis
- Diagnosis report
After detection of a failure at time \( t_a \):

- Determination of response times, prefixes and workload intensities (for each execution) for the time period \( [t_a - \delta, t_a] \):
Activities of Fault Localization (1/2)

After *detection* of a failure at time $t_a$:

- Determination of response times, prefixes and workload intensities (for each execution) for the time period $[t_a - \delta, t_a]$:

<table>
<thead>
<tr>
<th>O</th>
<th>TraceID</th>
<th>$t_{in}$</th>
<th>$t_{out}$</th>
<th>$t_{out} - t_{in}$</th>
<th>p</th>
<th>w</th>
</tr>
</thead>
<tbody>
<tr>
<td>Catalog.getBook(..)</td>
<td>121</td>
<td>1182</td>
<td>1201</td>
<td>19</td>
<td>p17</td>
<td>17</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bookshop.query(..)</td>
<td>131</td>
<td>1195</td>
<td>1221</td>
<td>26</td>
<td>p41</td>
<td>21</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
After *detection* of a failure at time $t_a$:

1. Determination of response times, prefixes and workload intensities (for each execution) for the time period $[t_a - \delta, t_a]$:

<table>
<thead>
<tr>
<th>O</th>
<th>TraceID</th>
<th>$t_{in}$</th>
<th>$t_{out}$</th>
<th>$t_{out} - t_{in}$</th>
<th>p</th>
<th>w</th>
</tr>
</thead>
<tbody>
<tr>
<td>Catalog.getBook(..)</td>
<td>121</td>
<td>1182</td>
<td>1201</td>
<td>19</td>
<td>p17</td>
<td>17</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bookshop.query(..)</td>
<td>131</td>
<td>1195</td>
<td>1221</td>
<td>26</td>
<td>p41</td>
<td>21</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2. Anomaly detection through computation of $1 - f_w^{RT_p}(rt)$:
Activities of Fault Localization (1/2)

After detection of a failure at time $t_a$:

1. Determination of response times, prefixes and workload intensities (for each execution) for the time period $[t_a - \delta, t_a]$:

<table>
<thead>
<tr>
<th>O</th>
<th>TraceID</th>
<th>$t_{in}$</th>
<th>$t_{out}$</th>
<th>$t_{out} - t_{in}$</th>
<th>p</th>
<th>w</th>
</tr>
</thead>
<tbody>
<tr>
<td>Catalog.getBook(..)</td>
<td>121</td>
<td>1182</td>
<td>1201</td>
<td>19</td>
<td>p17</td>
<td>17</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bookshop.query(..)</td>
<td>131</td>
<td>1195</td>
<td>1221</td>
<td>26</td>
<td>p41</td>
<td>21</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2. Anomaly detection through computation of $1 - f_{RT_p}^w (rt)$:

<table>
<thead>
<tr>
<th>O</th>
<th>TraceID</th>
<th>$t_{in}$</th>
<th>$t_{out}$</th>
<th>$t_{out} - t_{in}$</th>
<th>p</th>
<th>w</th>
<th>$1 - f_{RT_p}^w$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Catalog.getBook(..)</td>
<td>121</td>
<td>1182</td>
<td>1201</td>
<td>19</td>
<td>p2</td>
<td>17</td>
<td>$1 - f_{RT_p}^{17} (19) = 0.75$</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bookshop.query(..)</td>
<td>131</td>
<td>1195</td>
<td>1221</td>
<td>26</td>
<td>p41</td>
<td>21</td>
<td>$1 - f_{RT_p}^{21} (17) = 0.21$</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Activities of fault localization (2/2)

3 Anomaly analysis: Aggregation of many anomaly values
   - Mean degree of anomaly for each operation / component / deployment context
   - Analysis of anomalies in combination with component dependency graphs
   - Neural networks [Stransky, 2006]
   - Event correlation techniques [Steinder and Sethi, 2004]
Activities of fault localization (2/2)

3 Anomaly analysis: Aggregation of many anomaly values
   - Mean degree of anomaly for each operation / component / deployment context
   - Analysis of anomalies in combination with component dependency graphs
   - Neural networks [Stransky, 2006]
   - Event correlation techniques [Steinder and Sethi, 2004]

4 Presentation of results (diagnosis report)

\[
P_{ft} = 0.08
\]

\[
P_{ft} = 0.8
\]

\[
P_{ft} = 0.12
\]
Outline

1. Foundations
2. Creation of the timing behavior profile
3. Fault Localization
4. Evaluation
   - Lab studies
   - Field studies
5. Related work
6. Conclusions
Evaluation goals:
- Proof of concept: Failure diagnosis for injected faults
- Efficiency of anomaly detection and anomaly analysis

Setting:
- Generation of artificial (probabilistic) system usage
- Fault injection
- Example applications:
  - Sun Java PetStore Demo Application, (and reimplementations)
  - (Rubis Benchmark)
  - TPC-App Benchmark
Evaluation – Field studies

Evaluation goals:

- Applicability in real world systems
  - Complex system usage
  - Long execution sequences
- Effectiveness of anomaly detection
Evaluation goals:

- Applicability in real world systems
  - Complex system usage
  - Long execution sequences
- Effectiveness of anomaly detection

Field study in progress:

- Evaluation of 12 month timing behavior data from a customer portal of a middle-size telecommunication company (only highly aggregated response times)
Evaluation – Field studies

Evaluation goals:
- Applicability in real world systems
  - Complex system usage
  - Long execution sequences
- Effectiveness of anomaly detection

Field study in progress:
- Evaluation of 12 month timing behavior data from a customer portal of a middle-size telecommunication company (only highly aggregated response times)

Field studies in preparation:
- Telecommunication system of Siemens
Evaluation goals:
- Applicability in real world systems
  - Complex system usage
  - Long execution sequences
- Effectiveness of anomaly detection

Field study in progress:
- Evaluation of 12 month timing behavior data from a customer portal of a middle-size telecommunication company (only highly aggregated response times)

Field studies in preparation:
- Telecommunication system of Siemens
- E-learning management platform StudIP
Related work

Failure diagnosis based on analysis of timing behavior:

Failure diagnosis based on analysis of (component) execution sequences:

Failure diagnosis based on multiple runtime behavior metrics:
Related work

Failure diagnosis based on analysis of timing behavior:
- [Agarwal et al., 2004] Response time analysis in the context of average historic response times and SLA violations
- [Diaconescu and Murphy, 2005]: Anomalies as violations of relative threshold values (based on historic average)

Failure diagnosis based on analysis of (component) execution sequences:

Failure diagnosis based on multiple runtime behavior metrics:
Related work

Failure diagnosis based on analysis of timing behavior:
- [Agarwal et al., 2004] Response time analysis in the context of average historic response times and SLA violations
- [Diaconescu and Murphy, 2005]: Anomalies as violations of relative threshold values (based on historic average)

Failure diagnosis based on analysis of (component) execution sequences:
- [Kiciman and Fox, 2005; Kiciman, 2005; Aguilera et al., 2003; Barham et al., 2004]: Component interaction probabilities and component dependency graphs

Failure diagnosis based on multiple runtime behavior metrics:
Related work

Failure diagnosis based on analysis of timing behavior:

- [Agarwal et al., 2004] Response time analysis in the context of average historic response times and SLA violations
- [Diaconescu and Murphy, 2005]: Anomalies as violations of relative threshold values (based on historic average)

Failure diagnosis based on analysis of (component) execution sequences:

- [Kiciman and Fox, 2005; Kiciman, 2005; Aguilera et al., 2003; Barham et al., 2004]: Component interaction probabilities and component dependency graphs

Failure diagnosis based on multiple runtime behavior metrics:

- [Cohen et al., 2005]: Monitoring and evaluation of 62 platform metrics for failure diagnosis, response times of diagnosis of performance problems
- [Salfner and Malek, 2005]: Prediction of failures based on runtime behavior monitoring (for rejuvenation)
Conclusions

- New approach to the detection of timing behavior anomalies for the localization of faults
- Improvement of timing behavior analysis:
  - Workload intensity awareness
  - Awareness of service demand characteristics
- Anomaly detection is used to increase availability and reliability of enterprise-scale software systems
Conclusions

- New approach to the detection of timing behavior anomalies for the localization of faults
- Improvement of timing behavior analysis:
  - Workload intensity awareness
  - Awareness of service demand characteristics
- Anomaly detection is used to increase availability and reliability of enterprise-scale software systems
- Empirical evaluation requires much effort (fault injection & complex usage)
M. K. Agarwal, K. Appleby, M. Gupta, G. Kar, A. Neogi, and A. Sailer. Problem
determination using dependency graphs and run-time behavior models. In
15th IFIP/IEEE International Workshop on Distributed Systems: Operations
and Management (DSOM’04), volume 3278 of Lecture Notes in Computer

Performance debugging for distributed systems of black boxes. In SOSP
’03: Proceedings of the nineteenth ACM symposium on Operating systems

A. Avižienis, J.-C. Laprie, B. Randell, and C. Landwehr. Basic concepts and
taxonomy of dependable and secure computing. IEEE Transactions on
doi:10.1109/TDSC.2004.2.

extraction and workload modelling. In 6th Symposium On Operating

Capturing, indexing, clustering, and retrieving system history. In SOSP ’05:


