Production-Run Software Failure Diagnosis via Adaptive Communication Tracking

Mejbah ul Alam, Abdullah Muzahid
University of Texas at San Antonio
Production-run software failures are widespread and expensive
Production-run software failures are widespread and expensive
Quick failure analysis

I see no bug here!

How to reproduce the bug?

Need to analyze the new code
ACT
Hardware Neural Network for Failure Diagnosis

void foo{
    x++;  
    bar();
}

W1->R1, W2->R2, W2->R3, ... ...

Track Data Communication

Bug!
Communication Tracking for Failure Analysis

Bug in MySQL

thd->proc_info = ...
if(thd->proc_info) {
  ...
  fputs(thd->proc_info,f)
}

thd->proc_info = NULL

W1 -> R1, W1 -> R2
Communication Tracking for Failure Analysis

Bug in MySQL

```
if(thd->proc_info > proc_info)
{
    ... ...
    fputs(thd->proc_info,f)
}
```

```
thd->proc_info = NULL
```

W1 -> R1, W1 -> R2
W2 -> R1
Communication Tracking for Failure Analysis

Bug in MySQL

```
if(thd->proc_info > proc_info)
{
    ... ... ... 
    fputs(thd->proc_info, f)
}
```

```
... ... ... 
```

```
T1
```

```
W1
```

```
W2
```

```
T2
```

```
thd->proc_info = NULL
```

```
W1 -> R1, W1 -> R2
W2 -> R1
W1 -> R1, W2 -> R2
```
How ACT Works

Software

Offline Training

Postprocessing

Hardware

Online Testing

Online Training
Offline Training

- Collection of execution traces
- Sequence of Read After Write (RAW) dependences

- Input vector of N RAW dependences
- Train different topologies

Diagram:
- Program
  - Binary Instrumentation
    - Execution Traces
    - Neural Network Library
      - Neural network Topology & Weights

Online Testing

ACT: Failure Diagnosis via Comm. Tracking
Online Testing

### Input Layer

```
if (thd->proc_info > proc_info)
{
    ... ...
    fputs(thd->proc_info, f);
}
```

### Hidden Layer

- W1 -> R1
- W1 -> R2

### Output Layer

```
\omega_n, \omega_{n+1}, \omega_{n+4}
```

**ACT: Failure Diagnosis via Comm. Tracking**
Online Testing

```
if (thd->proc_info > proc_info) {
    ... ... ...
    fputs(thd->proc_info, f)
    thd->proc_info = NULL
}
```

ACT: Failure Diagnosis via Comm. Tracking
Online Testing

```
if(thd->proc_info) {
    ... ...
    fputs(thd->proc_info, f)
}
```

```
thd->proc_info = ...
R1
R2
```

```
W1 -> R1, W2 -> R2
```

```
W1 -> R1, W1 -> R2
```

```
T1
T2
```

```
Input Layer
Hidden Layer
Output Layer
```

```
w_i
w_{n+1}
w_{n+4}
```
Online Testing

```c
if (thd->proc_info) {
    ... ...
    puts(thd->proc_info, f);
}
```

**Input Layer** ➔ **Hidden Layer** ➔ **Output Layer**

- T1: thd->proc_info = ...
- T2: ... ...
- R1: thd->proc_info = NULL
- R2: ... ...

**Bug!**
Online Testing

ACT: Failure Diagnosis via Comm. Tracking
Online Testing

Input Layer → Hidden Layer → Output Layer

Invalid Counter: 0 0 0 1

Debug Buffer: W2→R2

ACT: Failure Diagnosis via Comm. Tracking
Adaptive to Change in Code

Online Testing  > Threshold
                  Invalid Counter

< Threshold
Invalid Counter

Online Training
Postprocessing: Offline Pruning and Ranking

• Postprocess Invalid Communication Sequences in after a failure:

  - **Rank** recorded entries
  - Compare with a **Correct Set**
  - Number of matched RAW dependences higher $\rightarrow$ higher rank
ACT Implementation

Reorder Buffer

Controller

Invalid Counter

Neural Network

Mode

Debug Buffer

Input Generator Buffer

Last Writer Instruction

Cache

Mejbah ul Alam

ACT: Failure Diagnosis via Comm. Tracking
Pipelined Implementation of Neural Network

Input

Pipeline Register

Hidden Layer

Pipeline Register

Output

\[ N \]

\[ N_1 \]

\[ N_2 \]

\[ N_m \]

\[ N_{out} \]
Pipelined Implementation of Neural Network

Mejbah ul Alam

ACT: Failure Diagnosis via Comm. Tracking
### Experimental Setup

<table>
<thead>
<tr>
<th>Parameters for Neuron</th>
<th>Parameters for ACT Module</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max input</td>
<td>Total neuron</td>
</tr>
<tr>
<td>10</td>
<td>11</td>
</tr>
<tr>
<td>Multiply-add unit</td>
<td>Input generator Buffer</td>
</tr>
<tr>
<td>1, 2, 5, 10 with 1 cycle latency</td>
<td>5 entries</td>
</tr>
<tr>
<td>Accumulator</td>
<td>Debug buffer</td>
</tr>
<tr>
<td>1 cycle latency</td>
<td>600 entries</td>
</tr>
<tr>
<td>Sigmoid unit</td>
<td>Misprediction threshold</td>
</tr>
<tr>
<td>1 cycle latency</td>
<td>5%</td>
</tr>
<tr>
<td>Input FIFO</td>
<td></td>
</tr>
<tr>
<td>4, 8, 16</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameters for System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Architecture</td>
</tr>
<tr>
<td>Coherence</td>
</tr>
</tbody>
</table>
Experimental Setup

• Applications for evaluation:
  ▪ SPLASH2, PARSEC, SPECT INT 2006, GNU Coreutil

• Real world bugs:
  ▪ Apache, MySQL, Memcached, Pbzip2, Aget, GNU Coreutil
## Diagnosis of Existing Bugs

<table>
<thead>
<tr>
<th>Bug</th>
<th>Bug Description</th>
<th>Rank</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aget</td>
<td>Ordering violation on <code>bwritten</code></td>
<td>4</td>
</tr>
<tr>
<td>Apache</td>
<td>Atomicity vio. on ref. counter</td>
<td>7</td>
</tr>
<tr>
<td>Memcached</td>
<td>Atomicity vio. on item data</td>
<td>2</td>
</tr>
<tr>
<td>MySQL#1</td>
<td>Atomicity vio. loss of logged data</td>
<td>1</td>
</tr>
<tr>
<td>MySQL#2</td>
<td>Atomicity vio. thd-&gt;proc_info</td>
<td>1</td>
</tr>
<tr>
<td>MySQL#3</td>
<td>Atomicity vio. on join_init_cache</td>
<td>3</td>
</tr>
<tr>
<td>PBzip22</td>
<td>Ordering vio. Between threads</td>
<td>1</td>
</tr>
<tr>
<td>Gzip</td>
<td>Semantic bug</td>
<td>1</td>
</tr>
<tr>
<td>seq</td>
<td>Semantic bug</td>
<td>5</td>
</tr>
<tr>
<td>ptx</td>
<td>Buffer overflow</td>
<td>1</td>
</tr>
<tr>
<td>paste</td>
<td>Reads out of buffer string</td>
<td>8</td>
</tr>
</tbody>
</table>
# Diagnosis of Existing Bugs

<table>
<thead>
<tr>
<th>Bug</th>
<th>Bug Description</th>
<th>Rank</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aget</td>
<td>Ordering violation on <em>bwritten</em></td>
<td>4</td>
</tr>
<tr>
<td>Apache</td>
<td>Atomicity vio. on ref. counter</td>
<td>7</td>
</tr>
<tr>
<td>Memcached</td>
<td>Atomicity vio. on item data</td>
<td>2</td>
</tr>
<tr>
<td>MySQL#1</td>
<td>Atomicity vio. loss of logged data</td>
<td>1</td>
</tr>
<tr>
<td>MySQL#2</td>
<td>Atomicity vio. thd-&gt;proc_info</td>
<td>1</td>
</tr>
<tr>
<td>MySQL#3</td>
<td>Atomicity vio. on join_init_cache</td>
<td>3</td>
</tr>
<tr>
<td>PBzip22</td>
<td>Ordering vio. Between threads</td>
<td>1</td>
</tr>
<tr>
<td>Gzip</td>
<td>Semantic bug</td>
<td>1</td>
</tr>
<tr>
<td>seq</td>
<td>Semantic bug</td>
<td>5</td>
</tr>
<tr>
<td>ptx</td>
<td>Buffer overflow</td>
<td>1</td>
</tr>
<tr>
<td>paste</td>
<td>Reads out of buffer string</td>
<td>8</td>
</tr>
</tbody>
</table>
## Diagnosis of Existing Bugs

<table>
<thead>
<tr>
<th>Bug</th>
<th>Bug Description</th>
<th>Rank</th>
<th>AVISO Rank</th>
<th>PBI Rank</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aget</td>
<td>Ordering violation on <code>bwritten</code></td>
<td>4</td>
<td>3</td>
<td>-</td>
</tr>
<tr>
<td>Apache</td>
<td>Atomicity vio. on ref. counter</td>
<td>7</td>
<td>-</td>
<td>136</td>
</tr>
<tr>
<td>Memcached</td>
<td>Atomicity vio. on item data</td>
<td>2</td>
<td>34</td>
<td>6</td>
</tr>
<tr>
<td>MySQL#1</td>
<td>Atomicity vio. loss of logged data</td>
<td>1</td>
<td>628</td>
<td>222</td>
</tr>
<tr>
<td>MySQL#2</td>
<td>Atomicity vio. thd-&gt;proc_info</td>
<td>1</td>
<td>294</td>
<td>153</td>
</tr>
<tr>
<td>MySQL#3</td>
<td>Atomicity vio. on join_init_cache</td>
<td>3</td>
<td>122</td>
<td>-</td>
</tr>
<tr>
<td>PBzip22</td>
<td>Ordering vio. Between threads</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Gzip</td>
<td>Semantic bug</td>
<td>1</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>seq</td>
<td>Semantic bug</td>
<td>5</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>ptx</td>
<td>Buffer overflow</td>
<td>1</td>
<td>-</td>
<td>1</td>
</tr>
<tr>
<td>paste</td>
<td>Reads out of buffer string</td>
<td>8</td>
<td>-</td>
<td>4</td>
</tr>
</tbody>
</table>
Diagnosis of Injected Bugs in New Code

<table>
<thead>
<tr>
<th>Program</th>
<th>Function</th>
<th>Rank</th>
</tr>
</thead>
<tbody>
<tr>
<td>fft</td>
<td>Touch-Array</td>
<td>7</td>
</tr>
<tr>
<td>fmm</td>
<td>VListInter-action</td>
<td>23</td>
</tr>
<tr>
<td>fluidanimate</td>
<td>ComputeDensitiesMT</td>
<td>18</td>
</tr>
<tr>
<td>lu</td>
<td>TouchA</td>
<td>1</td>
</tr>
<tr>
<td>swaptions</td>
<td>Worker</td>
<td>6</td>
</tr>
</tbody>
</table>
Experiments & Results

Execution Overhead - Combined Mode

- 10 Mul-Add
- 5 Mul-Add
- 2 Mul-Add
- 1 Mul-Add

8.2%
Conclusion

• We proposed **ACT**, the **first adaptive** approach based on Hardware Neural Network
• It provides more **accurate ranking** of root causes
• It does not need to **reproduce failure**
• Average execution overhead **8.2%**