FastTrack: Efficient and Precise Dynamic Race Detection

Stephen Freund
Williams College

Cormac Flanagan
UC Santa Cruz

FastTrack
• Design Criteria:
  - sound
  - complete
  - efficient
• Insight: Accesses to a var are almost always totally ordered in HB relation

Dynamic Race Detection

Happens Before [Lamport 78]

- Track locks held on all accesses to var.
- Empty lock set implies possible race.
- Unsound & Incomplete
  (Some Trace Generalization)

Eraser [SBN+ 97]

Vector Clocks [M 88]
Goldilocks [EQT 07]
Djit+ [ISZ 99, PS 03]
Trade [CB 01]
...

Barriers [PS 03]
Initialization [vPG 01]
...

RaceTrack [YRC 05]
MultiRace [PS 03]
Hybrid Race Detector [OC 03]
...

Dynamic Race Detection

Happens Before [Lamport 78]

- Compute partial order of operations
- Ensure conflicting operations are not concurrent
- Sound & Complete
  (No Trace Generalization)

Eraser [SBN+ 97]

Vector Clocks [M 88]
Goldilocks [EQT 07]
Djit+ [ISZ 99, PS 03]
Trade [CB 01]
...

Barriers [PS 03]
Initialization [vPG 01]
...

RaceTrack [YRC 05]
MultiRace [PS 03]
Hybrid Race Detector [OC 03]
...

Dynamic Race Detection

Happens Before [Lamport 78]

- Compute partial order of operations
- Ensure conflicting operations are not concurrent
- Sound & Complete
  (No Trace Generalization)

Eraser [SBN+ 97]

Vector Clocks [M 88]
Goldilocks [EQT 07]
Djit+ [ISZ 99, PS 03]
Trade [CB 01]
...

Barriers [PS 03]
Initialization [vPG 01]
...

RaceTrack [YRC 05]
MultiRace [PS 03]
Hybrid Race Detector [OC 03]
...

Dynamic Race Detection

Happens Before [Lamport 78]

- Compute partial order of operations
- Ensure conflicting operations are not concurrent
- Sound & Complete
  (No Trace Generalization)

Eraser [SBN+ 97]

Vector Clocks [M 88]
Goldilocks [EQT 07]
Djit+ [ISZ 99, PS 03]
Trade [CB 01]
...

Barriers [PS 03]
Initialization [vPG 01]
...

RaceTrack [YRC 05]
MultiRace [PS 03]
Hybrid Race Detector [OC 03]
...

Dynamic Race Detection

Happens Before [Lamport 78]

- Compute partial order of operations
- Ensure conflicting operations are not concurrent
- Sound & Complete
  (No Trace Generalization)

Eraser [SBN+ 97]

Vector Clocks [M 88]
Goldilocks [EQT 07]
Djit+ [ISZ 99, PS 03]
Trade [CB 01]
...

Barriers [PS 03]
Initialization [vPG 01]
...

RaceTrack [YRC 05]
MultiRace [PS 03]
Hybrid Race Detector [OC 03]
...

Dynamic Race Detection

Happens Before [Lamport 78]

- Compute partial order of operations
- Ensure conflicting operations are not concurrent
- Sound & Complete
  (No Trace Generalization)

Eraser [SBN+ 97]

Vector Clocks [M 88]
Goldilocks [EQT 07]
Djit+ [ISZ 99, PS 03]
Trade [CB 01]
...

Barriers [PS 03]
Initialization [vPG 01]
...

RaceTrack [YRC 05]
MultiRace [PS 03]
Hybrid Race Detector [OC 03]
...

Dynamic Race Detection

Happens Before [Lamport 78]

- Compute partial order of operations
- Ensure conflicting operations are not concurrent
- Sound & Complete
  (No Trace Generalization)

Eraser [SBN+ 97]

Vector Clocks [M 88]
Goldilocks [EQT 07]
Djit+ [ISZ 99, PS 03]
Trade [CB 01]
...

Barriers [PS 03]
Initialization [vPG 01]
...

RaceTrack [YRC 05]
MultiRace [PS 03]
Hybrid Race Detector [OC 03]
...
Happens-Before

- Event Ordering:
  - program order
  - synchronization order

- Types of Races:
  - Write-Write
  - Write-Read
    - (write before read)
  - Read-Write
    - (read before write)

Thread A

Thread B

x = 0

\( \text{rel}(m) \)

acq(m)

y = x

\( \text{Race} \)

O(n) time

VC_A

VC_B

L_m

W_x

R_x

\[
\begin{array}{cccc}
4 & 1 & 2 & 8 \\
\text{A} & \text{B} & \text{A} & \text{B} \\
\end{array}
\]

\[
\begin{array}{cccc}
2 & 1 & 3 & 0 & 0 & 1 \\
2 & 1 & 4 & 0 & 0 & 1 \\
\end{array}
\]

Write-Write Check: \( W_x \subseteq VC_A \) ?

\[
\begin{array}{cc}
3 & 0 \\
4 & 1 \\
\end{array}
\]

Yes

Read-Write Check: \( R_x \subseteq VC_A \) ?

\[
\begin{array}{cc}
0 & 1 \\
4 & 1 \\
\end{array}
\]

Yes

x = 0

rel(m)

acq(m)

x = 1

VC_A

VC_B

L_m

W_x

R_x

\[
\begin{array}{cccc}
4 & 1 & 2 & 8 \\
\text{A} & \text{B} & \text{A} & \text{B} \\
\end{array}
\]

\[
\begin{array}{cccc}
2 & 1 & 4 & 0 & 0 & 1 \\
2 & 1 & 4 & 0 & 0 & 1 \\
\end{array}
\]
Write-Write Check: \( W_x \subseteq VC_A \)?
\[
\begin{array}{c}
4 \ 1 \\
\text{rel}(m) \\
5 \ 1 \\
\text{acq}(m) \\
5 \ 1 \\
4 \ 8 \\
5 \ 1 \\
4 \ 8 \\
\end{array}
\]

No

\( O(n) \) time
Write-Read Check: $W_x \subseteq VC_x$?

$\begin{array}{c}
\text{ VC}_x \\
\hline
4 & 1 \\
\end{array}$

$x = 0$

$\begin{array}{c}
\text{ VC}_y \\
\hline
0 & 8 \\
\end{array}$

$\begin{array}{c}
\text{ L}_m \\
\hline
0 & 0 \\
\end{array}$

$\begin{array}{c}
\text{ W}_x \\
\hline
3 & @A \\
\end{array}$

$\begin{array}{c}
\text{ acq} (m) \\
\hline
4 & 8 \\
\end{array}$

$x = 1$

$\begin{array}{c}
\text{ rel} (m) \\
\hline
5 & 1 \\
\end{array}$

$\begin{array}{c}
\text{ acq} (m) \\
\hline
4 & 8 \\
\end{array}$

$\begin{array}{c}
\text{ rel} (m) \\
\hline
5 & 1 \\
\end{array}$

$\begin{array}{c}
\text{ W}_x \\
\hline
8 & @B \\
\end{array}$

$\begin{array}{c}
\text{ VC}_x \\
\hline
4 & 1 \\
\end{array}$

$x = 0$

$\begin{array}{c}
\text{ VC}_y \\
\hline
0 & 8 \\
\end{array}$

$\begin{array}{c}
\text{ L}_m \\
\hline
0 & 0 \\
\end{array}$

$\begin{array}{c}
\text{ W}_x \\
\hline
4 & @A \\
\end{array}$

$\begin{array}{c}
\text{ acq} (m) \\
\hline
4 & 1 \\
\end{array}$

$x = 1$

$\begin{array}{c}
\text{ rel} (m) \\
\hline
5 & 1 \\
\end{array}$

$\begin{array}{c}
\text{ acq} (m) \\
\hline
4 & 8 \\
\end{array}$

$\begin{array}{c}
\text{ rel} (m) \\
\hline
5 & 1 \\
\end{array}$

$\begin{array}{c}
\text{ W}_x \\
\hline
8 & @B \\
\end{array}$

$\begin{array}{c}
\text{ VC}_x \\
\hline
4 & 1 \\
\end{array}$

$x = 0$

$\begin{array}{c}
\text{ VC}_y \\
\hline
0 & 8 \\
\end{array}$

$\begin{array}{c}
\text{ L}_m \\
\hline
0 & 0 \\
\end{array}$

$\begin{array}{c}
\text{ W}_x \\
\hline
4 & @A \\
\end{array}$

$\begin{array}{c}
\text{ acq} (m) \\
\hline
4 & 1 \\
\end{array}$

$x = 1$

$\begin{array}{c}
\text{ rel} (m) \\
\hline
5 & 1 \\
\end{array}$

$\begin{array}{c}
\text{ acq} (m) \\
\hline
4 & 8 \\
\end{array}$

$\begin{array}{c}
\text{ rel} (m) \\
\hline
5 & 1 \\
\end{array}$

$\begin{array}{c}
\text{ W}_x \\
\hline
8 & @B \\
\end{array}$

Read-Write Check:

- **Ordered Reads**
  - Thread A
  - Thread B
  - Thread C
  - Thread D

- **Unordered Reads**
  - Thread A
  - Thread B
  - Thread C

Most common case: thread-local, lock-protected, ...

Read-Write Check:

- $R_x \subseteq VC_A$?

- $R_x \not\subseteq VC_A$? No
Validation

- Six race condition checkers
  - all use RoadRunner
  - share common components (eg, VectorClock)
  - profiled and optimized
- Further optimization opportunities
  - unsound extensions, dynamic escape analysis, static analysis, implement inside JVM, hardware support, ...
- 15 Benchmarks
  - 250 KLOC
  - locks, wait/notify, fork/join, barriers, ...

RoadRunner Architecture

Standard JVM

Instrumented Bytecode

Event Stream

Back-End Checker

Error: race on x...

Warnings

22 false positives
3 false negatives
**Slowdown (x Base Time)**

<table>
<thead>
<tr>
<th></th>
<th>Empty</th>
<th>Eraser</th>
<th>MultiRace</th>
<th>Goldlocks</th>
<th>Basic VC</th>
<th>DJIT+</th>
<th>FastTrack</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>4.1</td>
<td>8.6</td>
<td>21.7</td>
<td>31.6</td>
<td>89.8</td>
<td>20.2</td>
<td>8.5</td>
</tr>
</tbody>
</table>

**O(n) Vector Clock Operations**

- 96.4% of all ops are Reads/Writes
- R/W ops requiring O(n) time:
  - Basic VC: 100%
  - DJIT+: 26.0%
  - FastTrack: <0.1%

**Memory Usage**

- FastTrack allocated ~200x fewer VCs

<table>
<thead>
<tr>
<th>Checker</th>
<th>Memory Overhead</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic VC</td>
<td>7.9x</td>
</tr>
<tr>
<td>DJIT+</td>
<td></td>
</tr>
<tr>
<td>FastTrack</td>
<td>2.8x</td>
</tr>
</tbody>
</table>

(Note: VCs for dead objects can be garbage collected)

**Eclipse 3.4**

- Scale
  - > 6,000 classes
  - 24 threads
  - custom sync. idioms

- Precision (tested 5 common tasks)
  - Eraser: ~1000 warnings
  - FastTrack: ~30 warnings

- Performance on compute-bound tasks
  - > 2x speed of other precise checkers
  - same as Eraser