Automatically Comparing Memory Consistency Models

John Wickerson  Mark Batty  Tyler Sorensen  George A. Constantinides
Imperial  U Kent  Imperial  Imperial

S-REPLS @ Imperial
Tuesday 27 September 2016
Contents

• Context: memory consistency models (MCMs)
• Where our work fits in
• Key Ideas
• Applications
Contents

Context: memory consistency models (MCMs)

• Where our work fits in
• Key Ideas
• Applications
Relaxed-memory behaviours

\[
\begin{align*}
x &= y = 0; \\
x &= 1; & y &= 1; \\
r0 &= y; & r1 &= x;
\end{align*}
\]
Relaxed-memory behaviours

\[
\begin{align*}
    x &= y = 0; \\
    x &= 1; \\
    r0 &= y; \\
    y &= 1; \\
    r1 &= x; \\
    r0 &= 0, r1 &= 1
\end{align*}
\]
Relaxed-memory behaviours

\[
\begin{align*}
  x &= y = 0; \\
  x &= 1; \\
  r0 &= y; \\
  y &= 1; \\
  r1 &= x; \\
  r0 &= 0, \ r1 = 1 \\
  r0 &= 1, \ r1 = 1
\end{align*}
\]
Relaxed-memory behaviours

\[
\begin{align*}
x &= y = 0; \\
x &= 1; & y &= 1; \\
r0 &= y; & r1 &= x; \\
r0 &= 0, r1 = 1 \\
r0 &= 1, r1 = 1 \\
r0 &= 1, r1 = 0
\end{align*}
\]
Relaxed-memory behaviours

\[
x = y = 0; \\
x = 1; \quad y = 1; \quad r0 = y; \quad r1 = x; \\
r0 = 0, \ r1 = 0 \\
r0 = 0, \ r1 = 1 \\
r0 = 1, \ r1 = 1 \\
r0 = 1, \ r1 = 0
\]
Much confusion!

Subtleties related to relaxed memory have led to bugs in...

• **programming language specifications** [Batty+ POPL’11, Batty+ ESOP’13],

• **deployed processors** [Alglave+ CAV’10],

• **compilers** [Morisset+ PLDI’13, Sevcik+ ECOOP’08], and

• **vendor-endorsed programming guides** [Alglave+ ASPLOS’15].
Axiomatic models

\begin{align*}
x &= 1; & y &= 1; \\
r_0 &= y; & r_1 &= x;
\end{align*}
Axiomatic models

\[
\begin{align*}
    x &= 1; \\
    r_0 &= y; \\
    y &= 1; \\
    r_1 &= x;
\end{align*}
\]
Axiomatic models

\[
x = 1; \quad \text{or} \quad y = 1; \\
r_0 = y; \quad \text{or} \quad r_1 = x;
\]
Axiomatic models

\[
\begin{align*}
x &= 1; \\
r_0 &= y; \\
r_1 &= x;
\end{align*}
\]
Axiomatic models

\[ x = 1; \quad r_0 = y; \quad r_1 = x; \]

\[ y = 1; \]
Axiomatic models

\[ x = 1; \quad r_0 = y; \]
\[ y = 1; \quad r_1 = x; \]
Axiomatic models

\[
\begin{align*}
x &= 1; \\
r_0 &= y; \\
r_1 &= x; \\
y &= 1; \\
x &= 1; \\
r_0 &= y; \\
r_1 &= x;
\end{align*}
\]
Axiomatic models

\[ x = 1; \quad y = 1; \quad r_0 = y; \quad r_1 = x; \]
Axiomatic models

\[ x = 1; \quad r_0 = y; \]
\[ y = 1; \quad r_1 = x; \]
Axiomatic models

\[ \begin{align*}
x &= 1; \\
r_0 &= y; \\
r_1 &= x;
\end{align*} \]
Axiomatic models

\[ x = 1; \quad y = 1; \quad r0 = y; \quad r1 = x; \]

\[ W \ x=1 \quad R \ y=1 \quad W \ y=1 \quad R \ x=0 \]

\[ W \ x=1 \quad R \ y=0 \quad W \ y=1 \quad R \ x=1 \]

\[ W \ x=1 \quad R \ y=0 \quad W \ y=1 \quad R \ x=0 \]

\[ SC \ ✔ \]

\[ SC \ × \]
Axiomatic models

\[
x = 1; \\
y = 1; \\
r_0 = y; \\
r_1 = x;
\]
Axiomatic models

\[
x = 1; \quad y = 1; \quad r_0 = y; \quad r_1 = x;
\]

x86 ✓ SC ✓

x86 ✓ SC ×
Contents

• Context: memory consistency models (MCMs)

Where our work fits in

• Key Ideas

• Applications
Some challenges and current approaches to tackling them

<table>
<thead>
<tr>
<th>Question</th>
<th>Existing work</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Some challenges and current approaches to tackling them

<table>
<thead>
<tr>
<th>Question</th>
<th>Existing work</th>
</tr>
</thead>
<tbody>
<tr>
<td>Can a given litmus test pass under a given MCM?</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Some challenges and current approaches to tackling them

<table>
<thead>
<tr>
<th>Question</th>
<th>Existing work</th>
</tr>
</thead>
<tbody>
<tr>
<td>Can a given litmus test pass under a given MCM?</td>
<td>CppMem, Herd, MemSAT, Nemos, ...</td>
</tr>
</tbody>
</table>
Some challenges and current approaches to tackling them

<table>
<thead>
<tr>
<th>Question</th>
<th>Existing work</th>
</tr>
</thead>
<tbody>
<tr>
<td>Can a given litmus test pass under a given MCM?</td>
<td>CppMem, Herd, MemSAT, Nemos, ...</td>
</tr>
<tr>
<td>Which litmus tests can be run to check whether a machine conforms to a given MCM?</td>
<td></td>
</tr>
</tbody>
</table>
Some challenges and current approaches to tackling them

<table>
<thead>
<tr>
<th>Question</th>
<th>Existing work</th>
</tr>
</thead>
<tbody>
<tr>
<td>Can a given litmus test pass under a given MCM?</td>
<td>CppMem, Herd, MemSAT, Nemos, ...</td>
</tr>
<tr>
<td>Which litmus tests can be run to check whether a machine conforms to</td>
<td>semi-automatic generation with DIY [Alglave+ CAV'10];</td>
</tr>
<tr>
<td>a given MCM?</td>
<td></td>
</tr>
</tbody>
</table>
Some challenges and current approaches to tackling them

<table>
<thead>
<tr>
<th>Question</th>
<th>Existing work</th>
</tr>
</thead>
<tbody>
<tr>
<td>Can a given litmus test pass under a given MCM?</td>
<td>CppMem, Herd, MemSAT, Nemos, ...</td>
</tr>
<tr>
<td>Which litmus tests can be run to check whether a machine conforms to a given MCM?</td>
<td>semi-automatic generation with DIY [Alglave+ CAV'10];</td>
</tr>
<tr>
<td>Is one MCM more permissive than another?</td>
<td></td>
</tr>
</tbody>
</table>
Some challenges and current approaches to tackling them

<table>
<thead>
<tr>
<th>Question</th>
<th>Existing work</th>
</tr>
</thead>
<tbody>
<tr>
<td>Can a given litmus test pass under a given MCM?</td>
<td>CppMem, Herd, MemSAT, Nemos, ...</td>
</tr>
<tr>
<td>Which litmus tests can be run to check whether a machine conforms to a given MCM?</td>
<td>semi-automatic generation with DIY [Alglave+ CAV'10];</td>
</tr>
<tr>
<td>Is one MCM more permissive than another?</td>
<td>manual proof; manual examples [Batty+ POPL'16]; semi-automatic checking with DIY+Herd;</td>
</tr>
</tbody>
</table>
Some challenges and current approaches to tackling them

<table>
<thead>
<tr>
<th>Question</th>
<th>Existing work</th>
</tr>
</thead>
<tbody>
<tr>
<td>Can a given litmus test pass under a given MCM?</td>
<td>CppMem, Herd, MemSAT, Nemos, ...</td>
</tr>
<tr>
<td>Which litmus tests can be run to check whether a machine conforms to a given MCM?</td>
<td>semi-automatic generation with DIY [Alglave+ CAV'10]; manual proof; manual examples [Batty+ POPL'16]; semi-automatic checking with DIY+Herd;</td>
</tr>
<tr>
<td>Is one MCM more permissive than another?</td>
<td></td>
</tr>
<tr>
<td>Does my MCM allow a given compiler optimisation?</td>
<td></td>
</tr>
</tbody>
</table>
Some challenges and current approaches to tackling them

<table>
<thead>
<tr>
<th>Question</th>
<th>Existing work</th>
</tr>
</thead>
<tbody>
<tr>
<td>Can a given litmus test pass under a given MCM?</td>
<td>CppMem, Herd, MemSAT, Nemos, ...</td>
</tr>
<tr>
<td>Which litmus tests can be run to check whether a machine conforms to a given MCM?</td>
<td>semi-automatic generation with DIY [Alglave+ CAV'10];</td>
</tr>
<tr>
<td>Is one MCM more permissive than another?</td>
<td>manual proof; manual examples [Batty+ POPL'16];</td>
</tr>
<tr>
<td></td>
<td>semi-automatic checking with DIY+Herd;</td>
</tr>
<tr>
<td>Does my MCM allow a given compiler optimisation?</td>
<td>manual c'examples [Vafeiaidis+ POPL'15];</td>
</tr>
<tr>
<td></td>
<td>manual proof [Sevcik PLDI'11];</td>
</tr>
</tbody>
</table>
Some challenges and current approaches to tackling them

<table>
<thead>
<tr>
<th>Question</th>
<th>Existing work</th>
</tr>
</thead>
<tbody>
<tr>
<td>Can a given litmus test pass under a given MCM?</td>
<td>CppMem, Herd, MemSAT, Nemos, ...</td>
</tr>
<tr>
<td>Which litmus tests can be run to check whether a machine conforms to a given MCM?</td>
<td>semi-automatic generation with DIY [Alglave+ CAV'10];</td>
</tr>
<tr>
<td>Is one MCM more permissive than another?</td>
<td>manual proof; manual examples [Batty+ POPL'16]; semi-automatic checking with DIY+Herd;</td>
</tr>
<tr>
<td>Does my MCM allow a given compiler optimisation?</td>
<td>manual c'examples [Vafeiaidis+ POPL'15]; manual proof [Sevcik PLDI'11];</td>
</tr>
<tr>
<td>Does my MCM allow a given compiler mapping?</td>
<td></td>
</tr>
</tbody>
</table>
Some challenges and current approaches to tackling them

<table>
<thead>
<tr>
<th>Question</th>
<th>Existing work</th>
</tr>
</thead>
<tbody>
<tr>
<td>Can a given litmus test pass under a given MCM?</td>
<td>CppMem, Herd, MemSAT, Nemos, ...</td>
</tr>
<tr>
<td>Which litmus tests can be run to check whether a machine conforms to a given MCM?</td>
<td>semi-automatic generation with DIY [Alglave+ CAV'10];</td>
</tr>
<tr>
<td>Is one MCM more permissive than another?</td>
<td>manual proof; manual examples [Batty+ POPL'16]; semi-automatic checking with DIY+Herd;</td>
</tr>
<tr>
<td>Does my MCM allow a given compiler optimisation?</td>
<td>manual c'examples [Vafeiaidis+ POPL'15]; manual proof [Sevcik PLDI'11];</td>
</tr>
<tr>
<td>Does my MCM allow a given compiler mapping?</td>
<td>manual c'examples [Wickerson+ OOPSLA'15]; manual proof [Batty+ POPL'11, Batty+ POPL'12];</td>
</tr>
</tbody>
</table>
## Our contributions

<table>
<thead>
<tr>
<th>Question</th>
<th>Existing work</th>
</tr>
</thead>
<tbody>
<tr>
<td>Can a given litmus test pass under a given MCM?</td>
<td>CppMem, Herd, MemSAT, Nemos, ...</td>
</tr>
<tr>
<td>Which litmus tests can be run to check whether a machine conforms to a</td>
<td>semi-automatic generation with DIY [Alglave+ CAV'10];</td>
</tr>
<tr>
<td>given MCM?</td>
<td></td>
</tr>
<tr>
<td>Is one MCM more permissive than another?</td>
<td>manual proof; manual examples [Batty+ POPL'16];</td>
</tr>
<tr>
<td></td>
<td>semi-automatic checking with DIY+Herd;</td>
</tr>
<tr>
<td>Does my MCM allow a given compiler optimisation?</td>
<td>manual c'examples [Vafeiaidis+ POPL'15];</td>
</tr>
<tr>
<td></td>
<td>manual proof [Sevcik PLDI'11];</td>
</tr>
<tr>
<td>Does my MCM allow a given compiler mapping?</td>
<td>manual c'examples [Wickerson+ OOPSLA'15];</td>
</tr>
<tr>
<td></td>
<td>manual proof [Batty+ POPL'11, Batty+ POPL'12];</td>
</tr>
</tbody>
</table>
# Our contributions

<table>
<thead>
<tr>
<th>Question</th>
<th>Existing work</th>
</tr>
</thead>
<tbody>
<tr>
<td>Can a given litmus test pass under a given MCM?</td>
<td>CppMem, Herd, MemSAT, Nemos, ...</td>
</tr>
<tr>
<td>Which litmus tests can be run to check whether a machine conforms to a given MCM?</td>
<td>semi-automatic generation with DIY [Alglave+ CAV'10];</td>
</tr>
<tr>
<td></td>
<td><strong>automatic generation</strong></td>
</tr>
<tr>
<td>Is one MCM more permissive than another?</td>
<td>manual proof; manual examples [Batty+ POPL'16];</td>
</tr>
<tr>
<td></td>
<td>semi-automatic checking with DIY+Herd;</td>
</tr>
<tr>
<td>Does my MCM allow a given compiler optimisation?</td>
<td>manual c'examples [Vafeiaidis+ POPL'15];</td>
</tr>
<tr>
<td></td>
<td>manual proof [Sevcik PLDI'11];</td>
</tr>
<tr>
<td>Does my MCM allow a given compiler mapping?</td>
<td>manual c'examples [Wickerson+ OOPSLA'15];</td>
</tr>
<tr>
<td></td>
<td>manual proof [Batty+ POPL'11, Batty+ POPL'12];</td>
</tr>
</tbody>
</table>
## Our contributions

<table>
<thead>
<tr>
<th>Question</th>
<th>Existing work</th>
</tr>
</thead>
<tbody>
<tr>
<td>Can a given litmus test pass under a given MCM?</td>
<td>CppMem, Herd, MemSAT, Nemos, ...</td>
</tr>
<tr>
<td>Which litmus tests can be run to check whether a machine conforms to a given MCM?</td>
<td>semi-automatic generation with DIY [Alglave+ CAV'10];</td>
</tr>
<tr>
<td></td>
<td><strong>automatic generation</strong></td>
</tr>
</tbody>
</table>
| Is one MCM more permissive than another?                                                                                              | manual proof; manual examples [Batty+ POPL'16];  
|                                                                                                                                        | semi-automatic checking with DIY+Herd;                                                           |
|                                                                                                                                        | **automatic checking**                                                                             |
| Does my MCM allow a given compiler optimisation?                                                                                      | manual c'examples [Vafeiaidis+ POPL'15];  
|                                                                                                                                        | manual proof [Sevcik PLDI'11];                                                                   |
| Does my MCM allow a given compiler mapping?                                                                                           | manual c'examples [Wickerson+ OOPSLA'15];  
|                                                                                                                                        | manual proof [Batty+ POPL'11, Batty+ POPL'12];                                                   |
## Our contributions

<table>
<thead>
<tr>
<th>Question</th>
<th>Existing work</th>
</tr>
</thead>
<tbody>
<tr>
<td>Can a given litmus test pass under a given MCM?</td>
<td>CppMem, Herd, MemSAT, Nemos, ...</td>
</tr>
<tr>
<td>Which litmus tests can be run to check whether a machine conforms to a given MCM?</td>
<td>semi-automatic generation with DIY [Alglave+ CAV'10]; <strong>automatic generation</strong></td>
</tr>
<tr>
<td>Is one MCM more permissive than another?</td>
<td>manual proof; manual examples [Batty+ POPL'16]; semi-automatic checking with DIY+Herd; <strong>automatic checking</strong></td>
</tr>
<tr>
<td>Does my MCM allow a given compiler optimisation?</td>
<td>manual c'examples [Vafeiaidis+ POPL'15]; manual proof [Sevcik PLDI'11]; <strong>automatic checking</strong></td>
</tr>
<tr>
<td>Does my MCM allow a given compiler mapping?</td>
<td>manual c'examples [Wickerson+ OOPSLA'15]; manual proof [Batty+ POPL'11, Batty+ POPL'12];</td>
</tr>
</tbody>
</table>
Our contributions

<table>
<thead>
<tr>
<th>Question</th>
<th>Existing work</th>
</tr>
</thead>
<tbody>
<tr>
<td>Can a given litmus test pass under a given MCM?</td>
<td>CppMem, Herd, MemSAT, Nemos, ...</td>
</tr>
<tr>
<td>Which litmus tests can be run to check whether a machine conforms to a given MCM?</td>
<td>semi-automatic generation with DIY [Alglave+ CAV'10]; automatic generation</td>
</tr>
<tr>
<td>Is one MCM more permissive than another?</td>
<td>manual proof; manual examples [Batty+ POPL'16]; semi-automatic checking with DIY+Herd; automatic checking</td>
</tr>
<tr>
<td>Does my MCM allow a given compiler optimisation?</td>
<td>manual c'examples [Vafeiaidis+ POPL'15]; manual proof [Sevcik PLDI'11]; automatic checking</td>
</tr>
<tr>
<td>Does my MCM allow a given compiler mapping?</td>
<td>manual c'examples [Wickerson+ OOPSLA'15]; manual proof [Batty+ POPL'11, Batty+ POPL'12]; automatic checking</td>
</tr>
</tbody>
</table>
Contents

- Context: memory consistency models (MCMs)
- Where our work fits in
- Key Ideas
  - Applications
Key Idea 1
Key Idea 1

• What are M's conformance tests?
  Find \((P, \sigma)\) where \(\sigma \notin \text{obs}_M(P)\) and \(\sigma \in \text{obs}_0(P)\).
Key Idea 1

• What are $M$'s conformance tests?
  Find $(P, \sigma)$ where $\sigma \notin \text{obs}_M(P)$ and $\sigma \in \text{obs}_0(P)$.

• Is $M$ stronger than $N$?
  No if $\exists (P, \sigma)$ where $\sigma \notin \text{obs}_N(P)$ and $\sigma \in \text{obs}_M(P)$. 
Key Idea 1

• What are $\mathbf{M}$'s conformance tests?
  Find $(P, \sigma)$ where $\sigma \notin \text{obs}_M(P)$ and $\sigma \in \text{obs}_0(P)$.

• Is $\mathbf{M}$ stronger than $\mathbf{N}$?
  No if $\exists (P, \sigma)$ where $\sigma \notin \text{obs}_N(P)$ and $\sigma \in \text{obs}_M(P)$.

• Does $\mathbf{M}$ allow my optimisation?
  No if $\exists (P, Q, \sigma)$ where $\sigma \notin \text{obs}_M(P)$, $\sigma \in \text{obs}_M(Q)$ and $P$ optimises to $Q$. 
Key Idea 1

• What are $M$'s conformance tests?
  Find $(P,\sigma)$ where $\sigma \not\in \text{obs}_M(P)$ and $\sigma \in \text{obs}_0(P)$.

• Is $M$ stronger than $N$?
  No if $\exists (P,\sigma)$ where $\sigma \not\in \text{obs}_N(P)$ and $\sigma \in \text{obs}_M(P)$.

• Does $M$ allow my optimisation?
  No if $\exists (P,Q,\sigma)$ where $\sigma \not\in \text{obs}_M(P)$, $\sigma \in \text{obs}_M(Q)$ and $P$ optimises to $Q$.

• Can $M$ be implemented by my mapping to $N$?
  No if $\exists (P,Q,\sigma)$ where $\sigma \not\in \text{obs}_M(P)$, $\sigma \in \text{obs}_N(Q)$ and $P$ compiles to $Q$. 
Key Idea 1

• What are $M$'s conformance tests?
  Find $(P,\sigma)$ where $\sigma \notin \text{obs}_M(P)$ and $\sigma \in \text{obs}_0(P)$.

• Is $M$ stronger than $N$?
  No if $\exists (P,\sigma)$ where $\sigma \notin \text{obs}_N(P)$ and $\sigma \in \text{obs}_M(P)$.

• Does $M$ allow my optimisation?
  No if $\exists (P,Q,\sigma)$ where $\sigma \notin \text{obs}_M(P)$, $\sigma \in \text{obs}_M(Q)$ and $P$ optimises to $Q$.

• Can $M$ be implemented by my mapping to $N$?
  No if $\exists (P,Q,\sigma)$ where $\sigma \notin \text{obs}_M(P)$, $\sigma \in \text{obs}_N(Q)$ and $P$ compiles to $Q$.

$$\{(P,Q,\sigma) \mid \sigma \notin \text{obs}_M(P) \land \sigma \in \text{obs}_N(Q) \land P \xrightarrow{\text{P}} Q\}$$
Key Idea 2

\{ (P, Q, \sigma) \mid \sigma \notin \text{obs}_M(P) \land \sigma \in \text{obs}_N(Q) \land P \triangleright Q \}
Key Idea 2

\{ (P,Q,\sigma) \mid \sigma \notin \text{obs}_M(P) \land \sigma \in \text{obs}_N(Q) \land P \triangleright Q \}
Key Idea 2

\{ (P,Q,\sigma) \mid \sigma \notin \text{obs}_M(P) \land \sigma \in \text{obs}_N(Q) \land P \triangleright Q \}
Key Idea 2

\{ (P,Q,\sigma) \mid \sigma \not\in \text{obs}_M(P) \land \sigma \in \text{obs}_N(Q) \land P \rightarrow Q \}
Key Idea 2

\[\{(P, Q, \sigma) \mid \sigma \notin \text{obs}_M(P) \land \sigma \in \text{obs}_N(Q) \land P \xRightarrow{} Q\}\]
**Key Idea 2**

\[
\{(P,Q,\sigma) \mid \sigma \not\in \text{obs}_M(P) \land \sigma \in \text{obs}_N(Q) \land P \triangleright Q\}
\]
Key Idea 2

\{(P,Q,\sigma) \mid \sigma \notin \text{obs}_M(P) \land \sigma \in \text{obs}_N(Q) \land P \triangleright Q\}
Key Idea 2

\{ (P, Q, \sigma) \mid \sigma \notin \text{obs}_M(P) \land \sigma \in \text{obs}_N(Q) \land P \triangleright Q \}
Key Idea 2

\{ (P,Q,\sigma) \mid \sigma \notin \text{obs}_M(P) \land \sigma \in \text{obs}_N(Q) \land P \triangleright Q \}
Key Idea 2

\{(P,Q,\sigma) \mid \sigma \not\in \text{obs}_M(P) \land \sigma \in \text{obs}_N(Q) \land P \rhd Q\}
Key Idea 2

\[
\{(P, Q, \sigma) \mid \sigma \notin \text{obs}_M(P) \land \sigma \in \text{obs}_N(Q) \land P \triangleright Q\}
\]
Key Idea 2

\[
\{(P,Q,\sigma) \mid \sigma \notin \text{obs}_M(P) \land \sigma \in \text{obs}_N(Q) \land P \triangleright Q\}
\]
Key Idea 2

\{(P,Q,\sigma) \mid \sigma \not\in \text{obs}_M(P) \land \sigma \in \text{obs}_N(Q) \land P \triangleright Q\}
Key Idea 2

\{(P,Q,\sigma) \mid \sigma \notin \text{obs}_M(P) \land \sigma \in \text{obs}_N(Q) \land P \rhd Q\}
Key Idea 2

\{(P,Q,σ) \mid σ \not\in \text{obs}_M(P) \land σ \in \text{obs}_N(Q) \land P \triangleright Q\}
The Alloy Constraint Solver

```alloy
open .../sw/c11_nafence[E] as M1
open .../sw/c11_simp[E] as M2

sig E {}

pred gp [X : Exec_C] {

  // Prefer solutions with total sb per thread
  total_sb[X]

  // ignore RMWs
  no_RMWs[X]

  // The execution is forbidden in M1
  not(M1/consistent[X])
  //M1/dead[X]

  // The execution is allowed (and not faulty)
  M2/consistent[X]
}

run gp for 1 Exec, 5 E, 3 Int
```
Contents

- Context: memory consistency models (MCMs)
- Where our work fits in
- Key Ideas
- Applications
Comparing "strong release-acquire" to original release-acquire
Comparing "strong release-acquire" to original release-acquire

Taming Release-Acquire Consistency

Ori Lahav  Nick Giannarakis  Viktor Vafeiadis

Institute for Software Systems (MPI-SWS), Germany
Comparing Nienhuis et al.'s C++ variant to the original

- Cf. Nienhuis et al. (OOPSLA '16):
Comparing Nienhuis et al.'s C++ variant to the original

• Cf. Nienhuis et al. (OOPSLA '16):

\[
\begin{align*}
C_{RLX} y &\quad 4/5 \\
\downarrow &\quad sb \\
F_{REL} &
\end{align*}
\]

\[
\begin{align*}
C_{REL} y &\quad 2/3 \\
\downarrow &\quad sb \\
W_{RLX} y &\quad 4
\end{align*}
\]

\[
\begin{align*}
C_{REL} x &\quad 2/3 \\
\downarrow &\quad sb \\
W_{SC} x &\quad 4
\end{align*}
\]

\[
\begin{align*}
C_{REL} x &\quad 4/5 \\
\downarrow &\quad sb \\
F_{AR} &
\end{align*}
\]

\[
\begin{align*}
C_{RLX} y &\quad 4/5 \\
\downarrow &\quad sb \\
W_{RLX} x &\quad 1
\end{align*}
\]

\[
\begin{align*}
C_{REL} y &\quad 2/3 \\
\downarrow &\quad sb \\
W_{RLX} y &\quad 4
\end{align*}
\]

\[
\begin{align*}
C_{REL} x &\quad 2/3 \\
\downarrow &\quad sb \\
W_{SC} x &\quad 4
\end{align*}
\]

\[
\begin{align*}
C_{REL} x &\quad 4/5 \\
\downarrow &\quad sb \\
F_{AR} &
\end{align*}
\]

\[
\begin{align*}
C_{RLX} y &\quad 4/5 \\
\downarrow &\quad sb \\
W_{RLX} x &\quad 1
\end{align*}
\]
Comparing Batty et al.'s C++ variant to the original

Overhauling SC Atomics in C11 and OpenCL

Mark Batty
University of Kent, UK
m.j.batty@kent.ac.uk

Alastair F. Donaldson
Imperial College London, UK
alastair.donaldson@imperial.ac.uk

John Wickerson
Imperial College London, UK
j.wickerson@imperial.ac.uk
Comparing Batty et al.'s C++ variant to the original

Overhauling SC Atomics in C11 and OpenCL

Mark Batty
University of Kent, UK
m.j.batty@kent.ac.uk

Alastair F. Donaldson
Imperial College London, UK
alastair.donaldson@imperial.ac.uk

John Wickerson
Imperial College London, UK
j.wickerson@imperial.ac.uk

atomic_int x=0,y=0;
x.store(1,RLX);

r0=x.cas(1,2,SC,RLX);
r1=y.load(SC);
y.store(1,SC);
r2=x.load(SC);
r0==true && r1==0 && r2==1

atomic_int x=0,y=0;

x.store(1,RLX);

r0=x.cas(1,2,SC,RLX);
r1=y.load(SC);
y.store(1,SC);
r2=x.load(SC);
r0==true && r1==0 && r2==1
Does C++ allow "linearisation"?
Does C++ allow "linearisation"?

Common Compiler Optimisations are Invalid in the C11 Memory Model and what we can do about it
Is AMD's OpenCL compiler mapping sound?
Is AMD's OpenCL compiler mapping sound?

\[ a: C_{AR, WG} x 0/1 \]

\[ b: W_{REL, DV, REM} x 2 \]

\[ \left( x \mapsto 0 \right) fet x \left( x \mapsto_{vd} 0 \right) \]

\[ \left( x \mapsto_{vc} 0 \right) C x 0/1 \left( x \mapsto_{vd} 1 \right) \]

\[ \left( x \mapsto_{vc} 1 \right) flu x \left( x \mapsto_{vo} 2 \right) \]

\[ \left( x \mapsto_{vo} 0 \right) Lk x \left( x \mapsto_{L} 0 \right) \]

\[ \left( x \mapsto_{vo} 2 \right) InvA \left( x \mapsto_{vo} 2 \right) \]

\[ \left( x \mapsto_{L} 0 \right) Flu \left( x \mapsto_{L} 0 \right) \]

\[ \left( x \mapsto_{L} 0 \right) W x 2 \left( x \mapsto_{L} 0 \right) \]

\[ \left( x \mapsto_{L} 0 \right) Uk x \left( x \mapsto_{L} 0 \right) \]
Checking and fixing an OpenCL/PTX compiler mapping

- PTX MCM proposed by Alglave et al. (ASPLOS '15)
- "Obvious" OpenCL/PTX mapping is invalid
- Manually revise PTX MCM (to obtain "PTX2")
- Now mapping is valid
- Run litmus tests that distinguish PTX/PTX2 against GPU hardware to validate PTX2
Contents

• Context: memory consistency models (MCMs)
• Where our work fits in
• Key Ideas
• Applications
Automatically Comparing Memory Consistency Models

John Wickerson  Mark Batty  Tyler Sorensen  George A. Constantinides
Imperial    U Kent    Imperial    Imperial

Newcastle University
Wednesday 31 August 2016