Herding cats II

Jade Alglave

Arm and University College London

March 2019

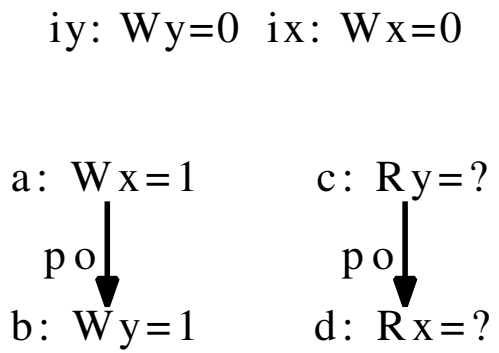
Remember our pony

Message passing

initially x=y=0

P_0	P_1
(a) x $\leftarrow$ 1	(c) r3 $\leftarrow$ y
(b) y $\leftarrow$ 1	(d) r4 $\leftarrow$ x

r3=1 and r4=0 allowed?

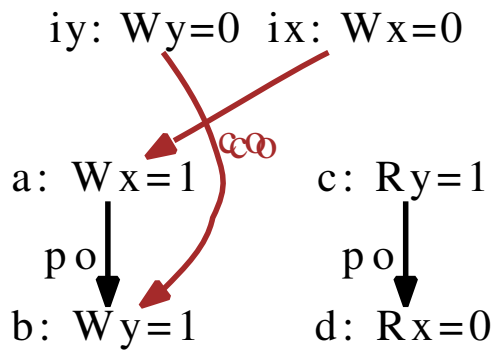


Message passing

initially x=y=0

P_0	P_1
(a) x $\leftarrow$ 1	(c) r3 $\leftarrow$ y
(b) y $\leftarrow$ 1	(d) r4 $\leftarrow$ x

r3=1 and r4=0 allowed?

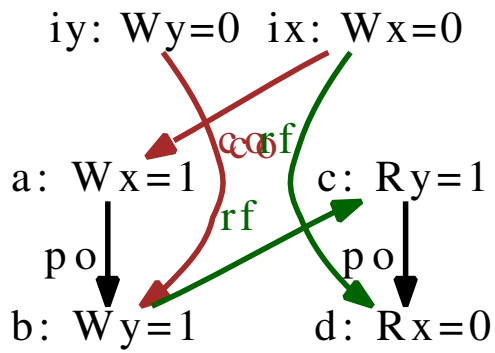


Message passing

initially x=y=0

P_0	P_1
(a) x $\leftarrow$ 1	(c) r3 $\leftarrow$ y
(b) y $\leftarrow$ 1	(d) r4 $\leftarrow$ x

r3=1 and r4=0 allowed?

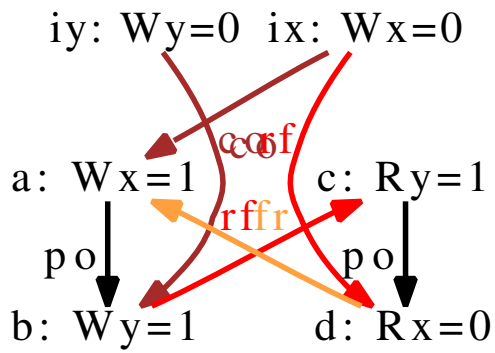


Message passing

initially x=y=0

P_0	P_1
(a) x $\leftarrow$ 1	(c) r3 $\leftarrow$ y
(b) y $\leftarrow$ 1	(d) r4 $\leftarrow$ x

r3=1 and r4=0 allowed?

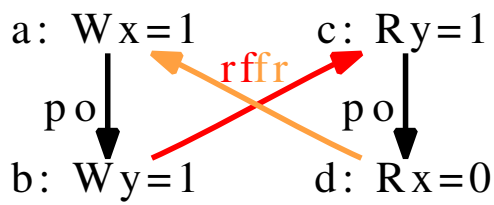


Message passing

initially x=y=0

P_0	P_1
(a) x $\leftarrow$ 1	(c) r3 $\leftarrow$ y
(b) y $\leftarrow$ 1	(d) r4 $\leftarrow$ x

r3=1 and r4=0 allowed?



Let's try it out

initially x=y=0

P_0	P_1
(a) x $\leftarrow$ 1	(c) r3 $\leftarrow$ y
(b) y $\leftarrow$ 1	(d) r4 $\leftarrow$ x

r3=1 and r4=0 allowed?

- on Intel x86 (see `herdtools7/herd/libdir/x86tso.cat`)
- on Armv8 (see `herdtools7/herd/libdir/aarch64.cat`)

MP on x86

```
X86 MP
{
}
 P0           | P1            ;
 MOV [x],$1   | MOV EAX,[y]   ;
 MOV [y],$1   | MOV EBX,[x]   ;
exists(1:EAX=1 /\ 1:EBX=0)
```

MP on x86

Do:

```
~/bin/herd7 -cat x86tso.cat mp-x86.litmus
Test MP Allowed
States 3
1:EAX=0; 1:EBX=0;
1:EAX=0; 1:EBX=1;
1:EAX=1; 1:EBX=1;
No
```

MP on Armv8

```
AArch64 MP
{
0:X1=x; 0:X3=y;
1:X1=y; 1:X3=x;
}
 P0            | P1            ;
 MOV W0,#1     | LDR W0,[X1]   ;
 STR W0,[X1]   | LDR W2,[X3]   ;
 MOV W2,#1     |               ;
 STR W2,[X3]   |               ;
exists(1:X0=1 /\ 1:X2=0)
```

MP on Armv8

Do:

```
~/bin/herd7 -cat aarch64.cat mp-armv8.litmus
Test MP Allowed
States 4
1:X0=0; 1:X2=0;
1:X0=0; 1:X2=1;
1:X0=1; 1:X2=0;
1:X0=1; 1:X2=1;
Ok
```

x86

- x86 implements a Total Store Order (TSO) model
- This can be modelled as two axioms:
  - SC per location
  - Total Store Order

Armv8

- Armv8 implements a multi-copy-atomic model
- It has two axioms:
  - SC per location (called Internal visibility by Arm)
  - External visibility

SC per location (Internal visibility)

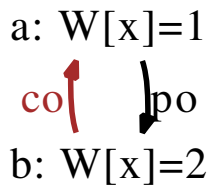
- Intuitively, this axiom means that the values that can be taken by a given location are as if on SC.
- In other words, weak memory concerns arise only when considering multiple locations.
- Thus this axiom ensures that non-relational program analyses are sound on weak memory systems.

SC per location (Internal visibility) in cat

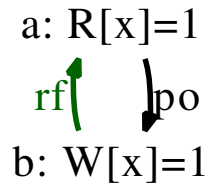
```
"SC per location"

include "cos.cat"
let po-loc = po & loc
let com = rf | fr | co
acyclic po-loc | com as sc-per-location
```

See also `herdtools7/herd/libdir/uniproc.cat`

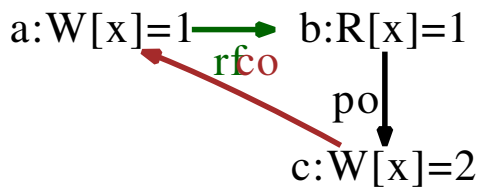
SC per location (Internal visibility)



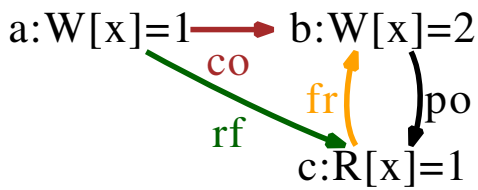
SC per location (Internal visibility)



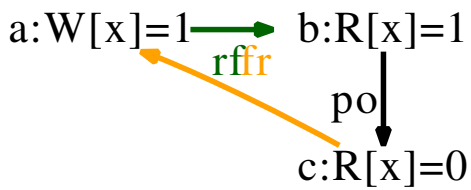
SC per location (Internal visibility)



SC per location (Internal visibility)



SC per location (Internal visibility)



That's it for today!