Deadlock-Free Monitors

Bart Jacobs (joint work with Jafar Hamin)

DistriNet, KU Leuven

IFIP WG 1.9/2.15 meeting @ Leuven, 11-12 May 2017

Contents

Contents

Programming Language

$$
\begin{array}{lcl}
e & ::= & v \mid x \\
c & ::= & e \mid \textbf{let } x := c \textbf{ in } c \\
 & & \mid \textbf{new}\{\overline{f := e}\} \mid e.f \mid e.f := e \\
 & & \mid \textbf{new lock} \mid \textbf{new channel} \\
 & & \mid \textbf{fork } c \\
 & & \mid e.\textbf{acquire}() \mid e.\textbf{release}() \\
 & & \mid e.\textbf{send}(e) \mid e.\textbf{receive}()
\end{array}
$$

Example Program: Producer-Consumer

```
let c := new channel() in
fork (
    c.receive();
    c.receive()
);
c.send(24);
c.send(42)
```

Example Proof: Producer-Consumer

$\{\mathsf{obs}(\mathbf{0})\}$
let c := **new channel**() **in**
$\{\mathsf{obs}(\mathbf{0})\}$
CreateObligation(c)
$\{\mathbf{obs}(\{\!|\mathsf{c}|\!\}) * \mathsf{c.credit}()\}$
CreateObligation(c)
$\{\mathbf{obs}(\{\!|2 \cdot \mathsf{c}|\!\}) * 2 \cdot \mathsf{c.credit}()\}$
fork (
  $\{\mathsf{obs}(\mathbf{0}) * 2 \cdot \mathsf{c.credit}()\}$
  c.**receive**();
  $\{\mathsf{obs}(\mathbf{0}) * \mathsf{c.credit}()\}$
  c.**receive**()
  $\{\mathsf{obs}(\mathbf{0})\}$
);
$\{\mathbf{obs}(\{\!|2 \cdot \mathsf{c}|\!\})\}$
c.**send**(24);
$\{\mathbf{obs}(\{\!|\mathsf{c}|\!\})\}$
c.**send**(42)
$\{\mathbf{obs}(\mathbf{0})\}$

Proof Rules

$$\frac{}{\{\mathsf{obs}(O)\}\ \textsc{CreateObligation}(c)\ \{\mathsf{obs}(O \uplus \{\!|c|\!\}) * c.\mathsf{credit}()\}}$$

$$\frac{\{\mathsf{obs}(O') * P\}\ c\ \{\mathsf{obs}(\mathbf{0})\}}{\{\mathsf{obs}(O \uplus O') * P\}\ \mathbf{fork}\ c\ \{\mathsf{obs}(O)\}}$$

$$\frac{}{\{\mathsf{obs}(O) * c.\mathsf{credit}() \wedge \mathsf{w}(c) \prec O\}\ c.\mathbf{receive}()\ \{\mathsf{obs}(O)\}}$$

$$\frac{}{\{\mathsf{obs}(O \uplus \{\!|c|\!\})\}\ c.\mathbf{send}(v)\ \{\mathsf{obs}(O)\}}$$

Example Program: Shared Counter

```
let c := new{x := 0} in
let lock := new lock in
let inc() = (
    lock.acquire();
    let v := c.x in
    c.x := v;
    lock.release()
) in
fork inc();
inc()
```

Example Proof: Shared Counter

$\{\mathsf{obs}(\mathbf{0})\}$
let c := **new**$\{\mathsf{x} := 0\}$ **in**
$\{\mathsf{obs}(\mathbf{0}) * \mathsf{c.x} \mapsto 0\}$
let lock := **new lock in**
$\{\mathsf{obs}(\mathbf{0}) * \mathsf{lock.lock}(\mathsf{c.x} \mapsto \_)\}$
let inc() = (
  $\{\mathsf{obs}(\mathbf{0}) * \mathsf{lock.lock}(\mathsf{c.x} \mapsto \_)\}$
  lock.**acquire**();
  $\{\mathsf{obs}(\{\!|\mathsf{lock}|\!\}) * \mathsf{lock.locked}(\mathsf{c.x} \mapsto \_) * \mathsf{c.x} \mapsto \_\}$
  let v := c.x **in**
  c.x := v;
  $\{\mathsf{obs}(\{\!|\mathsf{lock}|\!\}) * \mathsf{lock.locked}(\mathsf{c.x} \mapsto \_) * \mathsf{c.x} \mapsto \_\}$
  lock.**release**()
  $\{\mathsf{obs}(\mathbf{0}) * \mathsf{lock.lock}(\mathsf{c.x} \mapsto \_)\}$
) **in**
fork inc();
inc()

Proof Rules for Locks

$$\frac{}{\{I\}\ \textbf{new lock}\ \{\mathsf{res.lock}(I)\}} \qquad \ell.\mathsf{lock}(I) \Leftrightarrow \ell.\mathsf{lock}(I) * \ell.\mathsf{lock}(I)$$

$$\frac{}{\{\mathsf{obs}(O) * \ell.\mathsf{lock}(I) \wedge \mathsf{w}(\ell) \prec O\}\ \ell.\textbf{acquire}()\ \{\mathsf{obs}(O \uplus \{\!|\ell|\!\}) * \ell.\mathsf{locked}(I) * I\}}$$

$$\frac{}{\{\mathsf{obs}(O \uplus \{\!|\ell|\!\}) * \ell.\mathsf{locked}(I) * I\}\ \ell.\textbf{release}()\ \{\mathsf{obs}(O) * \ell.\mathsf{lock}(I)\}}$$

Soundness statement

$$\vdash \{\mathsf{obs}(\mathbf{0})\}\ c\ \{\mathsf{obs}(\mathbf{0})\} \Rightarrow (\mathbf{0}, \{\!|(\mathbf{0}, c)|\!\}) \rightsquigarrow^{*} \gamma \Rightarrow \neg(\gamma\ \mathsf{deadlocked})$$

Programming Language

$$
\begin{array}{rcl}
c & ::= & \cdots \mid \textbf{new monitor} \mid \textbf{new condvar} \\
& & \mid e.\textbf{enter}() \mid e.\textbf{exit}() \\
& & \mid e.\textbf{wait}() \mid e.\textbf{signal}() \mid e.\textbf{signalAll}()
\end{array}
$$

Example Program: Channels using monitors

```
let createChannel() :=
    let q := createQueue() in
    let m := new monitor() in
    let cv := new condvar() in
    new{q := q, m := m, cv := cv}
```

```
let send(c, v) :=
    c.m.enter();
    enqueue(c.q, v);
    c.cv.signal();
    c.m.exit()
```

```
let receive(c) :=
    c.m.enter();
    while queueSize(c.q) = 0 do
        c.cv.wait();
    let v := dequeue(c.q) in
    c.m.exit();
    v
```

Monitors: Proof Rules

$$\{\textbf{emp}\}\ \textbf{new monitor}\ \{\mathsf{res.monitor\_raw}(\emptyset)\}$$

$$\{m.\mathsf{monitor\_raw}(V)\}\ \textbf{new condvar}\ \{\mathsf{res} \notin V \wedge m.\mathsf{monitor\_raw}(V \cup \{\mathsf{res}\})\}$$

$$\frac{\forall (P, Q) \in \mathcal{S}, (P', Q') \in \mathcal{S}'.\ P * P' * \mathsf{stop\_perm}() \Rightarrow Q * Q'}{\{m.\mathsf{monitor\_raw}(V) * I(\lambda cv \in V.\ 0)\}\ \textsc{InitMonitor}\ \{m.\mathsf{monitor}(V, I, \mathcal{S}, \mathcal{S}')\}}$$

$$m.\mathsf{monitor}(V, I, \mathcal{S}, \mathcal{S}') \Rightarrow m.\mathsf{monitor}(V, I, \mathcal{S}, \mathcal{S}') * m.\mathsf{monitor}(V, I, \mathcal{S}, \mathcal{S}')$$

$\{m.\mathsf{obs}(O) * m.\mathsf{monitor}(V, I, \mathcal{S}, \mathcal{S}') \wedge \mathsf{w}(m) \prec O\}$
$m.\textbf{enter}()$
$\{m.\mathsf{obs}(O \uplus \{\!|m|\!\}) * m.\mathsf{monitor\_entered}(V, I, \mathcal{S}, \mathcal{S}', W) * I(W)\}$

$\{m.\mathsf{obs}(O \uplus \{\!|m|\!\}) * m.\mathsf{monitor\_entered}(V, I, \mathcal{S}, \mathcal{S}', W) * I(W)\}$
$m.\textbf{exit}()$
$\{m.\mathsf{obs}(O) * m.\mathsf{monitor}(V, I)\}$

Condition Variables: Proof Rules

$$\mathsf{obs}(O) \Leftrightarrow \mathsf{obs}(O \uplus \{\!|o|\!\}) * \mathsf{ob}(1, o) \qquad \mathsf{ob}(\pi_1 + \pi_2, o) \Leftrightarrow \mathsf{ob}(\pi_1, o) * \mathsf{ob}(\pi_2, o)$$

$$\mathsf{obs}(O) * \mathsf{ob}(\pi, o) \wedge o \prec O \Leftrightarrow \mathsf{obs\_calling}(O, \pi, o) * \mathsf{stop\_perm}()$$

$$\frac{(P, Q) \in \mathcal{S}}{\begin{array}{l}\{m.\mathsf{monitor\_entered}(V, I, \mathcal{S}, \mathcal{S}', W) * I(W[\mathsf{cv} := W(\mathsf{cv}) + 1]) * \mathsf{stop\_perm}() * P\} \\ cv.\mathbf{wait}() \\ \{\exists W'.\ m.\mathsf{monitor\_entered}(V, I, \mathcal{S}, \mathcal{S}', W') * I(W') * Q\}\end{array}}$$

$$\frac{(P', Q') \in \mathcal{S}'}{\begin{array}{l}\{m.\mathsf{monitor\_entered}(V, I, \mathcal{S}, \mathcal{S}', W) * P' \wedge W(cv) > 0\} \\ cv.\mathbf{signal}() \\ \{m.\mathsf{monitor\_entered}(V, I, \mathcal{S}, \mathcal{S}', W[\mathsf{cv} := W(\mathsf{cv}) - 1]) * Q'\}\end{array}}$$

Channels using monitors: Invariant

$$
\begin{aligned}
I(c)(W) = \exists \#elems, \#weakCredits, \#strongCredits, \#obs.\ & \\
& \mathsf{queue}(c.\mathsf{q}, \#elems) \\
& * \ c.\mathsf{wc} \hookrightarrow_\bullet \#weakCredits * c.\mathsf{sc} \hookrightarrow_\bullet \#strongCredits * c.\mathsf{o} \hookrightarrow_\bullet \#obs \\
& * \ (\#obs - W(c.\mathsf{cv})) \cdot \mathsf{ob}(1, c) \\
& \wedge \ W(c.\mathsf{cv}) + \#weakCredits + \#strongCredits = \#elems + \#obs \\
& \wedge \ (W(c.\mathsf{cv}) = 0 \vee W(c.\mathsf{cv}) \leq \#obs - \#strongCredits)
\end{aligned}
$$

$$
c.\mathsf{credit}() = c.\mathsf{sc} \hookrightarrow_\circ 1
$$

$$
\mathcal{S} = \{O.\ (\mathsf{obs\_calling}(O, 1, c), \mathsf{obs}(O) * c.\mathsf{wc} \hookrightarrow_\circ 1)\}
$$

$$
\mathcal{S}' = \{(c.\mathsf{wc} \hookrightarrow_\circ 1, \mathsf{ob}(1, c))\}
$$

Channels using monitors: Proof of receive

let receive(c) :=
$\{\mathsf{obs}(O) * \mathsf{channel}(\mathsf{c}) * \mathsf{c.credit}() \wedge \mathsf{w}(c) \prec O\}$
$\{\mathsf{obs}(O) * \mathsf{c.m.monitor}(V, I, \mathcal{S}, \mathcal{S}') * \mathsf{c.credit}() \wedge \mathsf{w}(c) \prec O\}$
c.m.**enter**();
$\{\mathsf{obs}(O) * \mathsf{c.m.mon\_entered}(V, I, \mathcal{S}, \mathcal{S}', W) * I(W) * \mathsf{c.credit}() \wedge \mathsf{w}(c) \prec O\}$
$\{\mathsf{obs}(O) * \mathsf{c.m.mon\_entered}(V, I, \mathcal{S}, \mathcal{S}', W) * I(W) * \mathsf{c.wc} \hookrightarrow_\circ 1 \wedge \mathsf{w}(c) \prec O\}$
while queueSize(c.q) $= 0$ **do**
$\{\mathsf{obs}(O) * \mathsf{c.m.mon\_entered}(V, I, \mathcal{S}, \mathcal{S}', W) * I(W+1) * \mathsf{ob}(1, \mathsf{c}) \wedge \mathsf{w}(c) \prec O\}$
$\{\mathsf{obs\_calling}(O, 1, \mathsf{c}) * \mathsf{c.m.mon\_entered}(V, I, \mathcal{S}, \mathcal{S}', W) * I(W+1) * \mathsf{stop\_perm}()$
c.cv.**wait**();
$\{\mathsf{obs}(O) * \mathsf{c.m.mon\_entered}(V, I, \mathcal{S}, \mathcal{S}', W') * I(W') * \mathsf{c.wc} \hookrightarrow_\circ 1\}$
let v := dequeue(c.q) **in**
c.m.**exit**();
v
$\{\mathsf{obs}(O)\}$

Channels using monitors: Proof of send

let send(c, v) :=
$\{\mathsf{channel}(c)\}$
$\{c.m.\mathsf{monitor}(V, I, \mathcal{S}, \mathcal{S}')\}$
c.m.**enter**();
$\{c.m.\mathsf{mon\_entered}(V, I, \mathcal{S}, \mathcal{S}', W) * I(W)\}$
enqueue(c.q, v);
Case $W > 0$
$\{c.m.\mathsf{mon\_entered}(V, I, \mathcal{S}, \mathcal{S}', W) * (\mathsf{ob}(1, c \twoheadrightarrow I(W-1)) * c.\mathsf{sc} \hookrightarrow_\circ 1 * c.\mathsf{wc} \hookrightarrow_\circ 1$
c.cv.**signal**();
$\{c.m.\mathsf{mon\_entered}(V, I, \mathcal{S}, \mathcal{S}', W-1) * I(W-1) * c.\mathsf{sc} \hookrightarrow_\circ 1\}$
c.m.**exit**()
$\{\mathsf{channel}(c) * c.\mathsf{credit}()\}$

Channels using monitors: Proof of obligation creation and destruction

let CreateObligation(c) :=
$\{\mathsf{obs}(O) * \mathsf{c.channel}()\}$
$\{\mathsf{obs}(O \uplus \{\!|\mathsf{c}|\!\}) * \mathsf{c.ob}() * \mathsf{c.credit}()\}$

let DestroyObligation(c) :=
$\{\mathsf{obs}(O \uplus \{\!|\mathsf{c}|\!\}) * \mathsf{c.ob}() * \mathsf{c.credit}()\}$
$\{\mathsf{obs}(O) * \mathsf{c.channel}()\}$