A Compositional Deadlock Detector for Android Java

James Brotherston[1] Paul Brunet[1] Nikos Gorogiannis[2]
Max Kanovich[1]

[1]UCL

[2]Facebook

IRIS meeting, Thurs 12th May, 2022

Problem overview

At Facebook, the aim was:

- to find deadlocks introduced by code revisions . . .
- . . . at code review time (< 15 mins) . . .
- . . . on thousands of revisions per day. . .
- . . . in codebases of tens of millions of LoC.

Deadlock analyses typically require the whole program. We cannot afford that.

A deadlock involves two (or more) traces. But a revision often affects only one trace.

Approach

- Compositional abstraction of each method;
- Sound and complete deadlock condition for abstract programs;
- Partial-program analysis of modified files in revision;
- Concurrency check — what methods may be run in parallel with Foo?
- Implementation in FB Infer (open source, OCaml).

Abstract programming language

We consider parallel compositions of sequential statements with procedure calls and balanced re-entrant locks:

$$\begin{aligned} C := \;& \texttt{skip} \mid p() \mid \texttt{acq}(\ell); C; \texttt{rel}(\ell) \mid C; C \\ & \mid \texttt{if}(*)\ \texttt{then}\ C\ \texttt{else}\ C \mid \texttt{while}(*)\ \texttt{do}\ C \end{aligned}$$

Program configurations are given by $\langle C, L\rangle$ where $L : \mathsf{Locks} \to \mathbb{N}$ is the lock state.

We write $\lfloor L \rfloor = \{\ell \in \mathsf{Locks} \mid L(\ell) > 0\}$ and $L \# L'$ to mean $\lfloor L_1 \rfloor \cap \lfloor L_2 \rfloor = \emptyset$. We write $\varnothing$ for the empty lock state sending all locks to 0.

Operational semantics

$$\langle \texttt{acq}(\ell), L\rangle \;\to\; \langle \texttt{skip}, L[\ell{+}{+}]\rangle \qquad (\texttt{acq})$$

$$\langle \texttt{rel}(\ell), L\rangle \;\to\; \langle \texttt{skip}, L[\ell{-}{-}]\rangle \; (L(\ell) > 0) \qquad (\texttt{rel})$$

$$\frac{\langle C_1, L_1\rangle \to \langle C_1', L_1'\rangle \qquad L_1' \,\#\, L_2}{\langle C_1 \;||\; C_2, (L_1, L_2)\rangle \rightsquigarrow \langle C_1 \;||\; C_2, (L_1', L_2)\rangle} \; (\texttt{par } 1)$$

$C_1 \;||\; C_2$ deadlocks if:

- $\langle C_1 \;||\; C_2, (\varnothing, \varnothing)\rangle \rightsquigarrow^* \langle C_1' \;||\; C_2', (L_1', L_2')\rangle$; and
- $\langle C_i', L_i'\rangle \to \langle C_i'', L_i''\rangle$ for each $i \in \{1, 2\}$; but
- there is no transition from $\langle C_1' \;||\; C_2', (L_1', L_2')\rangle$.

Critical pairs

(X, ℓ) is a critical pair of statement C if an execution of C acquires ℓ while holding the locks X:

$$\langle C, \varnothing \rangle \to^* \langle \mathtt{acq}(\ell); C', L \rangle \to \langle C', L[\ell{+}{+}] \rangle$$
$$\text{with } X = \lfloor L \rfloor \text{ and } \ell \notin X \ .$$

Theorem

For any statement C its set $\mathbf{Crit}(C)$ *of critical pairs is finite, and computable in* NP*-time (in the size of C).*

Critical pairs example

$$C_1 : \quad \texttt{acq}(x); \texttt{acq}(y); \texttt{skip}; \texttt{rel}(y); \texttt{rel}(x)$$
$$C_2 : \quad \texttt{acq}(y); \texttt{acq}(x); \texttt{skip}; \texttt{rel}(x); \texttt{rel}(y)$$

We have:

$$\mathbf{Crit}(C_1) = \{(\emptyset, x), (\{x\}, y)\}$$
$$\mathbf{Crit}(C_2) = \{(\emptyset, y), (\{y\}, x)\}$$

Now let $C_i' = \texttt{acq}(z); C_i'; \texttt{rel}(z)$, for $i \in \{1, 2\}$. Then:

$$\mathbf{Crit}(C_1) = \{(\emptyset, z), (\{z\}, x), (\{z, x\}, y)\}$$
$$\mathbf{Crit}(C_2) = \{(\emptyset, z), (\{z\}, y), (\{z, y\}, x)\}$$

Deadlock theorem

Theorem

$C_1 \mid\mid C_2$ *deadlocks iff* $(X_1, \ell_1) \in \mathbf{Crit}(C_1)$, $(X_2, \ell_2) \in \mathbf{Crit}(C_2)$ *with* $\ell_1 \in X_2$ *and* $\ell_2 \in X_1$ *and* $X_1 \cap X_2 = \emptyset$.

Example

$$C_1 : \quad \texttt{acq}(x); \texttt{acq}(y); \texttt{skip}; \texttt{rel}(y); \texttt{rel}(x)$$
$$C_2 : \quad \texttt{acq}(y); \texttt{acq}(x); \texttt{skip}; \texttt{rel}(x); \texttt{rel}(y)$$

$$\mathbf{Crit}(C_1) : \quad \{(\emptyset, x), (\{x\}, y)\}$$
$$\mathbf{Crit}(C_2) : \quad \{(\emptyset, y), (\{y\}, x)\}$$

Observe the condition is met and indeed $C_1 \mid\mid C_2$ deadlocks.

Deadlock theorem

Theorem

$C_1 \mid\mid C_2$ *deadlocks iff* $(X_1, \ell_1) \in \mathbf{Crit}(C_1)$, $(X_2, \ell_2) \in \mathbf{Crit}(C_2)$ *with* $\ell_1 \in X_2$ *and* $\ell_2 \in X_1$ *and* $X_1 \cap X_2 = \emptyset$.

Example

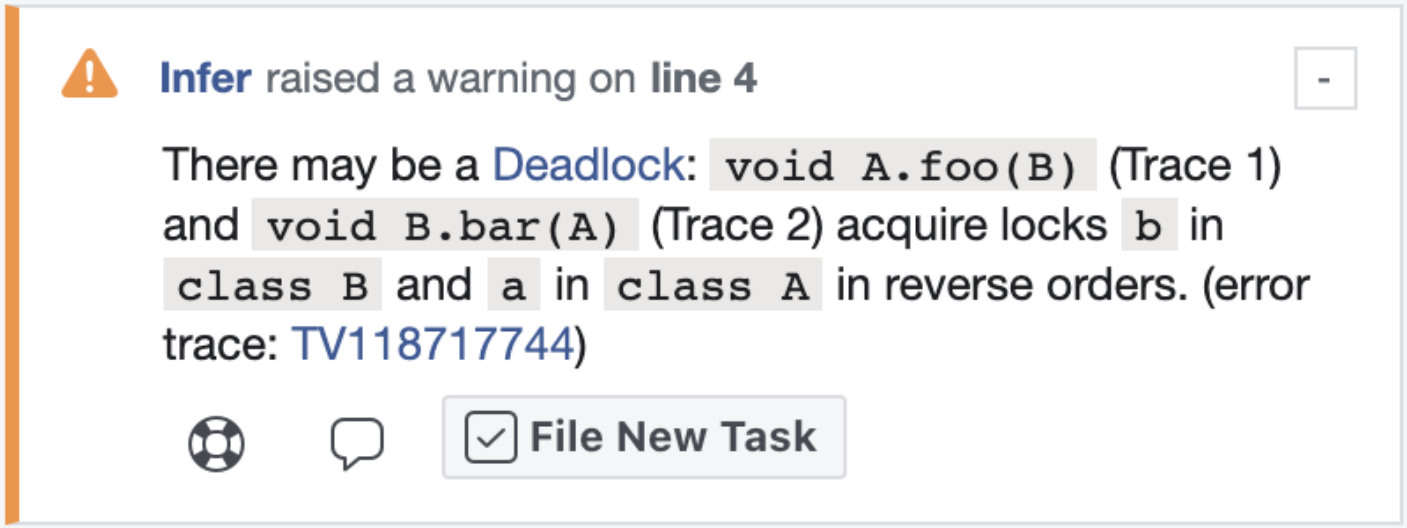
$$C_1' : \quad \texttt{acq}(z); \texttt{acq}(x); \texttt{acq}(y); \texttt{skip}; \texttt{rel}(y); \texttt{rel}(x); \texttt{rel}(z)$$
$$C_2' : \quad \texttt{acq}(z); \texttt{acq}(y); \texttt{acq}(x); \texttt{skip}; \texttt{rel}(x); \texttt{rel}(y); \texttt{rel}(z)$$

$$\mathbf{Crit}(C_1') : \quad \{(\emptyset, z), (\{z\}, x), (\{z, x\}, y)\}$$
$$\mathbf{Crit}(C_2') : \quad \{(\emptyset, z), (\{z\}, y), (\{z, y\}, x)\}$$

The condition is not met and indeed $C_1' \mid\mid C_2'$ does not deadlock.

Implementation

```
class A {
  public synchronized void foo(B b) { b.foo(); }
```




```
  public synchronized void bar() {}
}

class B {
  public synchronized void bar(A a) { a.bar(); }
  public synchronized void foo() {}
}
```

Impact

- Analysed $> 100k$ of revisions in > 2 years.
- Issued > 500 reports, with long traces.
- Fix rate is $> 50\%$.
- On average $\sim 5k$ methods analysed per revision.