Deadlock Checking by Behaviour Inference for Lock Handling

Ka I Violet Pun, Martin Steffen

PMA Group, University of Oslo, Norway

The 22nd Nordic Workshop for Programming Theory - NWPT '10
Turku, Finland

10 - 12 November, 2010

Agenda

1. Introduction
2. Syntax and Semantics
3. Type and Effect System
4. Abstract Behaviour
5. Summary

Four necessary conditions for a deadlock

- Mutual Exclusion
- Wait-for
- No-preemption
- Circular wait

Four necessary conditions for a deadlock

- Mutual Exclusion
- Wait-for
- No-preemption
- Circular wait

Deadlock

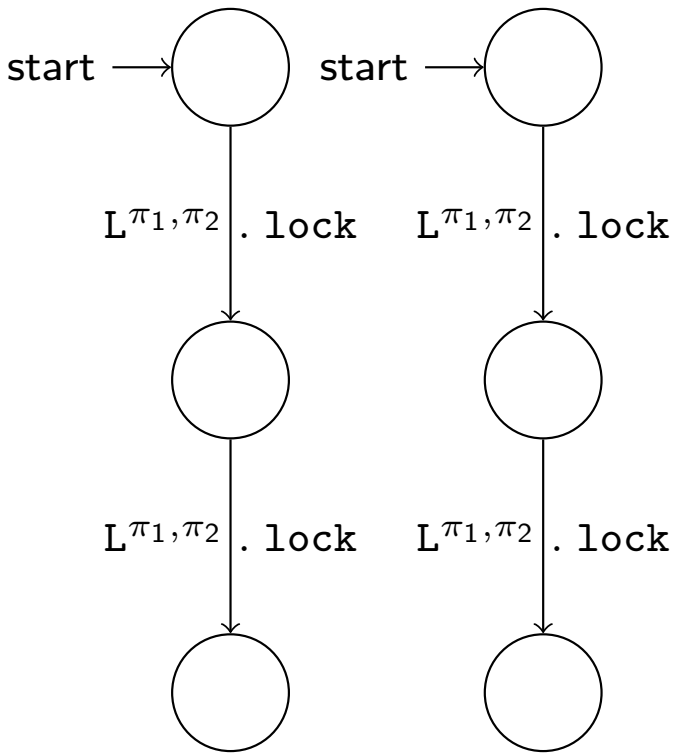
Each of two or more threads, which form a circular chain, wait for a lock that is held by the next thread in the chain.

Overview

- Find *potential* deadlocks in programs by detecting circular wait statically
  - Use *program points*, π, to characterize locks according to their origin
  - Calcuate abstract behaviour by effect inference
  - Execute the abstract behaviour
- Setting:
  - No lock creation in a loop/recursive function
  - Reentrant lock in a loop (recursive function), the lock counter is set to
    - ∞ (locking)
    - $\perp$ (unlocking)

Example

```
let t = fn (z1,z2). z1.lock;z2.lock;stop
in
    let x1 = new_π1 L in
    let x2 = new_π2 L in
    spawn (t (x1,x2)); t (x2,x1); stop
```



Syntax

$$
\begin{array}{lcl}
t & ::= & \texttt{stop} \mid v \mid \texttt{let}\ x{:}T = e\ \texttt{in}\ t \\
e & ::= & t \mid e\ e \mid \texttt{if}\ e\ \texttt{then}\ e\ \texttt{else}\ e \mid \texttt{spawn}\ e \mid \texttt{new}\ L \mid v.\,\texttt{lock} \\
 & \mid & v.\,\texttt{unlock} \\
v & ::= & x \mid l \mid \texttt{fn}\ x{:}T.t \mid \texttt{fun}\ f{:}T.x{:}T.t
\end{array}
$$

Sequential composition $e_1; e_2$ is represented by `let`-construct

$$\texttt{let}\ x{:}T = e_1\ \texttt{in}\ e_2\ , \qquad x \notin fv(e_2)$$

Syntax

$$
\begin{array}{lcl}
t & ::= & \texttt{stop} \mid v \mid \texttt{let}\ x{:}T = e\ \texttt{in}\ t \\
e & ::= & t \mid e\ e \mid \texttt{if}\ e\ \texttt{then}\ e\ \texttt{else}\ e \mid \texttt{spawn}\ e \mid \texttt{new}\ L \mid v.\,\texttt{lock} \\
 & \mid & v.\,\texttt{unlock} \\
v & ::= & x \mid l \mid \texttt{fn}\ x{:}T.t \mid \texttt{fun}\ f{:}T.x{:}T.t
\end{array}
$$

Sequential composition $e_1; e_2$ is represented by `let`-construct

$$\texttt{let}\ x{:}T = e_1\ \texttt{in}\ e_2\ , \qquad x \notin fv(e_2)$$

For the operational semantics:

$$P \quad ::= \quad \emptyset \mid p\langle t \rangle \mid P \parallel P$$

$$\sigma \vdash P \longrightarrow \sigma' \vdash P' \text{ with } \quad \sigma : L \mapsto \{\text{free}, p(n)\}$$

Syntax

$$
\begin{array}{lcl}
t & ::= & \texttt{stop} \mid v \mid \texttt{let}\ x{:}T = e\ \texttt{in}\ t \\
e & ::= & t \mid e\ e \mid \texttt{if}\ e\ \texttt{then}\ e\ \texttt{else}\ e \mid \texttt{spawn}\ e \mid \texttt{new}\ L \mid v.\,\texttt{lock} \\
 & \mid & v.\,\texttt{unlock} \\
v & ::= & x \mid l \mid \texttt{fn}\ x{:}T.t \mid \texttt{fun}\ f{:}T.x{:}T.t
\end{array}
$$

Sequential composition $e_1; e_2$ is represented by `let`-construct

$$\texttt{let}\ x{:}T = e_1\ \texttt{in}\ e_2\ , \qquad x \notin fv(e_2)$$

For the operational semantics:

$$P \quad ::= \quad \emptyset \mid p\langle t \rangle \mid P \parallel P$$

$$\sigma \vdash P \longrightarrow \sigma' \vdash P' \text{ with } \quad \sigma : L \mapsto \{\text{free}, p(n)\}$$

An example run:

$$\emptyset \vdash p_0\langle P \rangle \longrightarrow \ldots \longrightarrow [l_1 \mapsto p_1(1), l_2 \mapsto p_0(1)] \vdash p_1\langle l_2.\,\texttt{lock} \rangle \parallel p_0\langle l_1.\,\texttt{lock} \rangle$$

Circular Wait

Lemma

A deadlocked configuation $\sigma \vdash P$ is of the form:
$P = P' \parallel p_0\langle t_0 \rangle \parallel \ldots \parallel p_k\langle t_k \rangle$, where $k \geq 2$ and where the threads t_i are of the form $t_i = \texttt{let}\ x_i{:}T_i = l_{i+1}.\ \texttt{lock}\ \texttt{in}\ t_i'$ for $0 \leq i < k$, and $t_k = \texttt{let}\ x_k{:}T_k = l_0.\ \texttt{lock}\ \texttt{in}\ t_k'$.

Type and Effect System

The judgment of our type and effect system is given by:

$$\Gamma \vdash e : T :: \varphi$$

Types and effects are described by:

$$\begin{array}{rcl} T & ::= & \texttt{Bool} \mid \texttt{Int} \mid T \rightarrow^{\varphi} T \mid \texttt{L}^{r} \mid \texttt{Thread} \\ \varphi & ::= & \epsilon \mid x \mid \varphi;\varphi \mid \varphi + \varphi \mid \mu x.\varphi \mid \texttt{spawn}\,\varphi \mid \nu \texttt{L}^{r} \\ & \mid & \texttt{L}^{r}\,.\,\texttt{lock} \mid \texttt{L}^{r}\,.\,\texttt{unlock} \\ r & ::= & \{\pi\} \mid r \cup r \mid \emptyset \end{array}$$

Type and Effect System with Inference Algorithm

The judgment of the type and effect system with the inference algorithm is given by:

$$\Gamma \vdash e : T :: \varphi$$

Types and effects are modified as:

$$\begin{array}{rcl} T & ::= & \alpha \mid \texttt{Bool} \mid \texttt{Int} \mid T \rightarrow^{\varphi} T \mid \texttt{L}^{r} \mid \texttt{Thread} \\ \varphi & ::= & \beta \mid \epsilon \mid x \mid \varphi ; \varphi \mid \varphi + \varphi \mid \mu x.\varphi \mid \texttt{spawn}\, \varphi \mid \nu \texttt{L}^{r} \\ & \mid & \texttt{L}^{r}\,.\,\texttt{lock} \mid \texttt{L}^{r}\,.\,\texttt{unlock} \\ r & ::= & \rho \mid \{\pi\} \mid r \cup r \mid \emptyset \end{array}$$

Type and Effect System with Inference Algorithm

The judgment of the type and effect system with the inference algorithm is given by:

$$\Gamma \vdash e : T :: \varphi, \theta$$

where θ is a substitution which maps:

- $\alpha \mapsto T$, α is a type variable
- $\beta \mapsto \varphi$, β is an effect variable
- $\rho \mapsto r$, ρ is an annotation variable

Types and effects are modified as:

$$\begin{array}{rcl} T & ::= & \alpha \mid \texttt{Bool} \mid \texttt{Int} \mid T \rightarrow^{\varphi} T \mid \texttt{L}^{r} \mid \texttt{Thread} \\ \varphi & ::= & \beta \mid \epsilon \mid x \mid \varphi;\varphi \mid \varphi + \varphi \mid \mu x.\varphi \mid \texttt{spawn}\,\varphi \mid \nu \texttt{L}^{r} \\ & \mid & \texttt{L}^{r}\,.\,\texttt{lock} \mid \texttt{L}^{r}\,.\,\texttt{unlock} \\ r & ::= & \rho \mid \{\pi\} \mid r \cup r \mid \emptyset \end{array}$$

Type and Effect System

For our example:

```
let t = fn (z1,z2). z1.lock;z2.lock;stop
in
    let x1 = new_π1 L in
    let x2 = new_π2 L in
    spawn (t (x1,x2)); t (x2,x1); stop
```

Type and Effect System

For our example:

```
let t = fn (z1,z2). z1.lock;z2.lock;stop
in
    let x1 = new_π1 L in
    let x2 = new_π2 L in
    spawn (t (x1,x2)); t (x2,x1); stop
```

Effect:

$\varphi =$

Type and Effect System

For our example:

```
let t = fn (z1,z2). z1.lock;z2.lock;stop
in
    let x1 = new_π1 L in
    let x2 = new_π2 L in
    spawn (t (x1,x2)); t (x2,x1); stop
```

Effect:

$$\varphi = \nu \mathrm{L}^{\pi_1}; \nu \mathrm{L}^{\pi_2};$$

$$\Gamma \vdash \mathtt{new}_{\pi}\ L : \mathrm{L}^{\pi} :: \nu \mathrm{L}^{\pi} \qquad \text{TE-NewL}$$

Type and Effect System

For our example:

```
let t = fn (z1,z2). z1.lock;z2.lock;stop
in
    let x1 = new_π1 L in
    let x2 = new_π2 L in
    spawn (t (x1,x2)); t (x2,x1); stop
```

Effect:

$$\varphi = \nu \mathrm{L}^{\pi_1}; \nu \mathrm{L}^{\pi_2};\ \mathtt{spawn}\ (\qquad\qquad\qquad\qquad\qquad\qquad);$$

$$\frac{\Gamma \vdash e : T :: \varphi}{\Gamma \vdash \mathtt{spawn}\ e : \mathtt{Thread::spawn}\ \varphi}\ \text{TE-SPAWN}$$

Type and Effect System

For our example:

```
let t = fn (z1,z2). z1.lock;z2.lock;stop
in
    let x1 = new_π1 L in
    let x2 = new_π2 L in
    spawn (t (x1,x2)); t (x2,x1); stop
```

Effect:

$$\varphi = \nu \mathrm{L}^{\pi_1}; \nu \mathrm{L}^{\pi_2};\ \mathtt{spawn}\,(\,\mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}\,);$$

$$\frac{\Gamma \vdash v : \mathrm{L}^r :: \varphi}{\Gamma \vdash v.\,\mathtt{lock}: \mathrm{L}^r :: \varphi; \mathrm{L}^r\,.\,\mathtt{lock}}\ \text{TE-Lock}$$

$$\frac{\Gamma \vdash e_1 : T_2 \to^{\varphi} T_1 :: \varphi_1 \qquad \Gamma \vdash e_2 : T_2 :: \varphi_2}{\Gamma \vdash e_1\ e_2 : T_1 :: \varphi_1; \varphi_2; \varphi}\ \text{TE-App}$$

Type and Effect System

For our example:

```
let t = fn (z1,z2). z1.lock;z2.lock;stop
in
    let x1 = new_π1 L in
    let x2 = new_π2 L in
    spawn (t (x1,x2)); t (x2,x1); stop
```

Effect:

$$\varphi = \nu \mathrm{L}^{\pi_1}; \nu \mathrm{L}^{\pi_2};\ \mathtt{spawn}\,(\,\mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}\,);\ \mathrm{L}^{\pi_2}.\mathtt{lock}; \mathrm{L}^{\pi_1}.\mathtt{lock}$$

$$\frac{\Gamma \vdash v : \mathrm{L}^r :: \varphi}{\Gamma \vdash v.\,\mathtt{lock} : \mathrm{L}^r :: \varphi; \mathrm{L}^r\,.\,\mathtt{lock}}\ \text{TE-Lock}$$

$$\frac{\Gamma \vdash e_1 : T_2 \to^{\varphi} T_1 :: \varphi_1 \qquad \Gamma \vdash e_2 : T_2 :: \varphi_2}{\Gamma \vdash e_1\ e_2 : T_1 :: \varphi_1; \varphi_2; \varphi}\ \text{TE-App}$$

Type and Effect System

$$\frac{\Gamma, x : T_1 \vdash e : T_2 :: \varphi}{\Gamma \vdash \texttt{fn}\ x : T_1 . e : T_1 \rightarrow^{\varphi} T_2 :: \epsilon}\ \text{TE-Abs}$$

$$\frac{\Gamma \vdash e_1 : T_2 \rightarrow^{\varphi} T_1 :: \varphi_1 \qquad \Gamma \vdash e_2 : T_2 :: \varphi_2}{\Gamma \vdash e_1\ e_2 : T_1 :: \varphi_1; \varphi_2; \varphi}\ \text{TE-App}$$

$$\frac{\Gamma \vdash e_1 : T_1 :: \varphi_1 \qquad \Gamma, x{:}T_1 \vdash e_2 : T_2 :: \varphi_2}{\Gamma \vdash \texttt{let}\ x : T_1 = e_1\ \texttt{in}\ e_2 : T_2 :: \varphi_1; \varphi_2}\ \text{TE-Let}$$

$$\frac{\Gamma \vdash e : T :: \varphi}{\Gamma \vdash \texttt{spawn}\ e : \texttt{Thread::spawn}\ \varphi}\ \text{TE-Spawn}$$

$$\Gamma \vdash \mathsf{new}_{\pi}\ L : \mathrm{L}^{\pi} :: \nu \mathrm{L}^{\pi}\ \ \text{TE-NewL}$$

$$\frac{\Gamma \vdash v : \mathrm{L}^{r} :: \varphi}{\Gamma \vdash v.\ \texttt{lock}: \mathrm{L}^{r} :: \varphi; \mathrm{L}^{r}\ .\ \texttt{lock}}\ \text{TE-Lock}$$

Deadlock Checking

To detect a deadlock in a program, we execute the abstract behaviour of the program.

$$E : p\langle\varphi\rangle \mid p\langle\varphi\rangle \parallel E$$

$$\sigma : L^{\pi} \mapsto \{\mathrm{free}, p(n)\}$$

Deadlock Checking

In our example:

```
let t = fn (z1,z2). z1.lock;z2.lock;stop
in
    let x1 = new_π1 L in
    let x2 = new_π2 L in
    spawn (t (x1,x2)); t (x2,x1); stop
```

We have the effect:

$$\varphi = \nu \mathrm{L}^{\pi_1}; \nu \mathrm{L}^{\pi_2}; \mathtt{spawn}\,(\mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}); \mathrm{L}^{\pi_2}.\mathtt{lock}; \mathrm{L}^{\pi_1}.\mathtt{lock}$$

Deadlock Checking

$$\varphi = \nu \mathrm{L}^{\pi_1}; \nu \mathrm{L}^{\pi_2}; \mathtt{spawn}\ (\mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}); \mathrm{L}^{\pi_2}.\mathtt{lock}; \mathrm{L}^{\pi_1}.\mathtt{lock}$$

$$\sigma \vdash t_0\langle \nu \mathrm{L}^{\pi_1}; \nu \mathrm{L}^{\pi_2}; \mathtt{spawn}\ (\mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}); \mathrm{L}^{\pi_2}.\mathtt{lock}; \mathrm{L}^{\pi_1}.\mathtt{lock}\rangle$$

Deadlock Checking

$$\varphi = \nu \mathrm{L}^{\pi_1}; \nu \mathrm{L}^{\pi_2}; \mathtt{spawn}\ (\mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}); \mathrm{L}^{\pi_2}.\mathtt{lock}; \mathrm{L}^{\pi_1}.\mathtt{lock}$$

$$\sigma \vdash t_0\langle \nu \mathrm{L}^{\pi_1}; \nu \mathrm{L}^{\pi_2}; \mathtt{spawn}\ (\mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}); \mathrm{L}^{\pi_2}.\mathtt{lock}; \mathrm{L}^{\pi_1}.\mathtt{lock}\rangle$$

$$\frac{\sigma(\mathrm{L}^{\pi}) = \mathsf{undef} \qquad \sigma' = \sigma[\mathrm{L}^{\pi} \mapsto \mathit{free}]}{\sigma \vdash p\langle \nu \mathrm{L}^{r}\rangle \rightarrow \sigma' \vdash p\langle \epsilon \rangle}\ \text{RE-NewL}$$

Deadlock Checking

$$\varphi = \nu \mathrm{L}^{\pi_1}; \nu \mathrm{L}^{\pi_2}; \mathtt{spawn}\ (\mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}); \mathrm{L}^{\pi_2}.\mathtt{lock}; \mathrm{L}^{\pi_1}.\mathtt{lock}$$

$$\begin{array}{ll} & \sigma \vdash t_0\langle \nu \mathrm{L}^{\pi_1}; \nu \mathrm{L}^{\pi_2}; \mathtt{spawn}\ (\mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}); \mathrm{L}^{\pi_2}.\mathtt{lock}; \mathrm{L}^{\pi_1}.\mathtt{lock}\rangle \\ \rightarrow & \sigma \vdash t_0\langle \epsilon; \nu \mathrm{L}^{\pi_2}; \mathtt{spawn}\ (\mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}); \mathrm{L}^{\pi_2}.\mathtt{lock}; \mathrm{L}^{\pi_1}.\mathtt{lock}\rangle \end{array}$$

$$\frac{\sigma(\mathrm{L}^{\pi}) = \mathsf{undef} \qquad \sigma' = \sigma[\mathrm{L}^{\pi} \mapsto \mathit{free}]}{\sigma \vdash p\langle \nu \mathrm{L}^{r} \rangle \rightarrow \sigma' \vdash p\langle \epsilon \rangle}\ \text{RE-NewL}$$

Deadlock Checking

$$\varphi = \nu \mathrm{L}^{\pi_1}; \nu \mathrm{L}^{\pi_2}; \mathtt{spawn}\ (\mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}); \mathrm{L}^{\pi_2}.\mathtt{lock}; \mathrm{L}^{\pi_1}.\mathtt{lock}$$

$$\begin{array}{ll} & \sigma \vdash t_0\langle \nu \mathrm{L}^{\pi_1}; \nu \mathrm{L}^{\pi_2}; \mathtt{spawn}\ (\mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}); \mathrm{L}^{\pi_2}.\mathtt{lock}; \mathrm{L}^{\pi_1}.\mathtt{lock}\rangle \\ \rightarrow & \sigma \vdash t_0\langle \epsilon; \nu \mathrm{L}^{\pi_2}; \mathtt{spawn}\ (\mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}); \mathrm{L}^{\pi_2}.\mathtt{lock}; \mathrm{L}^{\pi_1}.\mathtt{lock}\rangle \\ \rightarrow & \sigma \vdash t_0\langle \epsilon;\ \mathtt{spawn}\ (\mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}); \mathrm{L}^{\pi_2}.\mathtt{lock}; \mathrm{L}^{\pi_1}.\mathtt{lock}\rangle \end{array}$$

Deadlock Checking

$$\varphi = \nu \mathrm{L}^{\pi_1}; \nu \mathrm{L}^{\pi_2}; \mathtt{spawn}\ (\mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}); \mathrm{L}^{\pi_2}.\mathtt{lock}; \mathrm{L}^{\pi_1}.\mathtt{lock}$$

$$\begin{array}{ll} & \sigma \vdash t_0\langle \nu \mathrm{L}^{\pi_1}; \nu \mathrm{L}^{\pi_2}; \mathtt{spawn}\ (\mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}); \mathrm{L}^{\pi_2}.\mathtt{lock}; \mathrm{L}^{\pi_1}.\mathtt{lock}\rangle \\ \rightarrow & \sigma \vdash t_0\langle \epsilon; \nu \mathrm{L}^{\pi_2}; \mathtt{spawn}\ (\mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}); \mathrm{L}^{\pi_2}.\mathtt{lock}; \mathrm{L}^{\pi_1}.\mathtt{lock}\rangle \\ \rightarrow & \sigma \vdash t_0\langle \epsilon;\ \mathtt{spawn}\ (\mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}); \mathrm{L}^{\pi_2}.\mathtt{lock}; \mathrm{L}^{\pi_1}.\mathtt{lock}\rangle \end{array}$$

$$\sigma \vdash p_1\langle (\mathtt{spawn}\ \varphi); \varphi' \rangle \rightarrow \sigma \vdash p_1\langle \varphi' \rangle \parallel p_2\langle \varphi \rangle \qquad \text{RE-SPAWN}$$

Deadlock Checking

$$\varphi = \nu \mathrm{L}^{\pi_1}; \nu \mathrm{L}^{\pi_2}; \mathtt{spawn}\ (\mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}); \mathrm{L}^{\pi_2}.\mathtt{lock}; \mathrm{L}^{\pi_1}.\mathtt{lock}$$

$$\begin{array}{ll} & \sigma \vdash t_0\langle \nu \mathrm{L}^{\pi_1}; \nu \mathrm{L}^{\pi_2}; \mathtt{spawn}\ (\mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}); \mathrm{L}^{\pi_2}.\mathtt{lock}; \mathrm{L}^{\pi_1}.\mathtt{lock}\rangle \\ \rightarrow & \sigma \vdash t_0\langle \epsilon; \nu \mathrm{L}^{\pi_2}; \mathtt{spawn}\ (\mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}); \mathrm{L}^{\pi_2}.\mathtt{lock}; \mathrm{L}^{\pi_1}.\mathtt{lock}\rangle \\ \rightarrow & \sigma \vdash t_0\langle \epsilon;\ \mathtt{spawn}\ (\mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}); \mathrm{L}^{\pi_2}.\mathtt{lock}; \mathrm{L}^{\pi_1}.\mathtt{lock}\rangle \\ \rightarrow & \sigma \vdash t_0\langle\ \mathrm{L}^{\pi_2}.\mathtt{lock}; \mathrm{L}^{\pi_1}.\mathtt{lock}\rangle \parallel t\langle \mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}\rangle \end{array}$$

Deadlock Checking

$$\varphi = \nu \mathtt{L}^{\pi_1}; \nu \mathtt{L}^{\pi_2}; \mathtt{spawn}\ (\mathtt{L}^{\pi_1}.\mathtt{lock}; \mathtt{L}^{\pi_2}.\mathtt{lock}); \mathtt{L}^{\pi_2}.\mathtt{lock}; \mathtt{L}^{\pi_1}.\mathtt{lock}$$

$$\begin{array}{ll} & \sigma \vdash t_0\langle \nu \mathtt{L}^{\pi_1}; \nu \mathtt{L}^{\pi_2}; \mathtt{spawn}\ (\mathtt{L}^{\pi_1}.\mathtt{lock}; \mathtt{L}^{\pi_2}.\mathtt{lock}); \mathtt{L}^{\pi_2}.\mathtt{lock}; \mathtt{L}^{\pi_1}.\mathtt{lock}\rangle \\ \longrightarrow & \sigma \vdash t_0\langle \epsilon; \nu \mathtt{L}^{\pi_2}; \mathtt{spawn}\ (\mathtt{L}^{\pi_1}.\mathtt{lock}; \mathtt{L}^{\pi_2}.\mathtt{lock}); \mathtt{L}^{\pi_2}.\mathtt{lock}; \mathtt{L}^{\pi_1}.\mathtt{lock}\rangle \\ \longrightarrow & \sigma \vdash t_0\langle \epsilon;\ \mathtt{spawn}\ (\mathtt{L}^{\pi_1}.\mathtt{lock}; \mathtt{L}^{\pi_2}.\mathtt{lock}); \mathtt{L}^{\pi_2}.\mathtt{lock}; \mathtt{L}^{\pi_1}.\mathtt{lock}\rangle \\ \longrightarrow & \sigma \vdash t_0\langle\ \mathtt{L}^{\pi_2}.\mathtt{lock}; \mathtt{L}^{\pi_1}.\mathtt{lock}\rangle \parallel t\langle \mathtt{L}^{\pi_1}.\mathtt{lock}; \mathtt{L}^{\pi_2}.\mathtt{lock}\rangle \end{array}$$

$$\frac{\pi \in r \quad \sigma(\mathtt{L}^{\pi}) = \mathit{free} \vee \sigma(\mathtt{L}^{\pi}) = p(n) \qquad \sigma' = \sigma[\mathtt{L}^{\pi} \mapsto \sigma(\mathtt{L}^{\pi}) + 1]}{\sigma \vdash p\langle \mathtt{L}^{r}.\,\mathtt{lock}\rangle \longrightarrow \sigma' \vdash p\langle \epsilon \rangle}\ \text{RE-Lock}$$

Deadlock Checking

$$\varphi = \nu \mathrm{L}^{\pi_1}; \nu \mathrm{L}^{\pi_2}; \mathtt{spawn}\ (\mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}); \mathrm{L}^{\pi_2}.\mathtt{lock}; \mathrm{L}^{\pi_1}.\mathtt{lock}$$

$$\begin{array}{ll} & \sigma \vdash t_0\langle \nu \mathrm{L}^{\pi_1}; \nu \mathrm{L}^{\pi_2}; \mathtt{spawn}\ (\mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}); \mathrm{L}^{\pi_2}.\mathtt{lock}; \mathrm{L}^{\pi_1}.\mathtt{lock}\rangle \\ \longrightarrow & \sigma \vdash t_0\langle \epsilon; \nu \mathrm{L}^{\pi_2}; \mathtt{spawn}\ (\mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}); \mathrm{L}^{\pi_2}.\mathtt{lock}; \mathrm{L}^{\pi_1}.\mathtt{lock}\rangle \\ \longrightarrow & \sigma \vdash t_0\langle \epsilon;\ \mathtt{spawn}\ (\mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}); \mathrm{L}^{\pi_2}.\mathtt{lock}; \mathrm{L}^{\pi_1}.\mathtt{lock}\rangle \\ \longrightarrow & \sigma \vdash t_0\langle\ \mathrm{L}^{\pi_2}.\mathtt{lock}; \mathrm{L}^{\pi_1}.\mathtt{lock}\rangle \parallel t\langle \mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}\rangle \\ \longrightarrow & \sigma[\mathrm{L}^{\pi_2} \mapsto t_0(1)] \vdash t_0\langle \epsilon; \mathrm{L}^{\pi_1}.\mathtt{lock}\rangle \parallel t\langle\ \mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}\rangle \end{array}$$

$$\frac{\pi \in r \quad \sigma(\mathrm{L}^{\pi}) = \mathit{free} \vee \sigma(\mathrm{L}^{\pi}) = p(n) \qquad \sigma' = \sigma[\mathrm{L}^{\pi} \mapsto \sigma(\mathrm{L}^{\pi}) + 1]}{\sigma \vdash p\langle \mathrm{L}^{r}\,.\,\mathtt{lock}\rangle \longrightarrow \sigma' \vdash p\langle \epsilon\rangle}\ \text{RE-Lock}$$

Deadlock Checking

$$\varphi = \nu \mathtt{L}^{\pi_1}; \nu \mathtt{L}^{\pi_2}; \mathtt{spawn}\ (\mathtt{L}^{\pi_1}.\mathtt{lock}; \mathtt{L}^{\pi_2}.\mathtt{lock}); \mathtt{L}^{\pi_2}.\mathtt{lock}; \mathtt{L}^{\pi_1}.\mathtt{lock}$$

$$\begin{aligned}
& \sigma \vdash t_0\langle \nu \mathtt{L}^{\pi_1}; \nu \mathtt{L}^{\pi_2}; \mathtt{spawn}\ (\mathtt{L}^{\pi_1}.\mathtt{lock}; \mathtt{L}^{\pi_2}.\mathtt{lock}); \mathtt{L}^{\pi_2}.\mathtt{lock}; \mathtt{L}^{\pi_1}.\mathtt{lock}\rangle \\
\longrightarrow\ & \sigma \vdash t_0\langle \epsilon; \nu \mathtt{L}^{\pi_2}; \mathtt{spawn}\ (\mathtt{L}^{\pi_1}.\mathtt{lock}; \mathtt{L}^{\pi_2}.\mathtt{lock}); \mathtt{L}^{\pi_2}.\mathtt{lock}; \mathtt{L}^{\pi_1}.\mathtt{lock}\rangle \\
\longrightarrow\ & \sigma \vdash t_0\langle \epsilon;\ \mathtt{spawn}\ (\mathtt{L}^{\pi_1}.\mathtt{lock}; \mathtt{L}^{\pi_2}.\mathtt{lock}); \mathtt{L}^{\pi_2}.\mathtt{lock}; \mathtt{L}^{\pi_1}.\mathtt{lock}\rangle \\
\longrightarrow\ & \sigma \vdash t_0\langle\ \mathtt{L}^{\pi_2}.\mathtt{lock}; \mathtt{L}^{\pi_1}.\mathtt{lock}\rangle \parallel t\langle \mathtt{L}^{\pi_1}.\mathtt{lock}; \mathtt{L}^{\pi_2}.\mathtt{lock}\rangle \\
\longrightarrow\ & \sigma[\mathtt{L}^{\pi_2} \mapsto t_0(1)] \vdash t_0\langle \epsilon; \mathtt{L}^{\pi_1}.\mathtt{lock}\rangle \parallel t\langle\ \mathtt{L}^{\pi_1}.\mathtt{lock}; \mathtt{L}^{\pi_2}.\mathtt{lock}\rangle
\end{aligned}$$

$$\frac{\pi \in r \quad \sigma(\mathtt{L}^{\pi}) = \mathit{free} \vee \sigma(\mathtt{L}^{\pi}) = p(n) \qquad \sigma' = \sigma[\mathtt{L}^{\pi} \mapsto \sigma(\mathtt{L}^{\pi}) + 1]}{\sigma \vdash p\langle \mathtt{L}^{r}.\,\mathtt{lock}\rangle \longrightarrow \sigma' \vdash p\langle \epsilon\rangle}\ \text{RE-Lock}$$

Deadlock Checking

$$\varphi = \nu \mathrm{L}^{\pi_1}; \nu \mathrm{L}^{\pi_2}; \mathtt{spawn}\ (\mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}); \mathrm{L}^{\pi_2}.\mathtt{lock}; \mathrm{L}^{\pi_1}.\mathtt{lock}$$

$$\begin{array}{ll}
 & \sigma \vdash t_0\langle \nu \mathrm{L}^{\pi_1}; \nu \mathrm{L}^{\pi_2}; \mathtt{spawn}\ (\mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}); \mathrm{L}^{\pi_2}.\mathtt{lock}; \mathrm{L}^{\pi_1}.\mathtt{lock}\rangle \\
\longrightarrow & \sigma \vdash t_0\langle \epsilon; \nu \mathrm{L}^{\pi_2}; \mathtt{spawn}\ (\mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}); \mathrm{L}^{\pi_2}.\mathtt{lock}; \mathrm{L}^{\pi_1}.\mathtt{lock}\rangle \\
\longrightarrow & \sigma \vdash t_0\langle \epsilon;\ \mathtt{spawn}\ (\mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}); \mathrm{L}^{\pi_2}.\mathtt{lock}; \mathrm{L}^{\pi_1}.\mathtt{lock}\rangle \\
\longrightarrow & \sigma \vdash t_0\langle\ \mathrm{L}^{\pi_2}.\mathtt{lock}; \mathrm{L}^{\pi_1}.\mathtt{lock}\rangle \parallel t\langle \mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}\rangle \\
\longrightarrow & \sigma[\mathrm{L}^{\pi_2} \mapsto t_0(1)] \vdash t_0\langle \epsilon; \mathrm{L}^{\pi_1}.\mathtt{lock}\rangle \parallel t\langle\ \mathrm{L}^{\pi_1}.\mathtt{lock}; \mathrm{L}^{\pi_2}.\mathtt{lock}\rangle \\
\longrightarrow & \sigma[\mathrm{L}^{\pi_2} \mapsto t_0(1), \mathrm{L}^{\pi_1} \mapsto t(1)] \vdash t_0\langle \mathrm{L}^{\pi_1}.\mathtt{lock}\rangle \parallel t\langle \mathrm{L}^{\pi_2}.\mathtt{lock}\rangle
\end{array}$$

Deadlock checking

$$\sigma \vdash t_0\langle \mathrm{L}^{\pi_2} \,.\, \texttt{lock}; \mathrm{L}^{\pi_1} \,.\, \texttt{lock}\rangle \parallel t\langle \mathrm{L}^{\pi_1} \,.\, \texttt{lock}; \mathrm{L}^{\pi_2} \,.\, \texttt{lock}\rangle$$

$$\rightarrow \quad \sigma[\mathrm{L}^{\pi_2} \mapsto t_0(1), \mathrm{L}^{\pi_1} \mapsto t(1)] \vdash t_0\langle \mathrm{L}^{\pi_1} \,.\, \texttt{lock}\rangle \parallel t\langle \epsilon; \mathrm{L}^{\pi_2} \,.\, \texttt{lock}\rangle$$

OR

$$\rightarrow \quad \sigma[\mathrm{L}^{\pi_1} \mapsto t(1), \mathrm{L}^{\pi_2} \mapsto t_0(1)] \vdash t_0\langle \epsilon; \mathrm{L}^{\pi_1} \,.\, \texttt{lock}\rangle \parallel t\langle \mathrm{L}^{\pi_2} \,.\, \texttt{lock}\rangle$$

$\ldots$

Operational Semantics for Abstract Behaviour

$$\frac{\sigma(\mathtt{L}^{\pi}) = \mathsf{undef} \qquad \sigma' = \sigma[\mathtt{L}^{\pi} \mapsto \mathit{free}]}{\sigma \vdash p\langle \nu \mathtt{L}^{r} \rangle \rightarrow \sigma' \vdash p\langle \epsilon \rangle} \text{ RE-NewL}$$

$$\frac{\pi \in r \quad \sigma(\mathtt{L}^{\pi}) = \mathit{free} \vee \sigma(\mathtt{L}^{\pi}) = p(n) \qquad \sigma' = \sigma[\mathtt{L}^{\pi} \mapsto \sigma(\mathtt{L}^{\pi}) + 1]}{\sigma \vdash p\langle \mathtt{L}^{r}\,.\,\mathtt{lock} \rangle \rightarrow \sigma' \vdash p\langle \epsilon \rangle} \text{ RE-Lock}_1$$

$$\sigma \vdash p_1\langle (\mathtt{spawn}\ \varphi); \varphi' \rangle \rightarrow \sigma \vdash p_1\langle \varphi' \rangle \parallel p_2\langle \varphi \rangle \qquad \text{RE-Spawn}$$

Summary

- Conclusion:
  - Type and effect system for locks
  - Algorithm for inferring behaviour of locks
  - Program points abstract concrete locks
  - Effect analysis for potential deadlock
- Future Work:
  - Applying to communication analysis of asynchronous systems
  - Relaxing the condition (e.g. lock creation in loop)
  - Abstracting processes

References I

[Agarwal et al., 2006] Agarwal, R., Wang, L., and Stoller, S. D. (2006).
Detecting potential deadlocks with state analysis and run-time monitoring.
In Ur, S., Bin, E., and Wolfsthal, Y., editors, *Proceedings of the Haifa Verification Conference 2005*, volume 3875 of *Lecture Notes in Computer Science*, pages 191–207. Springer-Verlag.

[Boyapati et al., 2002] Boyapati, C., Lee, R., and Rinard, M. (2002).
Ownership types for safe programming: Preventing data races and deadlocks.
In *Object Oriented Programming: Systems, Languages, and Applications (OOPSLA) '02 (Seattle, USA)*. ACM.
In *SIGPLAN Notices*.

[Coffman Jr. et al., 1971] Coffman Jr., E. G., Elphick, M., and Shoshani, A. (1971).
System deadlocks.
Computing Surveys, 3(2):67–78.

[de Boer and Grabe, 2007] de Boer, F. S. and Grabe, I. (2007).
Finite-state call-chain abstractions for deadlock detection in multithreaded object-oriented languages (extended abstract).
In Johnsen, E. B., Owe, O., and Schneider, G., editors, *Proceedings of the 19th Nordic Workshop on Programming Theory (NWPT'07). Extended Abstracts. University of Oslo, Dept. of Computer Science, Technical Report 366*.

[Engler and Ashcraft, 2003] Engler, D. R. and Ashcraft, K. (2003).
Effective, static detection of race conditions and deadlocks: RacerX.
In *Proceedings of the 24th ACM Symposium on Operating Systems Principles*, pages 237–252.

[Williams et al., 2005] Williams, A., Thies, W., and Ernst, M. D. (2005).
Static deadlock detection for Java libraries.
In *Proceedings of the Conference on Object-Oriented Programming Systems, Languages (ECOOP 2005)*.