Formalisation d'analyses statiques pour Java multithreadé

LANDE, (ParSec, Rennes, 09/06/2008)

[1]INRIA, LANDE

Outline

Contexte

Analyse statique de bytecode Java séquentiel

- e.g : non-null pointer, contrôle de ressources,...
- interprétation abstraite
- certification Coq

Objectif

Etendre ces techniques au multithreading

Deux activités parallèles

Prouver la bonne synchronisation

Certification[a] d'une analyse statique (Naik & Aiken, POPL'07).

[a] (définition ?)

Analyser des programmes bien synchronisés

Définition et **certification** d'analyses statiques reposant sur l'**hypothèse de bonne synchronisation**.

Outline

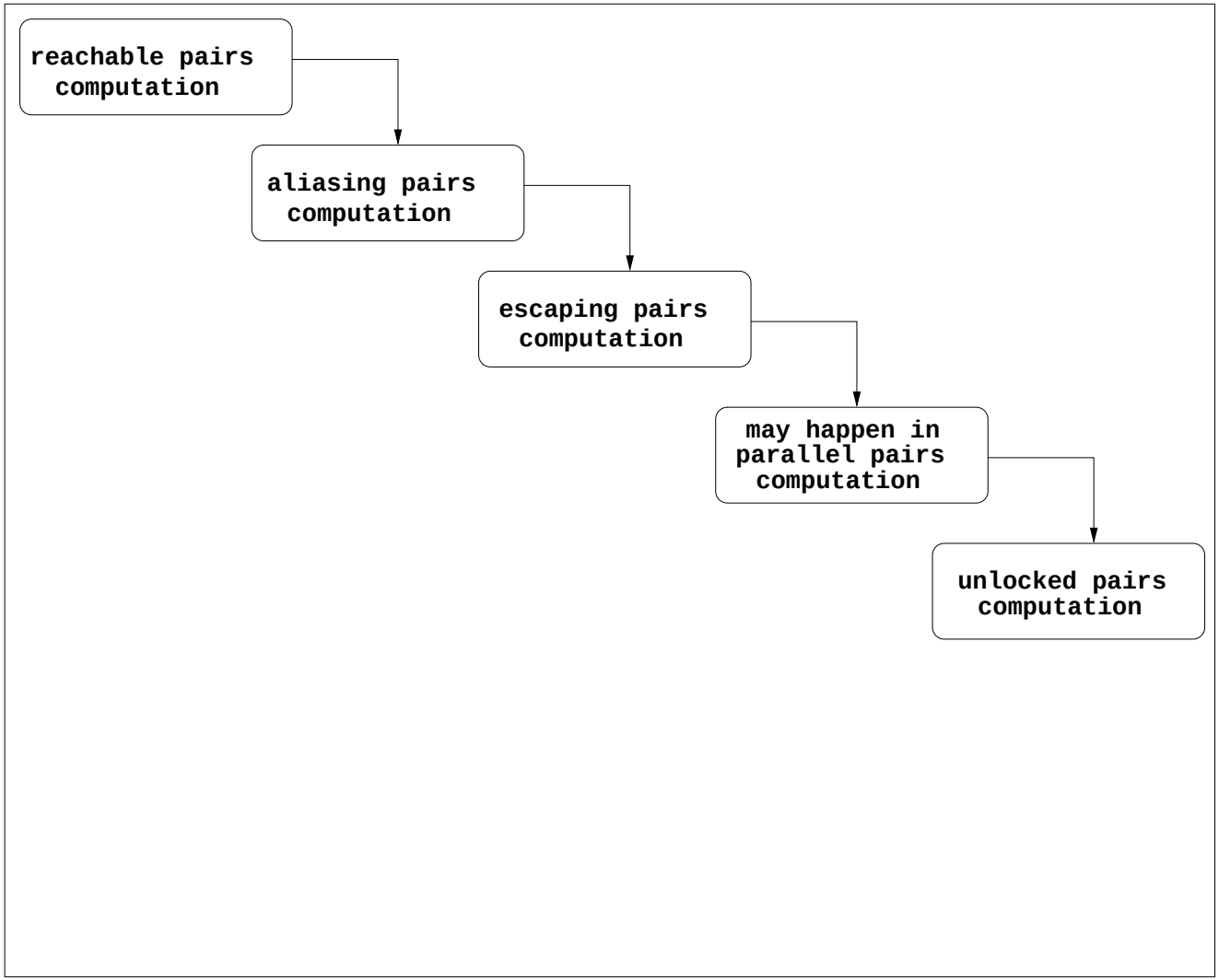
Naik & Aiken, POPL'07

Naik & Aiken, POPL'07

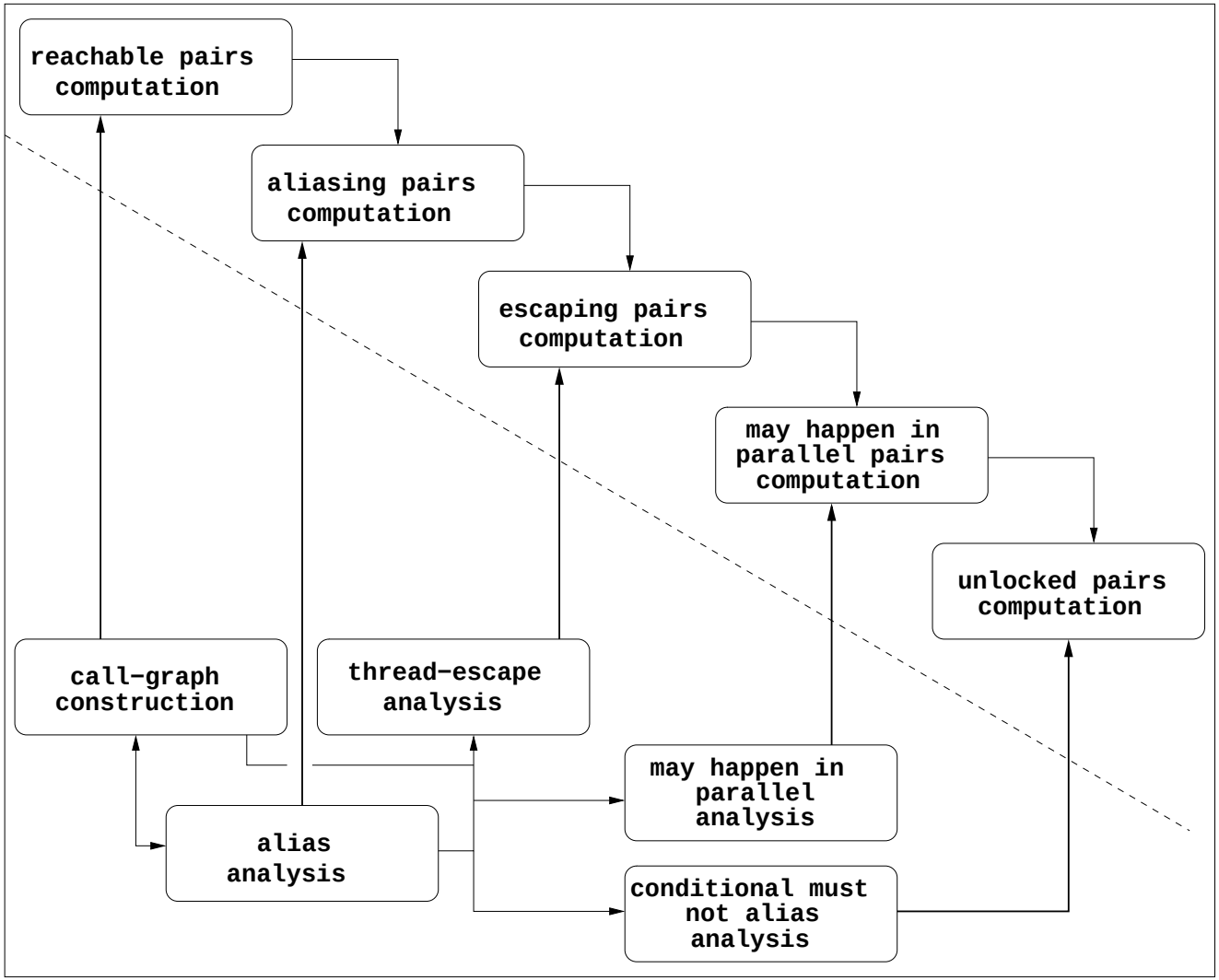
- calcul des **paires d'accès**[a] du programme (au moins une écriture)
- Elimination, par étapes successives[b], **de** paires **non impliquées dans une datarace**
- But du jeu : terminer avec un ensemble de paires vide

[a] points de programme
[b] analyse statique

Etapes


reachable pairs computation
aliasing pairs computation
escaping pairs computation
may happen in parallel pairs computation
unlocked pairs computation

Etapes


reachable pairs computation
aliasing pairs computation
escaping pairs computation
may happen in parallel pairs computation
unlocked pairs computation
call-graph construction
thread-escape analysis
may happen in parallel analysis
alias analysis
conditional must not alias analysis

Unlocked pairs

Exemple

thread t	thread t'
$sync(A)\{$ $\ldots$ $x.f.g = \_;\}$	$sync(A)\{$ $\ldots$ $x.f.g = \_;\}$

`case A of`

Unlocked pairs

Exemple

thread t	thread t'
$sync(A)\{$ $\ldots$ $x.f.g = \_;\}$	$sync(A)\{$ $\ldots$ $x.f.g = \_;\}$

`case A of`

- *ClassId*.*f*, ok(ME)

Unlocked pairs

Exemple

thread t	thread t'
$sync(A)\{$ $\ldots$ $x.f.g = \_;\}$	$sync(A)\{$ $\ldots$ $x.f.g = \_;\}$

`case A of`

- *ClassId*.f, ok(ME)
- $x.f$
  - $[x.f]_t = [x.f]_{t'}$, ok (ME)
  - $[x.f]_t \neq [x.f]_{t'}$, ok

Unlocked pairs

Exemple

thread t	thread t'
$sync(A)\{$ $\ldots$ $x.f.g = \_;\}$	$sync(A)\{$ $\ldots$ $x.f.g = \_;\}$

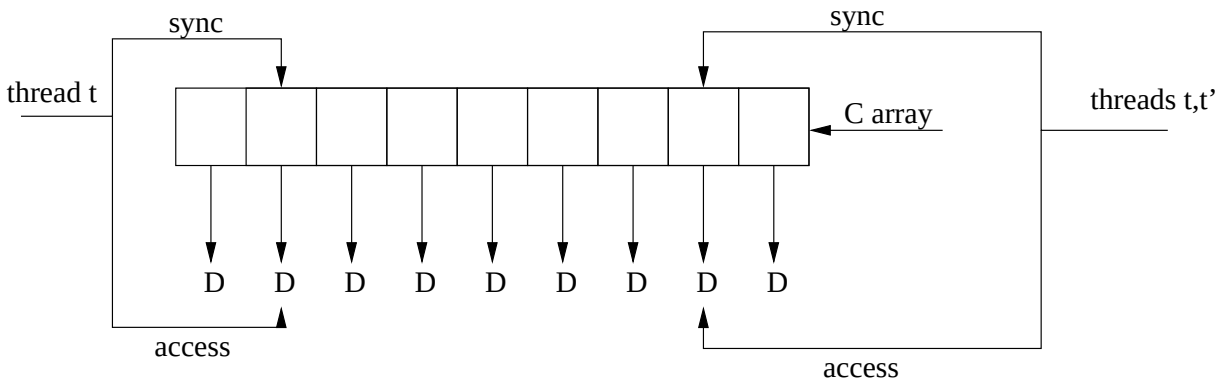
`case A of`

- *ClassId*.f, ok(ME)
- $x.f$
  - $[x.f]_t = [x.f]_{t'}$, ok (ME)
  - $[x.f]_t \neq [x.f]_{t'}$, ok
- x
  - $[x]_t = [x]_{t'}$, ok(ME)
  - $[x]_t \neq [x]_{t'}$, ok if $[x.f]_t \neq [x.f]_{t'}$

Unlocked pairs

Exemple

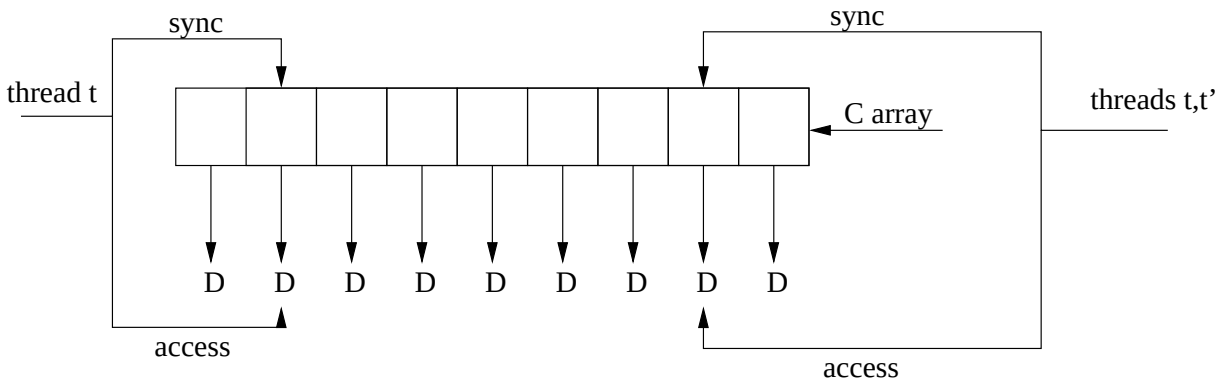
$$\left|\begin{array}{l}\texttt{for}\ (i = 0; i < n; i++)\{ \\ \quad array[i] = \texttt{new}\ C; \\ \quad array[i].f = \texttt{new}\ D\ \}\end{array}\right| \rightarrow \left|\begin{array}{l}\texttt{for}\ (i = 0; i < m; i++)\{ \\ \quad y = array[*]; \\ \quad spawn\{\texttt{sync}(y)\{y.f.val = *;\}\}\}\end{array}\right|$$



Unlocked pairs

Exemple

$$\left|\begin{array}{l}\texttt{for}\ (i = 0; i < n; i++)\{ \\ \quad array[i] = \texttt{new}\ C; \\ \quad array[i].f = \texttt{new}\ D\ \}\end{array}\right| \rightarrow \left|\begin{array}{l}\texttt{for}\ (i = 0; i < m; i++)\{ \\ \quad y = array[*]; \\ \quad spawn\{\texttt{sync}(y)\{y.f.val = *;\}\}\}\end{array}\right|$$



Conditional must-not aliasing

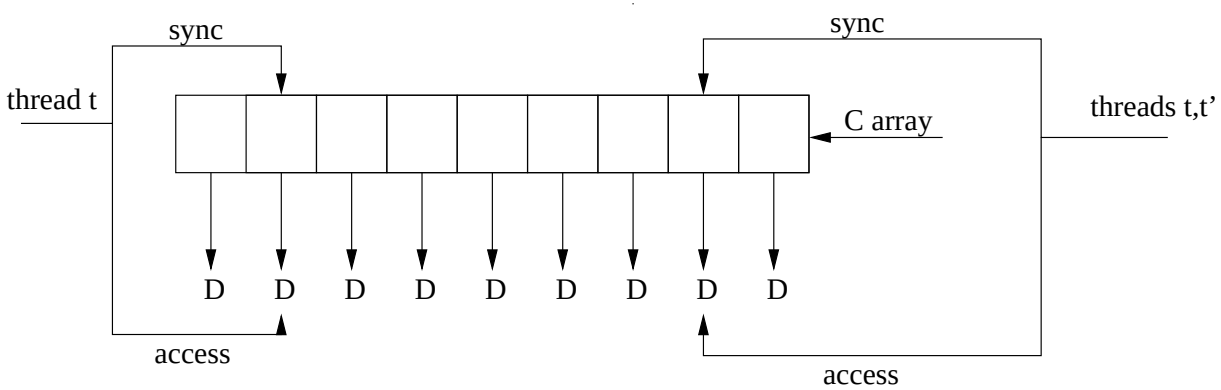
$MustNotAlias(x, y) \Rightarrow MustNotAlias(x.\_.f, y.\_.f)$

Disjoint Reachability

Disjoint Reachability

$$DR(\{a_1, \ldots, a_n\}) = \{a \mid (a \in \bigcap_{i=1,2} Reachable(a_{k_i})) \Rightarrow k_1 = k_2\}$$

Example



$$\{d_1, \ldots, d_n\} \subseteq DR(\{c_1, \ldots, c_n\}) \qquad c_i : C,\ d_i : D$$

Conditionnal must-not aliasing

$$sync(x)\{\ldots x.\_.f = \_\ldots\} \mid sync(y)\{\ldots y.\_.f = \_\ldots\}$$

$$P^a(x.\_.f) \cap P(y.\_.f) \subseteq DR^{\#}(P(x) \cup P(y))$$

[a] Points-to analysis

Calcul de $DR^{\#}$

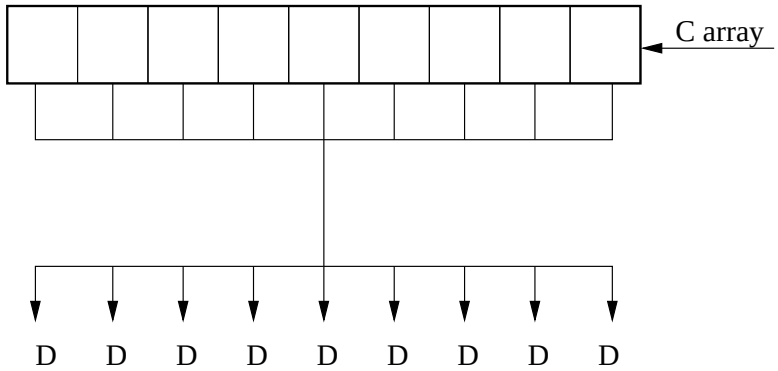
Problème

Distinguer les objets alloués au même site

Calcul de $DR^{\#}$

Problème

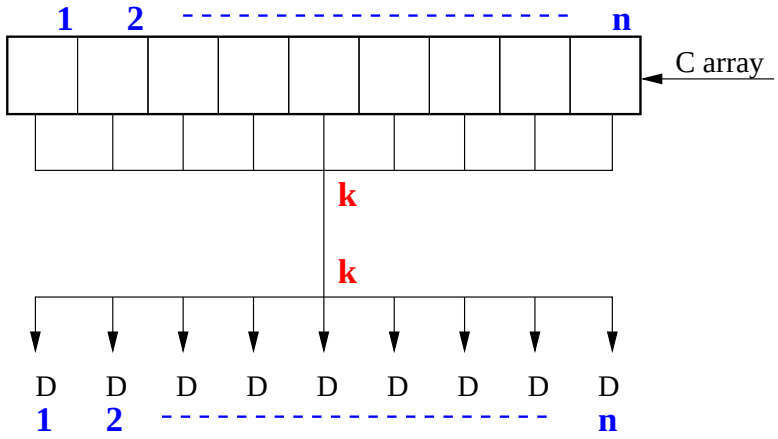
Distinguer les objets alloués au même site



Calcul de $DR^{\#}$

Solution

- Ajouter des compteurs pour distinguer les objets alloués au même site (sémantique **dynamique**).
- Garantir que deux objets sont crées à la même itération (sémantique **statique**).



Dans le papier

- Langage `while`, pas de concurrence/procedure
- Preuve de la correction de la disjoint reachability
- Pas de formalisation du lien avec les dataraces

Nous

- sous-ensemble du bytecode Java
- Procédures et Multithreading
- Preuve de la correction de la disjoint reachability
- Application aux dataraces

Formalisation

Formalisation

Classes

$$\mathit{ClassId} \ni \{\mathit{cid}, \ldots\} \qquad \mathit{Field} \ni \{f, g, h, \ldots\}$$
$$\mathit{MethodId} \ni \{\mathit{mid}, \ldots\} \qquad \mathit{Var} \ni \{x, y, z, \ldots\}$$
$$\mathit{Type} = \mathit{ClassId} \cup \{\texttt{void}\}$$

$$\mathbb{C} \ni \{ \begin{array}{lll} \texttt{name} & \in & \mathit{ClassId}; \\ \texttt{fields} & \subseteq & \mathit{Field}; \\ \texttt{methods} & \subseteq & \mathbb{M}\} \end{array}$$

$$\mathbb{M} \ni \{ \begin{array}{lll} \texttt{name} & \in & \mathit{MethodId}; \\ \texttt{param} & \in & \mathit{ClassId}^n; \\ \texttt{rtype} & \in & \mathit{Type}; \\ \texttt{body} & \in & \mathbb{N} \rightharpoonup \mathit{inst}\} \end{array}$$

Langage

Instructions

$$
\begin{aligned}
\mathit{inst} \quad ::= \quad & \texttt{aconstnull} \mid \texttt{new}\ \ \mathit{cid} \\
& \mid \texttt{aload}\ x \mid \texttt{astore}\ x \\
& \mid \texttt{getfield}\ f \mid \texttt{putfield}\ f \\
& \mid \texttt{invokevirtual}\ \mathit{mid} : (\mathit{ptype})\mathit{rtype} \\
& \mid \texttt{if}\ (\mathit{cond})\ \ell \mid \texttt{goto}\ \ell \\
& \mid \texttt{monitorenter} \mid \texttt{monitorexit} \mid \texttt{start}
\end{aligned}
$$

Sémantique

Compteurs

$p ::= \langle m, i, c, \omega, \pi \rangle$ (code pointer)

- appels de méthode/contexte $\omega : \mathbb{M} \times \mathbb{C} \to \mathbb{N}$
- arcs de flux de contrôle $\pi : \mathbb{M} \times \mathbb{C} \times \mathbb{N}^2 \to \mathbb{N}$

$$\begin{array}{rcccll} \mathbb{O}_\perp & \ni & o & ::= & (a, p) \mid \perp & \text{(address)} \\ & & cs & ::= & (p, s, \rho) :: cs \mid \varepsilon & \text{(call stack)} \end{array}$$

$$s \in \mathbb{O}_\perp \ stack \qquad \rho \in Var \to \mathbb{O}_\perp$$

Etats : $st ::= L, \sigma, \mu, \omega$

$L = \{(cs, o), \ldots, \}$ $\quad \sigma : \mathbb{O} \rightharpoonup \mathbb{F} \to \mathbb{O}_\perp$ $\quad \mu : \mathbb{O} \to \{\texttt{free}\} \cup (\mathbb{O} \times \mathbb{N}^*)$

ω : compteur global

Sémantique

Fonctionnement des compteurs

Allocation

$$(\langle m, i, c, \omega, \pi \rangle, s, \rho) \rightarrow (\langle m, i+1, c, \omega, \pi[(m, c, i, i+1) + +] \rangle, o :: s, \rho)$$
$$o = (a, \langle m, i, c, \omega, \pi \rangle)$$

Sémantique

Fonctionnement des compteurs

Allocation

$$(\langle m, i, c, \omega, \pi\rangle, s, \rho) \longrightarrow (\langle m, i+1, c, \omega, \pi[(m, c, i, i+1) + +]\rangle, o :: s, \rho)$$
$$o = (a, \langle m, i, c, \omega, \pi\rangle)$$

Appel de méthode $(m, c \longrightarrow m', c')$

$$\frac{(\langle m, i, c, \omega, \pi\rangle, s, \rho)}{\vdots} \longrightarrow \frac{\begin{array}{c}(\langle m', 0, c', \omega', \pi'\rangle \\ (\langle m, i+1, c, \omega, \pi[(m, c, i, i+1) + +]\rangle, s, \rho)\end{array}}{\vdots}$$

$$\begin{array}{lcl} \omega_{global} & \longrightarrow & \omega_{global}[(m', c') + +] \\ \omega' & = & \omega[m', c' \mapsto \omega_{global}(m', c')] \\ \pi' & = & \pi[m', c', \_, \_ \mapsto 0] \end{array}$$

Sémantique

Boucles

$$\texttt{while}^{1}(*)\{ \quad x = new_{h_1}\ O; \quad y = new_{h_2}\ O; \quad x.f = y; \quad \}$$

Sémantique

Boucles

$$\texttt{while}^1(*)\{ \quad x = new_{h_1}\ O; \quad y = new_{h_2}\ O; \quad x.f = y; \quad \}$$

$$\mathcal{L} : \mathbb{M} \times \mathbb{N} \rightarrow \mathbb{N}^2$$

$$+$$

Condition **wf**

Sémantique

Boucles

$$\begin{array}{l} \texttt{while}^1(*)\{ \\ \quad x = new_{h_1}\ O; \\ \quad y = new_{h_2}\ O; \\ \quad x.f = y; \\ \} \end{array} \quad \Big| \quad \begin{array}{c} \mathcal{L} : \mathbb{M} \times \mathbb{N} \rightarrow \mathbb{N}^2 \\ + \\ \text{Condition } \mathbf{wf} \end{array}$$

Propriété d'unicité

$$\left(\begin{array}{lcl} (a_i, \langle m, i, c, \omega_i, \pi_i \rangle) & \in & dom(\sigma), i = 1, 2 \\ \omega_1(m, c) & = & \omega_2(m, c) \\ \pi_1(m, c, \mathcal{L}(m, i)) & = & \pi_2(m, c, \mathcal{L}(m, i)) \end{array} \right) \Rightarrow a_1 = a_2$$

Analyse

Abstraction

$$(a, \langle m, i, c, \omega, \pi \rangle) : \tau = \langle m, i, c, \Omega, \Pi \rangle \in \mathbb{T}$$

$$\begin{array}{lcl} \Omega & : & \mathbb{M} \times \mathbb{C} \rightarrow \mathbb{N}_\top \\ \Pi & : & \mathbb{M} \times \mathbb{C} \times \mathbb{N}^2 \rightarrow \mathbb{N}_\top \end{array} \qquad \mathbb{N}_\top = \{\mathbf{1}, \top\}$$

Analyse

Flow insensitive : Σ

$$\Sigma \subseteq \mathbb{T} \times \mathit{Field} \times \mathbb{T}$$

Flow sensitive : $\mathcal{M}, \mathcal{A}$

$$\begin{array}{lcl} \mathcal{M} & : & \mathbb{M} \times \mathbb{C} \rightarrow (\mathcal{P}(\mathbb{T}^n), \mathcal{P}(\mathbb{T}) \cup \{\texttt{void}\}) \\ \mathcal{A} & : & \texttt{ppt} \rightarrow \mathcal{P}(\mathbb{T})\ \mathit{stack} \times (\texttt{var} \rightarrow \mathcal{P}(\mathbb{T})) \end{array}$$

Analyse

Contraintes

(1)	$\left(\begin{array}{l} m.\mathtt{body}(i) = \mathtt{putfield}\ \ f \\ \mathcal{A}(m,i,c) = \phi_2 :: \phi_1 :: S, \Gamma \end{array} \right) \Rightarrow \{(\tau_1, f, \tau_2) \mid \tau_i \in \phi_i\} \subseteq \Sigma$
(2)	$\mathcal{A}(m,0,c) = \mathit{kill}_{m,c}(\mathcal{A}(m,0,c))$
(3)	$\mathit{CFlow}(m,i,j) \Rightarrow \mathit{kill}_{m,c,i,j}([m.\mathtt{body}(i)]^{\#}_{m,i,c}(\mathcal{A}(m,i,c))) \sqsubseteq \mathcal{A}(m,j,c)$

Analyse

Fonctions de transfert

$$[\texttt{new}\ \ C]^{\#}_{m,i,c}(S,\Gamma) = \phi :: S,\Gamma$$
$$\phi = \{\langle m, i, c, \lambda\_.1,\ \ \lambda\_.1\rangle\}$$

$$[\texttt{getfield}\ \ f]^{\#}_{m,i,c}(\phi :: S,\Gamma) = \phi' :: S,\Gamma$$
$$\phi' = \{\langle m', i', c', \omega_\top, \pi_\top\rangle \mid (\tau, f, \langle m', i', c', \_,\_\rangle) \in \Sigma,\ \tau \in \phi\}$$

Analyse

Disjoint Reachability

$$(m,i,c) \in DR_\sigma(A) \Leftrightarrow \left(\begin{array}{c} (o_i, o) \in Reach^+(\sigma),\ i = 1,2 \\ o.\mathbf{cp} = (m,i,c) \\ o_1.\mathbf{cp},\ o_2.\mathbf{cp} \in A \end{array} \right) \Rightarrow \quad o_1 = o_2$$

$$(m,i,c) \in DR_\Sigma(A) \Leftrightarrow \left(\begin{array}{c} (\tau_1, \tau_3) \in \Sigma^+ \\ (\tau_2, \tau_4) \in \Sigma^+ \\ \tau_3.\mathbf{cp} = (m,i,c) \\ \tau_4.\mathbf{cp} = (m,i,c) \\ \tau_1.\mathbf{cp},\ \tau_2.\mathbf{cp} \in A \end{array} \right) \Rightarrow \left(\begin{array}{l} \tau_1 = \tau_2 \\ \tau_1.\mathbf{cp} = (m',i,c') \\ \tau_1.\mathbf{\Omega}(m',c') = \tau_1.\mathbf{\Pi}(m',c',\mathcal{L}(m',i')) = \mathbf{1} \\ \tau_3.\mathbf{\Omega}(m',c') = \tau_3.\mathbf{\Pi}(m',c',\mathcal{L}(m',i')) = \mathbf{1} \\ \tau_4.\mathbf{\Omega}(m',c') = \tau_4.\mathbf{\Pi}(m',c',\mathcal{L}(m',i')) = \mathbf{1} \end{array} \right)$$

Correction

$$\sigma \textbf{ accessible} \Rightarrow (m,i,c) \in DR_\sigma(A) \Rightarrow (m,i,c) \in DR_\Sigma(A)$$

Analyse

Example

$$
\begin{array}{ll}
\mathit{while}^1(*)\{ & \\
\quad x = \mathit{new}_{h_1}\ O; & x \mapsto (h_1, \langle \top, \top \rangle),\ y \mapsto (h_2, \langle \top, \top \rangle) \\
\quad \mathit{while}^2(*)\{ & \\
\quad y = \mathit{new}_{h_2}\ O; & x \mapsto (h_1, \langle 1, \top \rangle),\ y \mapsto (h_2, \langle \top, \top \rangle) \\
\quad x.f = y; & x \mapsto (h_1, \langle 1, \top \rangle),\ y \mapsto (h_2, \langle 1, 1 \rangle) \\
\quad \} & \\
\} & (h_1, \langle 1, \top \rangle) \rightarrow (h_2, \langle 1, 1 \rangle)
\end{array}
$$

Analyse

Exemple

$$
\begin{array}{ll}
m^1(\{ & \\
\quad x = new_{h_1}\ O; & x \mapsto \ldots \\
\quad n(x); & x \mapsto (h_1, \langle 1,1,1 \rangle) \\
\} & (*) \\
n^2(x)\{ & \\
\quad y = new_{h_2}\ O; & x \mapsto (h_1, \langle 1,\top,1 \rangle),\ y \mapsto \ldots \\
\quad setfield(x,y); & x \mapsto (h_1, \langle 1,\top,1 \rangle),\ y \mapsto (h_2, \langle 1,1,1 \rangle) \\
\} & (*) \\
setfield^3(x,y)\{ & \\
\quad x.f = y; & x \mapsto (h_1, \langle 1,\top,\top \rangle),\ y \mapsto (h_2, \langle 1,1,\top \rangle) \\
\} & (h_1, \langle 1,\top,\top \rangle) \rightarrow (h_2, \langle 1,1,\top \rangle) \quad (*)
\end{array}
$$

Dataraces

Dataraces

Actions / Exécutions

$$e = (a, \mathcal{R}, a') \qquad \mathcal{R} \in \{?_f, !_f, \nearrow, \downarrow, \uparrow\}$$

$$E = st_0 \xrightarrow{e_0} st_1 \xrightarrow{e_1} \ldots \xrightarrow{e_{n-1}} e_n$$

Conflit

$$\frac{\mathcal{R} \in \{?, !\}}{a'?a \not\leftrightarrows a''\mathcal{R}a} \qquad \not\leftrightarrows \text{ sym.}$$

Datarace

$$DRF(E) \Leftrightarrow \begin{array}{c} \forall i < j \in \{0, 1, \ldots, n-1\}. \\ e_i \not\leftrightarrows e_j \Rightarrow e_i \preceq_{hb} e_j \end{array}$$

Sémantique

happens-before

$$\frac{i \leq j \qquad e_i = a \mathcal{R} a' \qquad e_j = a \mathcal{R} a''}{e_i \leq^0_{hb} e_j}$$

$$\frac{i \leq j \qquad e_i = a \nearrow a' \qquad e_j = a' \mathcal{R} a''}{e_i \leq^0_{hb} e_j}$$

$$\frac{i \leq j \qquad e_i = a' \downarrow a \qquad e_j = a'' \uparrow a}{e_i \leq^0_{hb} e_j}$$

$\leq_{hb}$: clotûre transitive de $\leq^0_{hb}$

N.B. : Notion de race mieux adaptée que le choix non déterministe.

Dataraces

Elimination

$$F_\Sigma(R) = R - \{(B_1, B_2) \mid \mathcal{P}(B_1) \cap \mathcal{P}(B_2) \subseteq DR_\Sigma(\mathcal{P}(A_1) \cup \mathcal{P}(A_2))\}$$

$$sync(A_i)\{\ldots B_i \ldots\} \qquad B_i \in Reachable(A_i) \qquad i = 1, 2$$

Dataraces

Elimination

$$F_\Sigma(R) = R - \{(B_1, B_2) \mid \mathcal{P}(B_1) \cap \mathcal{P}(B_2) \subseteq DR_\Sigma(\mathcal{P}(A_1) \cup \mathcal{P}(A_2))\}$$

$$sync(A_i)\{\ldots B_i \ldots\} \qquad B_i \in Reachable(A_i) \qquad i = 1, 2$$

Correction

$$Safe(R) \Rightarrow Safe(F_\Sigma(R))$$

Pour la suite ...

Pour la suite ...

Activité 1

- Certification Coq (en cours)
- Gérer les autres étapes

Pour la suite ...

Activité 1

- Certification Coq (en cours)
- Gérer les autres étapes

Activité 2

- Cadre d'interprétation abstraite pour l'analyse de Bytecode Java bien synchronisés (en cours)
  - Null-pointer
  - Points-to
  - Must-alias
  - Polyèdres

Pour la suite ...

Activité 1

- Certification Coq (en cours)
- Gérer les autres étapes

Activité 2

- Cadre d'interprétation abstraite pour l'analyse de Bytecode Java bien synchronisés (en cours)
  - Null-pointer
  - Points-to
  - Must-alias
  - Polyèdres
- Application à la gestion des ressources MIDP (Extension du papier ESORICS)

Pour la suite ...

Activité 1

- Certification Coq (en cours)
- Gérer les autres étapes

Activité 2

- Cadre d'interprétation abstraite pour l'analyse de Bytecode Java bien synchronisés (en cours)
  - Null-pointer
  - Points-to
  - Must-alias
  - Polyèdres
- Application à la gestion des ressources MIDP (Extension du papier ESORICS)
- Certification Coq