

Formalisation d'analyses statiques pour Java multithreadé

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

¹INRIA, 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

- 1 Naik & Aiken, POPL'07
- 2 Formalisation
- 3 Dataraces
- 4 Conclusion

Naik & Aiken, POPL'07

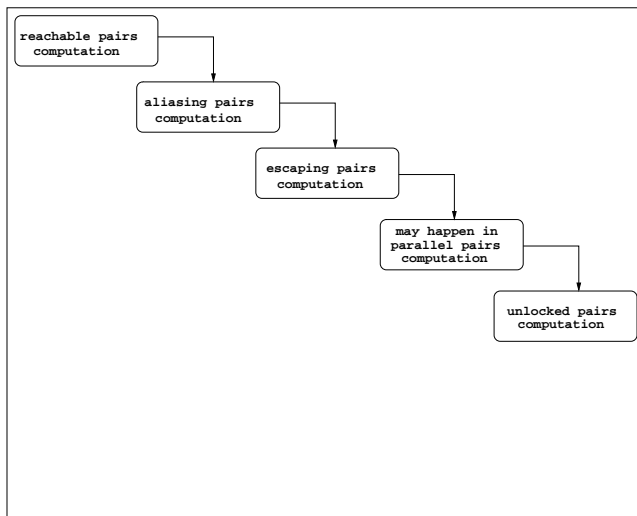
Naik & Aiken, POPL'07

- 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

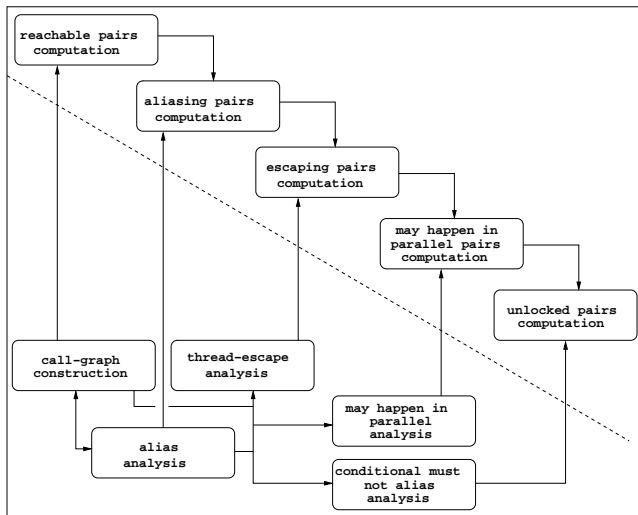
^apoints de programme

^banalyse statique

Etapas



Etapas



Unlocked pairs

Exemple

| thread t | thread t' |
|--|--|
| $\text{sync}(A)\{$ \dots $x.f.g = -; \}$ | $\text{sync}(A)\{$ \dots $x.f.g = -; \}$ |

case A of

Unlocked pairs

Exemple

| thread t | thread t' |
|--|--|
| $\text{sync}(A)\{$ \dots $x.f.g = -; \}$ | $\text{sync}(A)\{$ \dots $x.f.g = -; \}$ |

case A of

- $\text{ClassId}.f$, $\text{ok}(\text{ME})$

Unlocked pairs

Exemple

| thread t | thread t' |
|--|--|
| $\text{sync}(A)\{$ \dots $x.f.g = -; \}$ | $\text{sync}(A)\{$ \dots $x.f.g = -; \}$ |

case A of

- $\text{ClassId}.f$, 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' |
|--|--|
| $\text{sync}(A)\{$ \dots $x.f.g = -; \}$ | $\text{sync}(A)\{$ \dots $x.f.g = -; \}$ |

case A of

• x

• $\text{ClassId}.f$, ok(ME)

• $x.f$

• $[x.f]_t = [x.f]_{t'}$, ok (ME)

• $[x.f]_t \neq [x.f]_{t'}$, ok

• $[x]_t = [x]_{t'}$, ok(ME)

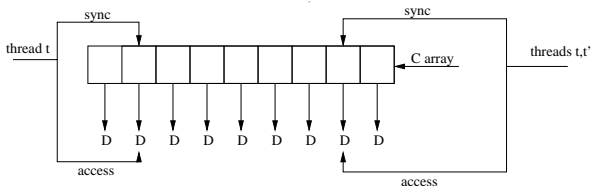
• $[x]_t \neq [x]_{t'}$, ok if
 $[x.f]_t \neq [x.f]_{t'}$

Unlocked pairs

Exemple

```
for (i = 0; i < n; i++) {
  array[i] = new C;
  array[i].f = new D }
→
```

```
for (i = 0; i < m; i++) {
  y = array[*];
  spawn { sync(y) { y.f.val = *; } } }
```

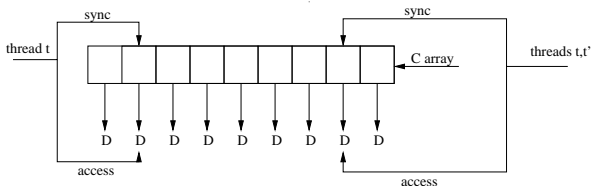


Unlocked pairs

Exemple

```
for (i = 0; i < n; i++) {
  array[i] = new C;
  array[i].f = new D }
→
```

```
for (i = 0; i < m; i++) {
  y = array[*];
  spawn { sync(y) { y.f.val = *; } } }
```



Conditional must-not aliasing

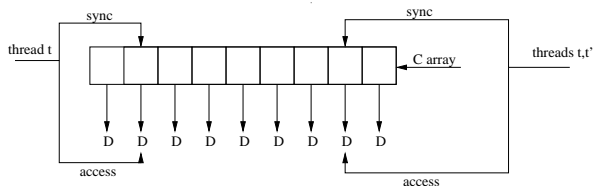
$MustNotAlias(x, y) \Rightarrow MustNotAlias(x...f, y...f)$

Disjoint Reachability

Disjoint Reachability

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

Example



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

Conditionnal must-not aliasing

$$\text{sync}(x)\{\dots x \dots f = \dots\} \mid \text{sync}(y)\{\dots y \dots f = \dots\}$$

$$P^a(x \dots f) \cap P(y \dots f) \subseteq DR^\#(P(x) \cup P(y))$$

^aPoints-to analysis

Calcul de $DR^\#$

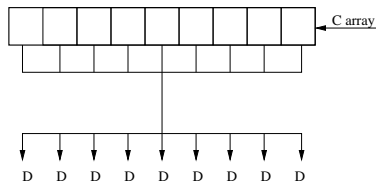
Problème

Distinguer les objets alloués au même site

Calcul de $DR^\#$

Problème

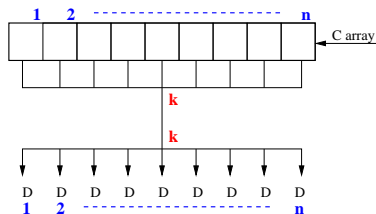
Distinguer les objets alloués au même site



Calcul de $DR^\#$

Solution

- Ajouter des compteurs pour distinguer les objets alloués au même site (sémantique **dynamique**).
- Garantir que deux objets sont créés 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

$$\begin{aligned}
 \text{ClassId} &\ni \{cid, \dots\} & \text{Field} &\ni \{f, g, h, \dots\} \\
 \text{MethodId} &\ni \{mid, \dots\} & \text{Var} &\ni \{x, y, z, \dots\} \\
 \text{Type} &= \text{ClassId} \cup \{\text{void}\}
 \end{aligned}$$

$$\mathbb{C} \ni \left\{ \begin{array}{ll} \text{name} & \in \text{ClassId}; \\ \text{fields} & \subseteq \text{Field}; \\ \text{methods} & \subseteq \mathbb{M} \end{array} \right\}$$

$$\mathbb{M} \ni \left\{ \begin{array}{ll} \text{name} & \in \text{MethodId}; \\ \text{param} & \in \text{ClassId}^n; \\ \text{rtype} & \in \text{Type}; \\ \text{body} & \in \mathbb{N} \rightarrow \text{inst} \end{array} \right\}$$

Langage

Instructions

```
inst ::=  aconstnull | new cid  
        |  aload x | astore x  
        |  getfield f | putfield f  
        |  invokevirtual mid : (ptype)rtype  
        |  if (cond) l | goto l  
        |  monitorenter | monitorexit | start
```

Sémantique

Compteurs

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

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

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

$$s \in \mathbb{O}_\perp \text{ stack} \quad \rho \in \text{Var} \rightarrow \mathbb{O}_\perp$$

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

$$L = \{(cs, o), \dots, \} \quad \sigma : \mathbb{O} \rightarrow \mathbb{F} \rightarrow \mathbb{O}_\perp \quad \mu : \mathbb{O} \rightarrow \{\text{free}\} \cup (\mathbb{O} \times \mathbb{N}^*)$$

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

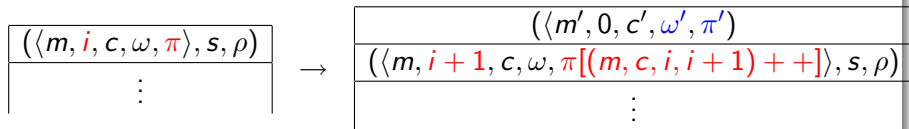
Fonctionnement des compteurs

Allocation

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

$$o = (a, \langle m, i, c, \omega, \pi \rangle)$$

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



$$\omega_{global} \rightarrow \omega_{global}[(m', c') ++]$$

$$\omega' = \omega[m', c' \mapsto \omega_{global}(m', c')]$$

$$\pi' = \pi[m', c', -, - \mapsto 0]$$

Sémantique

Boucles

```
while1(*){  
    x = newh1 O;  
    y = newh2 O;  
    x.f = y;  
}
```

Sémantique

Boucles

```

while1(*){
  x = newh1 O;
  y = newh2 O;
  x.f = y;
}

```

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

+

Condition **wf**

Sémantique

Boucles

```

while1(*){
  x = newh1 O;
  y = newh2 O;
  x.f = y;
}

```

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

+

Condition **wf**

Propriété d'unicité

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

Analyse

Abstraction

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

$$\begin{array}{l} \Omega : \mathbb{M} \times \mathbb{C} \rightarrow \mathbb{N}_T \\ \Pi : \mathbb{M} \times \mathbb{C} \times \mathbb{N}^2 \rightarrow \mathbb{N}_T \end{array} \quad \mathbb{N}_T = \{\mathbf{1}, \top\}$$

Analyse

Flow insensitive : Σ

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

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

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

Analyse

Contraintes

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

Analyse

Fonctions de transfert

$$[\text{new } C]_{m,i,c}^\#(S, \Gamma) = \phi :: S, \Gamma$$

$$\phi = \{\langle m, i, c, \lambda_{-1}, \lambda_{-1} \rangle\}$$

$$[\text{getfield } f]_{m,i,c}^\#(\phi :: S, \Gamma) = \phi' :: S, \Gamma$$

$$\phi' = \{\langle m', i', c', \omega_T, \pi_T \rangle \mid (\tau, f, \langle m', i', c', -, - \rangle) \in \Sigma, \tau \in \phi\}$$

Analyse

Disjoint Reachability

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

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

Correction

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

Analyse

Example

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

Analyse

Example

$$\begin{array}{l}
 m^1()\{ \\
 \quad x = \text{new}_{h_1} O; \quad x \mapsto \dots \\
 \quad n(x); \quad x \mapsto (h_1, \langle 1, 1, 1 \rangle) \\
 \} \quad (*) \\
 n^2(x)\{ \\
 \quad y = \text{new}_{h_2} O; \quad x \mapsto (h_1, \langle 1, \top, 1 \rangle), y \mapsto \dots \\
 \quad \text{setfield}(x, y); \quad x \mapsto (h_1, \langle 1, \top, 1 \rangle), y \mapsto (h_2, \langle 1, 1, 1 \rangle) \\
 \} \quad (*) \\
 \text{setfield}^3(x, y)\{ \\
 \quad x.f = y; \quad x \mapsto (h_1, \langle 1, \top, \top \rangle), y \mapsto (h_2, \langle 1, 1, \top \rangle) \\
 \} \quad (h_1, \langle 1, \top, \top \rangle) \rightarrow (h_2, \langle 1, 1, \top \rangle) \quad (*)
 \end{array}$$

Dataraces

Dataraces

Actions / Exécutions

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

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

Conflit

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

Datarace

$$DRF(E) \Leftrightarrow \forall i < j \in \{0, 1, \dots, n-1\}. \\ e_i \not\approx e_j \Rightarrow e_i \preceq_{hb} e_j$$

Sémantique

happens-before

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

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

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

\leq_{hb} : clotûre transitive de \leq_{hb}^0

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

Dataraces

Elimination

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

$$\text{sync}(A_i)\{\dots B_i \dots\} \quad B_i \in \text{Reachable}(A_i) \quad i = 1, 2$$

Dataraces

Elimination

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

$$\text{sync}(A_i)\{\dots B_i \dots\} \quad B_i \in \text{Reachable}(A_i) \quad i = 1, 2$$

Correction

$$\text{Safe}(R) \Rightarrow \text{Safe}(F_{\Sigma}(R))$$

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