Controlling mobility

Hanne Riis Nielson & Flemming Nielson

Informatics & Mathematical Modelling
Technical University of Denmark

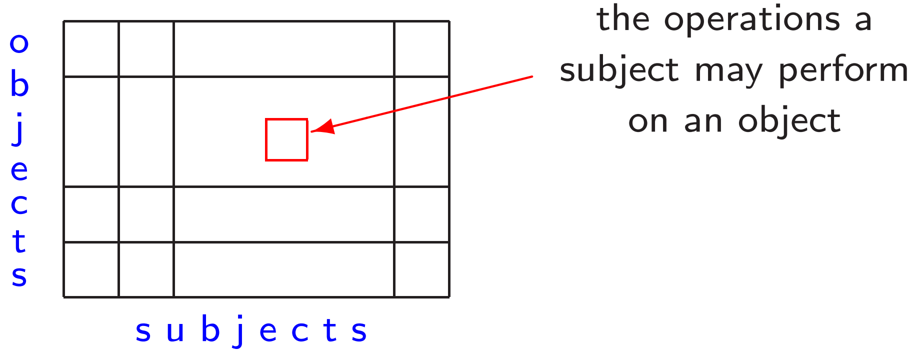
Overview

- Discretionary access control
- Static validation of access control
- Dynamic enforcement of access control
- Mandatory access control

Discretionary access control

An active subject accesses a passive object with some specific access operations.

The access control matrix is the traditional way of defining what operations may be performed:



Access control for mobility

- the subjects are the ambients possessing the capabilities: the ambient may move into another ambient, may move out of another ambient or may dissolve another ambient
- the objects are the ambients that are entered, left or dissolved
- the access operations are the capabilities

Static validation of access control

- Crossing control
- Opening control
- More powerful analyses
  - sets of configurations and global counting
  - sets of configurations and local counting
  - abstract transition systems

Static validation of access control

Validation of properties of the access operations with respect to an "access control policy":

- The program analysis already approximates the set of access operations that an ambient may perform:
  - crossing analysis: may an ambient move into another ambient?
  - opening analysis: may an ambient dissolve another ambient?

Similar properties are expressed by the type systems for mobility developed by Luca Cardelli, Giorgio Ghelli and Andrew Gordon (see POPL'99, ICALP'99 and TCS'99).

Crossing control

Property: The ambient n can cross the ambient n′ during the execution of P, i.e.

– n executes the in n′ or the out n′ capability

In terms of groups: Ambients of group μ can cross ambients in group μ' during the execution of P i.e.

– some n of group μ can cross some ambient n′ of group μ'

In the analysis: Ambients of group μ can cross ambients of group μ' during the execution of P, i.e.

– $\mathsf{in}\,\mu' \in \mathcal{I}(\mu) \vee \mathsf{out}\,\mu' \in \mathcal{I}(\mu)$

for the least $\mathcal{I}$ such that $\mathcal{I} \models^{\star}_{\Gamma} P$

Example: a packet on a network

The analysis can be used to validate:

- Ambients of group $\mathbb{P}$ may cross ambients in group $\mathbb{S}$
  – check: in $\mathbb{S} \in \mathcal{I}(\mathbb{P}) \ \vee\ $ out $\mathbb{S} \in \mathcal{I}(\mathbb{P})$
- Ambients in group $\mathbb{S}$ will never cross ambients in group $\mathbb{P}$
  – check: in $\mathbb{P} \notin \mathcal{I}(\mathbb{S}) \ \wedge\ $ out $\mathbb{P} \notin \mathcal{I}(\mathbb{S})$

A more precise analysis is needed to validate:

- Ambients of group $\mathbb{S}$ will never cross ambients in group $\mathbb{S}$
  we do not have: in $\mathbb{S} \notin \mathcal{I}(\mathbb{S}) \wedge$ out $\mathbb{S} \notin \mathcal{I}(\mathbb{S})$

Analysis estimate:

Γ	**Name** $\rightarrow$ **Group**
A	$\mathbb{S}$
B	$\mathbb{S}$
p	$\mathbb{P}$

$\mathcal{I}$	**Group** $\rightarrow \mathcal{P}($**Group** $\cup$ **Cap**$)$
$\star$	$\{\mathbb{S}, \mathbb{P}\}$
$\mathbb{S}$	$\{\mathbb{P}, \mathbb{S}$, in $\mathbb{S}$, out $\mathbb{S}$, open $\mathbb{P}\}$
$\mathbb{P}$	$\{$in $\mathbb{S}$, out $\mathbb{S}\}$

Opening control

Property: The ambient n can open the ambient n′ during the execution of P, i.e.

– n executes the open n′ capability

In terms of groups: Ambients of group μ can open ambients in group μ' during the execution of P i.e.

– some n of group μ can open some n′ of group μ'

In the analysis: Ambients of group μ may open ambients in group μ', i.e.

– open $\mu' \in \mathcal{I}(\mu)$

for the least $\mathcal{I}$ such that $\mathcal{I} \models^{\star}_{\Gamma} P$

Example: a packet on a network

The analysis can be used to validate:

- Ambients of group $\mathbb{S}$ may open ambients in group $\mathbb{P}$
  – check: open $\mathbb{P} \in \mathcal{I}(\mathbb{S})$
- Ambients in group $\mathbb{P}$ will never open any ambients
  – check: $\forall \mu$: open $\mu \notin \mathcal{I}(\mathbb{P})$

Analysis estimate:

Γ	**Name** $\rightarrow$ **Group**
A	$\mathbb{S}$
B	$\mathbb{S}$
p	$\mathbb{P}$

$\mathcal{I}$	**Group** $\rightarrow \mathcal{P}($**Group** $\cup$ **Cap**$)$
$\star$	$\{\mathbb{S}, \mathbb{P}\}$
$\mathbb{S}$	$\{\mathbb{P}, \mathbb{S}$, in $\mathbb{S}$, out $\mathbb{S}$, open $\mathbb{P}\}$
$\mathbb{P}$	$\{$in $\mathbb{S}$, out $\mathbb{S}\}$

Weaknesses of the analysis so far

$\mathcal{I} \models^{\star}_{\Gamma} P$ for

$$\mathcal{I} : \mathbf{Group} \to \mathcal{P}(\mathbf{Group} \cup \mathbf{Cap})$$

means that

- if $\mu' \in \mathcal{I}(\mu)$ then an ambient of group μ' may turn up inside an ambient of group μ in some configuration obtained by executing P
- it cannot capture that ambients of groups μ' and μ'' both may turn up inside an ambient of group μ but not at the same time

Problem: all configurations are represented by the same abstract configuration — there is only one $\mathcal{I}$

Sets of configurations & global counting

$\{\mathcal{I}_1, \cdots, \mathcal{I}_k\} \models^{\star}_{\Gamma} P$ for

$$\mathcal{I}_i : \mathbf{Group} \rightarrow ((\mathbf{Group} \cup \mathbf{Cap}) \rightarrow \{0, 1, {}^{\geq}2\})$$

means that

- $\mathcal{I}_1, \cdots, \mathcal{I}_k$ represent different sets of configurations
- $\{0, 1, {}^{\geq}2\}$ adds more precision:
  - $\mathcal{I}(\mu)(\mu') = 0$: there are no occurrences of μ' inside μ
  - $\mathcal{I}(\mu)(\mu') = 1$: there is exactly one occurrence of μ' inside μ
  - $\mathcal{I}(\mu)(\mu') = {}^{\geq}2$: there are more than one occurrence of μ' inside μ

  and similarly for capabilities

Why does it add precision?

$\{\mathcal{I}_1, \cdots, \mathcal{I}_k\} \models^{\star}_{\Gamma} P$ for

$$\mathcal{I}_i : \mathbf{Group} \to ((\mathbf{Group} \cup \mathbf{Cap}) \to \{0, 1, {}^{\geq}2\})$$

means that the analysis can distinguish between several possibilities, e.g.

- when there are ${}^{\geq}2$ occurrences of an ambient from group μ' inside group μ $(\mathcal{I}(\mu)(\mu') =^{\geq} 2)$ and it is moved then
  - there might be just 1 occurrence of μ' left (originally there were 2)
  - there might be ${}^{\geq}2$ occurrences left (originally there were 3 or more)

  The analysis keeps track of both possibilities by using different abstract configurations: $\mathcal{I}_1(\mu)(\mu') = 1$ and $\mathcal{I}_2(\mu)(\mu') =^{\geq} 2$
- when there is just 1 occurrence of an ambient in group μ' left inside μ $(\mathcal{I}(\mu)(\mu') = 1)$ and it is moved then there will be 0 left: $\mathcal{I}_3(\mu)(\mu') = 0$

Example: a packet on a network

$$\{\mathcal{I}_1, \mathcal{I}_2, \mathcal{I}_3, \mathcal{I}_4\} \models^{\star}_{\Gamma} \mathsf{A}\,[\mathsf{p}\,[\mathsf{out\,A.\ in\,B}\ ]\ ]\ \mid\ \mathsf{B}\,[\mathsf{open\,p}]$$

Γ	
A	$\mathbb{S}$
B	$\mathbb{S}$
p	$\mathbb{P}$

$\mathcal{I}_1$	
$\star$	$\mathbb{S}$:$^{\geq}2$
$\mathbb{S}$	$\mathbb{P}$:1; open $\mathbb{P}$:1
$\mathbb{P}$	in $\mathbb{S}$:1; out $\mathbb{S}$:1

$\mathcal{I}_2$	
$\star$	$\mathbb{S}$:$^{\geq}2$; $\mathbb{P}$:1
$\mathbb{S}$	open $\mathbb{P}$:1
$\mathbb{P}$	in $\mathbb{S}$:1

$\mathcal{I}_3$	
$\star$	$\mathbb{S}$:$^{\geq}2$
$\mathbb{S}$	$\mathbb{P}$:1; open $\mathbb{P}$:1
$\mathbb{P}$	

$\mathcal{I}_4$	
$\star$	$\mathbb{S}$:$^{\geq}2$
$\mathbb{S}$	
$\mathbb{P}$	

As with the simple analysis we can validate that

- ambients in $\mathbb{P}$ may cross ambients in $\mathbb{S}$
- ambients in $\mathbb{S}$ will never cross ambients in $\mathbb{P}$

Additionally we can validate that

- ambients in $\mathbb{S}$ will never cross ambients in $\mathbb{S}$

Weaknesses of the analysis so far

$\{\mathcal{I}_1, \cdots, \mathcal{I}_k\} \models^{\star}_{\Gamma} P$ for

$$\mathcal{I}_i : \mathbf{Group} \to ((\mathbf{Group} \cup \mathbf{Cap}) \to \{0, 1, {}^{\geq}2\})$$

means that

- if $\mathcal{I}(\mu)(\mu') = N$ then N occurrences of an ambient of group μ' may turn up inside an ambient of group μ in some configuration obtained by executing P
- it (still) cannot capture that ambients of groups μ' and μ'' both may turn up inside an ambient of group μ but that no occurrence of an ambient in a group μ contains more than one occurrence of ambients in group μ'

The analysis is context insensitive and flow insensitive

Sets of configurations & local counting

- the configurations of ambient processes have a tree structure that during the execurion may evolve into an arbitrarily wide and arbitrarily deep tree structure
- an extension of regular tree grammars called shape grammars can be used to give a finite representation of a configuration by
  - adding abstract counts to ensure finite branching
  - adding context information to give a finer distinction between ambients within the same group

Judgements of the analysis has the form $\{\mathcal{G}_1, \cdots, \mathcal{G}_k\} \models^{\star}_{\Gamma} P$ where $\mathcal{G}_1, \cdots, \mathcal{G}_k$ are shape grammars describing the abstract configurations.

See Nielson & Nielson in POPL'00, Nordic Journal of Computing 2001

Abstract transition systems

- the abstract configurations $\{\mathcal{I}_1, \cdots, \mathcal{I}_k\}$ describe the set of configurations
- we may also want to keep track of an abstraction of the transition relation:
  - $\mathcal{I}_i \blacktriangleright \mathcal{I}_j$ means that configurations described by $\mathcal{I}_i$ may evolve into configurations described by $\mathcal{I}_j$

Judgements of the analysis has the form $(\{\mathcal{I}_1, \cdots, \mathcal{I}_k\}, \blacktriangleright, \mathcal{I}_1) \models^{\star}_{\Gamma} P$ where $\mathcal{I}_1$ describes the initial configuration.

Dynamic enforcement of access control

- Introducing access control primitives
  - safe ambients
  - discretionary ambients
- Adapting the program analysis
  - taking co-capabilties into account
  - taking context into account

Safe ambients

Movement of ambients can only happen if both parties agree:

- if p wants to move out of A then A should be willing to let ambients leave
  - A must have the capability $\overline{\mathsf{out}}$ A
- if p wants to move into B then B should be willing to let ambients enter
  - B must have the capability $\overline{\mathsf{in}}$ B
- if B wants to dissolve p then p should be willing to be dissolved
  - p must have the capability $\overline{\mathsf{open}}$ p

Safe ambients were introduced by Levi and Sangiorgi at POPL 2000.

Syntax and semantics

$$\begin{array}{lllrcl} M & ::= & \mathsf{in}\ n \mid \mathsf{out}\ n \mid \mathsf{open}\ n & \text{capabilities} & \approx & \text{access operations} \\ & \mid & \overline{\mathsf{in}}\ n \mid \overline{\mathsf{out}}\ n \mid \overline{\mathsf{open}}\ n & \text{co-capabilities} & \approx & \text{access rights} \end{array}$$

Extensions to the transition relation:

$$n\,[\mathsf{in}\,m.\,P \mid Q\,] \mid m\,[\,\overline{\mathsf{in}}\ m.\,R \mid S\,] \;\rightarrow\; m\,[n\,[P \mid Q\,] \mid R \mid S\,]$$

$$m\,[n\,[\mathsf{out}\,m.\,P \mid Q\,] \mid \overline{\mathsf{out}}\ m.\,R \mid S\,] \;\rightarrow\; n\,[P \mid Q\,] \mid m\,[R \mid S\,]$$

$$\mathsf{open}\,n.\,P \mid n\,[\,\overline{\mathsf{open}}\ n.\,Q \mid R] \;\rightarrow\; P \mid Q \mid R$$

This amounts to integrating the reference monitor into the operational semantics.

In pictures

Move into an ambient: $\overset{n}{\boxed{\mathsf{in}\ m.P \mid Q}} \;\Big|\; \overset{m}{\boxed{\overline{\mathsf{in}}\ m.R \mid S}} \longrightarrow \overset{m}{\boxed{\overset{n}{\boxed{P \mid Q}} \;\Big|\; R \mid S}}$

Move out of an ambient: $\overset{m}{\boxed{\overset{n}{\boxed{\mathsf{out}\ m.P \mid Q}} \;\Big|\; \overline{\mathsf{out}}\ m.R \mid S}} \longrightarrow \overset{n}{\boxed{P \mid Q}} \;\Big|\; \overset{m}{\boxed{R \mid S}}$

Dissolve an ambient: $\mathsf{open}\ n.P \;\Big|\; \overset{n}{\boxed{\overline{\mathsf{open}}\ n.Q \mid R}} \longrightarrow P \;\Big|\; Q \mid R$

Example: a package on a network

$$\mathsf{A}[\,\mathsf{p}[\,\mathsf{out}\,\mathsf{A}.\,\mathsf{in}\,\mathsf{B}.\,\overline{\mathsf{open}}\;\mathsf{p}\,]\;|\;\overline{\mathsf{out}}\;\mathsf{A}\,]\;|\;\mathsf{B}[\,\overline{\mathsf{in}}\;\mathsf{B}.\,\mathsf{open}\,\mathsf{p}\,]$$

$$\longrightarrow\;\mathsf{A}[\;]\;|\;\mathsf{p}[\,\mathsf{in}\,\mathsf{B}.\,\overline{\mathsf{open}}\;\mathsf{p}\,]\;|\;\mathsf{B}[\,\overline{\mathsf{in}}\;\mathsf{B}.\,\mathsf{open}\,\mathsf{p}\,]$$

$$\longrightarrow\;\mathsf{A}[\;]\;|\;\mathsf{B}[\,\mathsf{p}[\,\overline{\mathsf{open}}\;\mathsf{p}\,]\;|\;\mathsf{open}\,\mathsf{p}\,]$$

$$\longrightarrow\;\mathsf{A}[\;]\;|\;\mathsf{B}[\;]$$