Propositional Dynamic Logic and Concurrency

Matteo Acclavio

UNIVERSITY OF SUSSEX

Joint Work With Fabrizio Montesi and Marco Peressotti

Copenhagen 24/06/2024

Motivations

Last year, in Odense

VILLUM FONDEN

“X-IDF: Explainable Internet Data Flows”

Mission: empowering citizens in gaining agency about their private data by build a technology that is accessible and actionable.

Last year, in Odense

VILLUM FONDEN

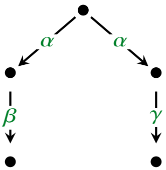
"X-IDF: Explainable Internet Data Flows"

Mission: empowering citizens in gaining agency about their private data by build a technology that is accessible and actionable.

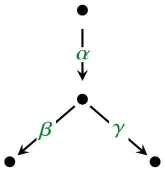
Goal: find a way to distinguish protocols
(w.r.t. transmitted private data)

Bisimulation as program equivalence

$\pi_1 = (\alpha;\beta) + (\alpha;\gamma)$ $\qquad$ $\pi_2 = \alpha;(\beta + \gamma)$

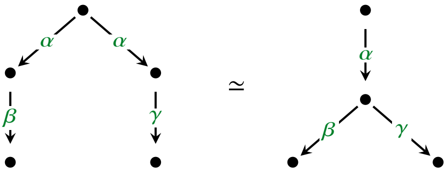


$\simeq$



Bisimulation as program equivalence

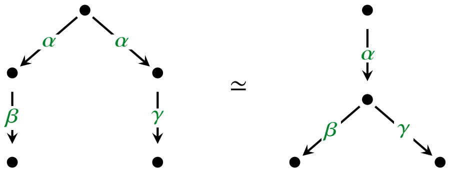
$\pi_1 = (\alpha;\beta) + (\alpha;\gamma)$ $\qquad$ $\pi_2 = \alpha;(\beta + \gamma)$



BUT

Bisimulation as program equivalence

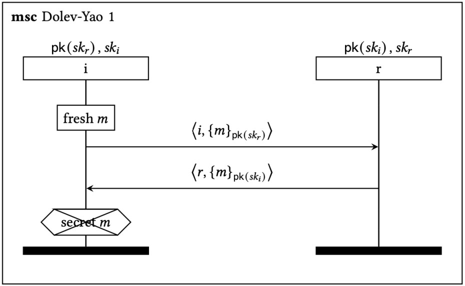
$\pi_1 = (\alpha;\beta) + (\alpha;\gamma)$ $\qquad$ $\pi_2 = \alpha;(\beta + \gamma)$



BUT

We do not want to reason about reachable states only,
we also want to express properties such as
"after the program α is executed, then agent p knows x"

Use the same methods used in formal verification of security protocols:



The secret is revealed if there is a state S such that

$$S \Vdash \langle \alpha_1 \rangle \cdots \langle \alpha_n \rangle \langle \text{secret } m \rangle \top$$

Logical framework: dynamic logic (Hennessy-Milner logic, modal μ-calulus) + epistemic Logic. . .

Propositional Dynamic Logic

Propositional Dynamic Logic (1976)

Formulas

$\phi, \psi :=$	$\top$		true
$\mid$	$\bot$		false
$\mid$	$p \in \mathcal{A}$	(with $p \in \mathcal{A}$)	atom
$\mid$	$\overline{p}$	(with $p \in \mathcal{A}$)	negated atom
$\mid$	$\phi \vee \psi$		disjunction
$\mid$	$\phi \wedge \psi$		conjunction
$\mid$	$[\alpha]\,\phi$		box
$\mid$	$\langle\alpha\rangle\,\phi$		diamond

Propositional Reasoning

Programs

$\alpha, \beta :=$	ϵ	terminated program
$\mid$	$\varnothing$	stacked program
$\mid$	$a \in \mathbb{I}$	instruction
$\mid$	$t \in \mathbb{T}$	test
$\mid$	$\alpha; \beta$	sequential composition
$\mid$	α^*	iteration
$\mid$	$\alpha \oplus \beta$	(non-deterministic) choice

Trace reasoning

Note: programs in PDL are elements of a regular language

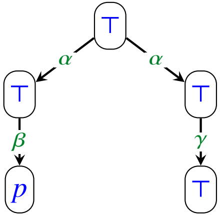
$$
\begin{aligned}
\mathfrak{m}(\top) &= W \\
\mathfrak{m}(\bot) &= \varnothing \\
\mathfrak{m}\left(\overline{\phi}\right) &= W \setminus \mathfrak{m}(\phi) \\
\mathfrak{m}(\phi \vee \psi) &= \mathfrak{m}(\phi) \cup \mathfrak{m}(\psi) \\
\mathfrak{m}(\phi \wedge \psi) &= \mathfrak{m}(\phi) \cap \mathfrak{m}(\psi) \\
\mathfrak{m}([\alpha]\,\phi) &= \{v \mid w \in \mathfrak{m}(\phi) \text{ for all } w \text{ s.t. } (v,w) \in \mathfrak{m}(\alpha)\} \\
\mathfrak{m}(\langle\alpha\rangle\,\phi) &= \{v \mid w \in \mathfrak{m}(\phi) \text{ for a } w \text{ s.t. } (v,w) \in \mathfrak{m}(\alpha)\}
\end{aligned}
$$

$$
\begin{aligned}
\mathfrak{m}(\epsilon) &= \{(v,v) \mid v \in W\} \\
\mathfrak{m}(\varnothing) &= \varnothing \\
\mathfrak{m}(\phi?) &= \{(v,v) \mid v \in \mathfrak{m}(\phi)\} \\
\mathfrak{m}(\alpha;\beta) &= \{(u,w) \mid \text{exists } v \text{ s.t. } (u,v) \in \mathfrak{m}(\alpha) \text{ and } (v,w) \in \mathfrak{m}(\beta)\} \\
\mathfrak{m}(\alpha \oplus \beta) &= \mathfrak{m}(\alpha) \cup \mathfrak{m}(\beta) \\
\mathfrak{m}(\alpha^*) &= \textstyle\bigcup_{n \geq 0} \mathfrak{m}(\alpha^n) \qquad \text{(where } \alpha^0 = \epsilon\text{)}
\end{aligned}
$$

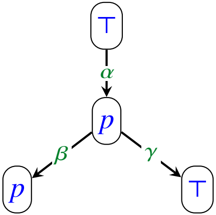
$$
\begin{aligned}
\mathfrak{m}(\top) &= W \\
\mathfrak{m}(\bot) &= \varnothing \\
\mathfrak{m}\left(\overline{\phi}\right) &= W \setminus \mathfrak{m}(\phi) \\
\mathfrak{m}(\phi \vee \psi) &= \mathfrak{m}(\phi) \cup \mathfrak{m}(\psi) \\
\mathfrak{m}(\phi \wedge \psi) &= \mathfrak{m}(\phi) \cap \mathfrak{m}(\psi) \\
\mathfrak{m}([\alpha]\,\phi) &= \{v \mid w \in \mathfrak{m}(\phi) \text{ for all } w \text{ s.t. } (v,w) \in \mathfrak{m}(\alpha)\} \\
\mathfrak{m}(\langle\alpha\rangle\,\phi) &= \{v \mid w \in \mathfrak{m}(\phi) \text{ for a } w \text{ s.t. } (v,w) \in \mathfrak{m}(\alpha)\}
\end{aligned}
$$

$$
\begin{aligned}
\mathfrak{m}(\epsilon) &= \{(v,v) \mid v \in W\} \\
\mathfrak{m}(\varnothing) &= \varnothing \\
\mathfrak{m}(\phi?) &= \{(v,v) \mid v \in \mathfrak{m}(\phi)\} \\
\mathfrak{m}(\alpha;\beta) &= \{(u,w) \mid \text{exists } v \text{ s.t. } (u,v) \in \mathfrak{m}(\alpha) \text{ and } (v,w) \in \mathfrak{m}(\beta)\} \\
\mathfrak{m}(\alpha \oplus \beta) &= \mathfrak{m}(\alpha) \cup \mathfrak{m}(\beta) \\
\mathfrak{m}(\alpha^*) &= \textstyle\bigcup_{n \geq 0} \mathfrak{m}(\alpha^n) \qquad \text{(where } \alpha^0 = \epsilon\text{)}
\end{aligned}
$$

$$\mathfrak{M}_1 \nVdash \langle(\alpha;\beta) + (\alpha;\gamma)\rangle\, p \vee [\beta + \gamma]\, p$$



$$\mathfrak{M}_2 \Vdash [(\alpha;\beta) + (\alpha;\gamma)]\, p \vee \langle\beta + \gamma\rangle\, p$$



Problem

No "pre-cooked" logics suitable for our purpose:

- No satisfactory Dynamic Logics handling both parallel/interleaving and recursion;
- The Hoare Logic[1] for choreographies (Cruz-Filipe, Graversen, Montesi & Peressotti, 2023) only reasons on formulas of the form

$$\phi \Rightarrow [\alpha]\,\psi$$

...But we want diamonds!

[1] Hoare 1969

WHY?

programs in PDL $=$ regular languages (elements of a Kleene Algebra)

WHY?

programs in PDL = regular languages (elements of a Kleene Algebra)
and

$$\text{Kleene Algebra} + \underbrace{\text{commutations}}_{\color{red}{\text{interleaving}}} \overset{\text{Kozen '96}}{\Longrightarrow} \text{undecidability whether } \color{green}{\alpha} = \color{green}{\beta}$$

WHY?

programs in PDL = regular languages (elements of a Kleene Algebra)
and

$$\text{Kleene Algebra} + \underbrace{\text{commutations}}_{\text{interleaving}} \overset{\text{Kozen '96}}{\Longrightarrow} \text{undecidability whether } \alpha = \beta$$

So in any "concurrent-PDL" $\vdash [\alpha]\top \Leftrightarrow [\beta]\top$ is undecidable.

WHY?

programs in PDL = regular languages (elements of a Kleene Algebra)
and

Kleene Algebra + $\underbrace{\text{commutations}}_{\text{interleaving}} \overset{\text{Kozen '96}}{\Longrightarrow}$ undecidability whether $\alpha = \beta$

So in any "concurrent-PDL" $\vdash [\alpha]\top \Leftrightarrow [\beta]\top$ is undecidable.

Solution: control the semantics of programs in the logic

Operational Propositional Dynamic Logic

O(perational Semantics)

$$\mathbf{A}_{\text{OS}} : [\alpha]\,\phi \Leftrightarrow \left(\bigwedge_{\alpha \xrightarrow{\beta} \gamma}^{\beta \text{ atomic}} [\beta]\,[\gamma]\,\phi \right)$$

PDL

- **PL** : Axiomatization of propositional classical logic
- $\mathbf{Neg} : [\alpha]\,\phi \Leftrightarrow \overline{\left(\langle\alpha\rangle\,\overline{\phi}\right)}$
- $\mathbf{K} : ([\alpha]\,(\phi \Rightarrow \psi)) \Rightarrow ([\alpha]\,\phi \Rightarrow [\alpha]\,\psi)$
- $\mathbf{A}_{\varnothing} : [\varnothing]\,\phi$
- $\mathbf{A}_{\epsilon} : [\epsilon]\,\phi \Leftrightarrow \phi$

- $\mathbf{A}_{?} : [\psi?]\,\phi \Leftrightarrow (\overline{\psi} \vee \phi)$
- $\mathbf{A}_{\oplus} : [\alpha \oplus \beta]\,\phi \Leftrightarrow ([\alpha]\,\phi \wedge [\beta]\,\phi)$
- $\mathbf{A}_{;} : [\alpha;\beta]\,\phi \Leftrightarrow [\alpha]\,[\beta]\,\phi$
- $\mathbf{A}_{*} : [\alpha^*]\,\phi \Leftrightarrow (\phi \wedge [\alpha]\,[\alpha^*]\phi)$

$$\text{MP}\ \frac{\vdash \phi \quad \vdash \phi \Rightarrow \psi}{\vdash \psi} \qquad \text{NEC}\ \frac{\vdash \phi}{\vdash [\alpha]\,\phi} \qquad \text{LI}\ \frac{\vdash \phi \Rightarrow [\alpha]\,\phi}{\vdash \phi \Rightarrow [\alpha^*]\,\phi}$$

Meaning of a non-atomic program:

$$\mathfrak{m}\,(\alpha) = \bigcup_{\alpha \xrightarrow{\beta} \gamma} \mathfrak{m}\,(\beta;\gamma)$$

Proof Theoretical Properties of OPDL

$$\top \frac{}{\vdash \top} \quad \text{AX} \frac{}{\vdash \phi, \overline{\phi}} \quad \text{W} \frac{\vdash \Gamma}{\vdash \Gamma, \phi} \quad \vee \frac{\vdash \Gamma, \phi, \psi}{\vdash \Gamma, \phi \vee \psi} \quad \wedge \frac{\vdash \Gamma, \phi \quad \vdash \Gamma, \psi}{\vdash \Gamma, \phi \wedge \psi} \quad \Big| \quad \text{K}_\alpha \frac{\vdash \Gamma, \phi}{\vdash \langle\alpha\rangle \Gamma, [\alpha]\,\phi} \; \alpha \notin \{\epsilon, \varnothing\} \quad \Big| \quad \text{cut} \frac{\vdash \Gamma, \phi \quad \vdash \Gamma, \overline{\phi}}{\vdash \Gamma}$$

$$[\epsilon] \frac{\vdash \Gamma, \phi}{\vdash \Gamma, [\epsilon]\,\phi} \quad [\varnothing] \frac{}{\vdash [\varnothing]\,\phi} \quad [?] \frac{\vdash \Gamma, \overline{\phi} \vee \psi}{\vdash \Gamma, [\phi?]\,\psi} \quad [\oplus] \frac{\vdash \Gamma, [\alpha]\,\phi \quad \vdash \Gamma, [\beta]\,\phi}{\vdash \Gamma, [\alpha \oplus \beta]\,\phi} \quad [;] \frac{\vdash \Gamma, [\alpha]\,[\beta]\,\phi}{\vdash \Gamma, [\alpha;\beta]\,\phi} \quad [*] \frac{\vdash \Gamma, \phi \quad \vdash \Gamma, [\alpha;\alpha^*]\,\phi}{\vdash \Gamma, [\alpha^*]\,\phi}$$

$$\langle\epsilon\rangle \frac{\vdash \Gamma, \phi}{\vdash \Gamma, \langle\epsilon\rangle\,\phi} \quad \langle\varnothing\rangle \frac{\vdash \Gamma, \psi}{\vdash \Gamma, \psi, \langle\varnothing\rangle\,\phi} \quad \langle?\rangle \frac{\vdash \Gamma, \phi \wedge \psi}{\vdash \Gamma, \langle\phi?\rangle\,\psi} \quad \langle\oplus\rangle \frac{\vdash \Gamma, \langle\alpha\rangle\,\phi, \langle\beta\rangle\,\phi}{\vdash \Gamma, \langle\alpha \oplus \beta\rangle\,\phi} \quad \langle;\rangle \frac{\vdash \Gamma, \langle\alpha\rangle\,\langle\beta\rangle\,\phi}{\vdash \Gamma, \langle\alpha;\beta\rangle\,\phi} \quad \langle*\rangle \frac{\vdash \Gamma, \phi, \langle\alpha;\alpha^*\rangle\,\phi}{\vdash \Gamma, \langle\alpha^*\rangle\,\phi}$$

$$[\text{OS}] \frac{\vdash \Gamma, [\beta_1]\,[\gamma_1]\,\phi \quad \cdots \quad \vdash \Gamma, [\beta_n]\,[\gamma_n]\,\phi}{\vdash \Gamma, [\alpha]\,\phi} \dagger \qquad \langle\text{OS}\rangle \frac{\vdash \Gamma, \langle\beta_1\rangle\,\langle\gamma_1\rangle\,\phi, \dots, \langle\beta_n\rangle\,\langle\gamma_n\rangle\,\phi}{\vdash \Gamma, \langle\alpha\rangle\,\phi} \dagger$$

$$\dagger \coloneqq \{(\beta_i, \gamma_i) \mid i \in \{1, \dots, n\}\} = \{(\beta, \gamma) \mid \alpha \xrightarrow{\beta} \gamma\}$$

Theorem

Let Γ be a sequent. Then $\vdash_{\mathsf{LOPD}} \Gamma$ iff $\vdash_{\mathsf{LOPD} \cup \{\mathsf{cut}\}} \Gamma$.

Theorem

Let Γ be a sequent. Then $\vdash_{\mathsf{LOPD}} \Gamma$ iff $\vdash_{\mathsf{OPDL}} \Gamma$.

Conclusion and Future Work

The standard PDL = OPDL with the following operational semantics:

$$a;\beta \xrightarrow{a} \beta \qquad \alpha \oplus \beta \xrightarrow{\epsilon} \alpha \qquad \alpha \oplus \beta \xrightarrow{\epsilon} \beta$$

$$\phi?;\beta \xrightarrow{\phi?} \beta \qquad \alpha^* \xrightarrow{\epsilon} \epsilon \qquad \alpha^* \xrightarrow{\epsilon} \alpha;\alpha^*$$

OPDL for CCS (previous attempt not satisfactory[2])

		processes			**labels**
$P, Q \coloneqq$	$\mathbf{0}$	terminated process	$\lambda \coloneqq$	a	actions ($a \in Act$)
$\mid$	$\lambda.P$	action prefix	$\mid$	$\overline{a}$	co-actions ($a \in Act$)
$\mid$	$P \mid Q$	parallel composition	$\mid$	τ	silent
$\mid$	$P + Q$	choice			
$\mid$	$P \backslash a$	action restriction			
$\mid$	X	process name			

PRE	$\lambda.P$	$\xrightarrow{\lambda}$	P	
PAR_1	$P \mid Q$	$\xrightarrow{\lambda}$	$P' \mid Q$	if $P \xrightarrow{\lambda} P'$
PAR_2	$P \mid Q$	$\xrightarrow{\lambda}$	$P \mid Q'$	if $Q \xrightarrow{\lambda} Q'$
COM	$P \mid Q$	$\xrightarrow{\tau}$	$P' \mid Q'$	if $P \xrightarrow{a} P'$ and $Q \xrightarrow{\overline{a}} Q'$
SUM_1	$P + Q$	$\xrightarrow{\lambda}$	P'	if $P \xrightarrow{\lambda} P'$
SUM_2	$P + Q$	$\xrightarrow{\lambda}$	Q'	if $Q \xrightarrow{\lambda} Q'$
RES	$P \backslash a$	$\xrightarrow{\lambda}$	$P' \backslash a$	if $P \xrightarrow{\lambda} P'$ and $\lambda \notin \{a, \overline{a}\}$
REC	X	$\xrightarrow{\lambda}$	P'	if $X \stackrel{\text{def}}{=} P$ and $P \xrightarrow{\lambda} P'$

[2] No nested parallel, and iteration instead of recursion

OPDL for choreographic programming

	choreographies	$\mathsf{pn}(C) =$
$C \coloneqq \mathbf{0}$	inactive process	$\varnothing$
$\mid I; C$	sequential composition	$\mathsf{pn}(I) \cup \mathsf{pn}(C)$
$\mid \mathsf{if}\ \mathsf{p}.b\ \mathsf{then}\ C_1\ \mathsf{else}\ C_2$	conditional	$\{\mathsf{p}\} \cup \mathsf{pn}(C_1) \cup \mathsf{pn}(C_2)$
$\mid X$	call	$\mathsf{pn}(C)$ where $X \stackrel{\mathsf{def}}{=} C$
	instructions	
$I \coloneqq \mathsf{p}.x \coloneqq e$	local assignment	$\{\mathsf{p}\}$
$\mid \mathsf{p}.e \to \mathsf{q}.x$	communication	$\{\mathsf{p}, \mathsf{q}\}$
$\mid \mathsf{p} \to \mathsf{q}[\mathrm{L}]$	selection	$\{\mathsf{p}, \mathsf{q}\}$
$\mid \mathsf{p} \colon X$	(call continuation, runtime)	$\{\mathsf{p}\}$
$\mid \mathsf{p}.b?$	test ($\mathbb{T}$)	$\{\mathsf{p}\}$
$\mid \overline{\mathsf{p}.b}?$	(negative) test	$\{\mathsf{p}\}$

Main results:

- Cut-elimination for PDL;
- More general framework OPDL parametric w.r.t. the desired OS
- . . . able to support concurrent programs!

Future work:

- Formalize;
- Add epistemic reasoning;
- Use results on differential privacy to define expert systems;
- ~~Bake cookies~~ Back to cookies!

Thanks

Questions?