Construction of invariance proof methods for parallel programs (with sequential consistency)

Patrick Cousot

NYU, NYC, NY
pcousot@cims.nyu.edu
History

Turing (1948) invents invariance + termination proofs for sequential programs.

Naur (1966) re-invents invariance proofs.

Floyd (1967) re-invents invariance + termination proofs.


... thousands of (forgotten) publications.

Owuchi [and Gries] (1976) generalize HL to parallel processes with sequential consistency (SC) (incomplete without auxiliary variables).

Lamport (1977) generalize Turing/Floyd/Naur for parallel processes with SC (complete thanks to program counters).

... thousands of (forgotten) publications.
Radhia Count (1980): all this is abstract interpretation.

--- thousands of (forgotten) publications

Today: researchers revive everything for weak memory models (WMM)

→ based on Owidi & Bier (incomplete!)

→ empirically, without any methodology.

Objective:

Explain a methodology for designing an invariance proof method by abstract interpretation of an operational semantics of the language.
DEFINITION OF INVARIANCE
BASED ON AN OPERATIONAL SEMANTICS
Operational semantics of a sequential process

- **States**: \( \langle c, m \rangle \in S \)
  - Memory state, \( m(x) \) is the value of (shared) variable \( x \)
  - Control point, specifies what remains to be executed in the program

- **Transitions**: \( \tau \in \mathcal{B}(S \times S) \)
  \[ \langle c, m \rangle \xrightarrow{\tau} \langle c', m' \rangle \]
  - i.e. \( \langle \langle qm \rangle, \langle c'/m' \rangle \rangle \in \tau \)
  - If execution of a computation step of the process at control point \( c \) in memory state \( m \) moves to control point \( c' \) in new memory state \( m' \)
Example

1: while $x < 10$ do
    2: $x := x + 1$
    3: od;

$\langle 1, m \rangle \xrightarrow{t} \langle e, m \rangle$ if $m(x) < 10$
$\langle 1, m \rangle \xrightarrow{t} \langle 3, m \rangle$ if $m(x) \geq 10$
$\langle e, m \rangle \xrightarrow{t} \langle 1, m' \rangle$

\[
\begin{align*}
&\text{if } m'(x) = m(x) + 1 \\
&m'(y) = m(y) \quad \text{for } y \neq x
\end{align*}
\]

\[\text{denoted } m' = m \left[ x \leftarrow m(x) + 1 \right] \]

Initial states: $I \subseteq S$

$I = \{ \langle 1, m \rangle \mid \forall x \in \mathbb{X}. m(x) \in \mathbb{Z} \}$
Transition system

\(<S, I, \tau>\)

- transition relation \(\tau \in \Phi(S \times S)\)
- initial state \(I \subseteq S\)
- states \(S\)

Also called "small-steps operational semantics"
Reflexive transitive closure

\[ t^0 = \{ <s, s'> \mid s = s' \} \]

\[ t_{n+1} = t \triangleleft t_n \]

\[ t_{n+1} = \{ <s, s'> \mid \exists s'' : <s, s'> \in t_n \land <s', s''> \in t_n \} \]

\[ t^* \subseteq \bigcup_{n \geq 0} t_n \]
Reachable states

- \( <S, I, t> \): transition system

- Reachable states \( R \):

  \[
  R = \{ s' \in S | \exists a \in I: t^*(s, a, s') \}
  \]
Invariance

- \( <S, I, t> \): transition system
- \( R \): reachable states of \( <S, I, t> \)
- Invariant:
  - Any subset of the reachable states
    \[ \phi \text{ is invariant for } <S, I, t> \]
    \[ \Rightarrow R <S, I, t> \subseteq \phi \]
Example

\[ \{ x \leq 10 \} \quad \text{Initial states (by hypothesis)} \]

1: \( \{ x \leq 18 \} \)
   while \( x < 10 \) do
   2: \( \{ x < 10 \} \)
      \( x := x + 1 \)
   od
3: \( \{ 10 \leq x \leq 11 \} \)

Reachable states:

\[ R = \{ \langle 1, m \rangle \mid m(x) \leq 10 \} \]
\[ \cup \{ \langle 2, m \rangle \mid m(x) < 10 \} \]
\[ \cup \{ \langle 3, m \rangle \mid m(x) = 10 \} \]

Invariant:

\[ Q = \{ \langle 1, m \rangle \mid m(x) \leq 11 \} \]
\[ \cup \{ \langle 2, m \rangle \mid m(x) < 10 \} \]
\[ \cup \{ \langle 3, m \rangle \mid 10 \leq m(x) \leq 11 \} \]
Relational Invariant

- $<s, I, t>$: transition system

- Relational invariant $Q$:
  
  - $Q \in \mathcal{F}(S \times S)$
  
  - $\{<s, s'> \mid s \in I \land t^*(s, s')\} \subseteq Q$
Fix Points
Example of fixpoint

\[ t^* \triangleq \bigcup_{n \geq 0} t^n \]

\[ t^* \text{ is a fixpoint of } F(x) = t^0 \cup x \ast t \]

Proof

\[ F(t^*) \]

\[ = t^0 \cup \left( \bigcup_{n \geq 0} t^n \right) \ast t \]

\[ = t^0 \cup \bigcup_{n \geq 0} (t^n \ast t) \]

\[ = t^0 \cup \bigcup_{n \geq 0} t^{n+1} \]

\[ = t^0 \cup \bigcup_{m \geq 1} t^m \]

\[ = \bigcup_{n \geq 0} t^n \]

\[ = t^* \]

\[ \square \]
\( t^* \) is the least fixpoint of \( F(x) = t^0 \cup x \cup t \)

Proof: Assume \( r = F(r) \) is a fixpoint of \( F \)
- \( t^0 \leq r \)
- \( t^n \leq r \) \quad \text{(ind. hyp.)}
- \( t^{n+1} \)
  \[ = t^n \cup t \]
  \[ \leq r \cup t \]
  \[ \leq r \cup r \]
  \[ = F(r) = r \]
- \( \forall n : t^n \leq r \) \quad \text{(by recurrence)}
  \[ \Rightarrow t^* = \bigcup_{n \geq 0} t^n \leq r \quad \text{(def. least upper bound U)} \]

\[ \Box \]

Notation: \( t^* = \text{fixp} \ F \)
Tarski's fixed point theorem (I)

If \( L (\bot, \top, \lor, \land, \oplus) \) is a complete lattice and \( F \in L \rightarrow L \) is \( \bot \)-increasing then
\[
\text{sup } F = \bigcap \{ x \in L : F(x) \leq x \}.
\]

Example: \( \mathbb{Q}(\times, \leq) (\bot, \phi, \times, \leq, \lor, \land) \)
\[
F(x) = t \lor u \lor x
\]
\[
t^* = \text{sup } F = \bigcap \{ x : t \lor u \lor x \leq r \}.
\]
Proof

\[ \rho \equiv \{ x \in L : f(x) \leq x \} \]
\[ \alpha \equiv \Pi \rho \]

- \( \forall x \in \rho : \]
  \[ a = \Pi \rho \in x \]
  \[ \Rightarrow f(a) \leq f(a) \]
  \[ \Rightarrow f(a) \leq x \]
  \[ \Rightarrow f(a) \text{ is a lower bound of } \rho \]
  \[ \Rightarrow f(a) \leq a \]
  \[ \Rightarrow f(f(a)) \leq f(a) \]
  \[ \Rightarrow f(a) \in \rho \]
  \[ \Rightarrow a \leq f(a) \]

- If \( x \) is any \( \Pi \times \rho \) point of \( F \) (which has at least one \( \alpha \))
  \[ f(x) = x \]
  \[ \Rightarrow f(x) \leq x \]
  \[ \Rightarrow x \in \rho \]
  \[ \Rightarrow a \leq x \]

- \( a = \text{gfp}(F) \) (def. \( \text{gfp} \))

(\( \text{gfp} \in \text{gfp}(F) \) since \( \text{TeP} \))

(greatest lower bound, \( \text{gfp} \))

(def. \( \text{gfp} \))

(F increasing)

(\( x \in \rho \) so \( f(x) \leq x \))

(a is the \( \text{gfp} \) of \( \rho \))

(F increasing)

(def. \( \rho \))

(a is the \( \text{gfp} \) of \( \rho \))

(anti-symmetry of \( \leq \))

(\( \text{def. \( \rho \)} \))

(\( \leq \) is reflexive)(\( \rho \))

(def. \( \rho \))

(a is the \( \text{gfp} \) of \( \rho \))

(\( \Pi \rho \text{ is } \exists x \in x \))

(\( \Pi \text{ exists } \exists x \))

(\( \Pi \text{ is } \exists x \))

(\( \Pi \text{ exists } \exists x \))

(\( \Pi \text{ exists } \exists x \))
Tarski's fixpoint theorem (Ⅱ)

\[ \text{If } L \left( E, \sqcup, \sqcap, \sqcup, \sqcap, \sqcap \right) \text{ is a complete lattice} \]

\[ \text{and } F \in L \to L \text{ preserves joins } \sqcup \text{ then} \]

\[ \text{\textit{eff}} F = \bigcup_{n \geq 0} F^n (\bot) \]

**Example:**

\[ \mathcal{F}(S \times S) \left( \leq, \emptyset, S \times S, \sqcup, \sqcap \right) \]

\[ F(x) = t^0 \sqcup x \sqcap t \]

\[ t^\ast = \text{\textit{eff}} F = \bigcup_{n \geq 0} t^n \]

- \[ F^0 (x) = x \]

\[ F^{n+1} (x) = F \left( F^n (x) \right) \]

- \[ F \left( \bigcup_{i \in \Delta} x_i \right) = \bigcup_{i \in \Delta} F(x_i) \] \hspace{1cm} \text{join preservation}

\[ F \left( \bigcup X \right) = \bigcup \left\{ F(x) : x \in X \right\} \]
Proof.

\[ a = \bigcup_{n \geq 0} F^n(\bot) \]

= \bigcup_{n \geq 0} F(F^n(\bot))

= \bigcup_{n \geq 0} F^{n+1}(\bot)

= \bot \bigcup_{n \geq 1} F^n(\bot)

= \bigcup_{n \geq 0} F^n(\bot)

= a

If \( x \) is any fixed point of \( F \):

1. \( F^0(\bot) = \bot \in x \)
2. \( F^n(\bot) \in x \)
3. \( F^{n+1}(\bot) = F(F^n(\bot)) \in F(\bot) = x \) (\( F \) preserves join hence increasing)
4. \( \forall n : F^n(\bot) \in x \)
5. \( a = \bigcup_{n \geq 0} F^n(\bot) \in x \) (def. \( a \) and \( \bot \) is the g eb)
- Th. wrongly attributed to Kleene
- $F$ is increasing so
  
  $\bot \subseteq F(\bot) \subseteq F^2(\bot) \subseteq \ldots \subseteq F^n(\bot) \subseteq \ldots$

- It is sufficient to assume that $F$ preserves the lub of increasing chain (Scott continuity).

- Generalizable to increasing functions by considering transfinite iterates.
Fixpoint Induction
Fixpoint over-approximation

Prove that \( \text{Eff} \subseteq \text{FEP} \)

(under the hypothesis of Tarshis' fixpoint theorem)
**Fixpoint Induction**

\[ \text{EFP } F \in P \iff \exists I : f(I) \in I \wedge I \in P \]

**Proof**

**Soundness** \( \iff \) :

\[ \begin{align*}
  f(I) & \in I \\
  \Rightarrow & I \in \{ x \mid f(x) \in x \} \\
  \Rightarrow & \text{EFP } F = \sqcap \{ x \mid f(x) \in x \} \in I \\
  \text{(Tarshis & def. glb } \sqcap) \\
  \Rightarrow & \text{EFP } F \in I \quad (\neg \in P \text{ and transitivity})
\end{align*} \]

**Completeness** \( \Rightarrow \) :

choose \( I = \text{EFP } (F) \) so \( f(I) = I \) implies \( f(I) \in I \) by reflexivity and \( I \in P \) by hypothesis.

**Relative completeness** :

In a logic (e.g., HL with first order logic), \( \text{EFP } (F) \) might not be expressible in that logic, a source of incompleteness.
Example

\( t^* \in \mathbb{R} \)

\( \iff \exists F \in \mathbb{R} \) where \( F(x) = t^0 u x g t \)

\( \iff \exists I : F(I) \subseteq I \land I \subseteq \mathbb{R} \)

\( \iff \exists I : t^0 \leq I \land I \subseteq \mathbb{R} \)

It is called the "inductive argument" (or invariant in the specific case of invariance proofs).
ABSTRACT INTERPRETATION
**ABSTRACTION**

**Propriétés :**
\[ x \in S \text{ a la propriété } p \]
\[ \iff \exists x \text{ appartenent à l'ensemble des éléments qui ont cette propriété} \]
\[ \iff \text{une propriété et un élément de } \mathcal{P}(S) \]

**Exemple :**
\[ \text{even}(x) \iff x \in \{ n \in \mathbb{N} \mid n \text{ est pair} \} \]

**Abstraction :**
A correspondance between properties.

\[ \alpha : \mathcal{P}(S) \rightarrow \mathcal{P}(A) \]

\[ \text{abstraction} \quad \text{propriétés concrets} \quad \text{propriétés abstraites} \]
Exemples

Reachable states

\[ R = \{ s' \in S | \exists x \in T : <s, s'> \in e^* \} \]

= \( \alpha (e^*) \)

where \( \alpha (X) = \{ s' \in S | \exists x \in T : <s, s'> \in e^* \} \)

\[ \alpha \in \mathcal{F}(S \times S) \rightarrow \mathcal{F}(S) \]

relation = properties of pairs of states

properties of states
Galois connection

\[ \{ \forall (s), \leq \} \overset{\tau}{\iff} \{ \forall (s'), \leq \} \]

\[ \forall s \in \forall (s) : \forall q \in \forall (s') : \]

\[ \alpha(p) \leq q \iff p \leq \delta(q) \]
Example

\[ \alpha(p) \subseteq \varnothing \]

\[ \iff \forall \sigma' \in S \mid \exists \alpha, I \colon <s, \sigma'> \in P \rightarrow \sigma' \subseteq \varnothing \]  
\[ \text{[def \alpha]} \]

\[ \forall \sigma' \colon (\exists \alpha, I \colon <s, \sigma'> \in P) \Rightarrow \sigma' \subseteq \varnothing \]

\[ \forall \sigma' \colon \forall \sigma \in I : <s, \sigma> \in P \Rightarrow \sigma' \subseteq \varnothing \]

\[ \forall \sigma, \forall \sigma' : <s, \sigma> \in P \Rightarrow (\exists \alpha, I \Rightarrow \sigma' \subseteq \varnothing) \]

\[ \forall \sigma, \forall \sigma' : <s, \sigma> \in P \Rightarrow \sigma' \subseteq \varnothing \]

\[ P \subseteq \{ <s, \sigma> | \exists \alpha, I \Rightarrow \sigma' \subseteq \varnothing \} \]

\[ \Rightarrow P \subseteq \alpha(\varnothing) \]

\[ \alpha(\varnothing) \]

\[ \text{[SFLA 2016]} \]
Intuition for Galois connections

- \( \chi(P) \) is an over-approximation of \( P \)
- \( \chi(Q) \) is the meaning of \( Q \).

- \( P \leq \chi(Q) \) i.e. \( P \) is over-approximated by \( \chi(Q) \) with meaning \( \chi(Q) \)

\[ \Rightarrow \chi(P) \leq Q \] i.e. \( \chi(P) \) is a more precise approximation of \( P \) than \( Q \)

- \( \chi(P) \leq Q \) i.e. \( Q \) is an over-approximation of the best approximation \( \chi(P) \) of \( P \)

\[ \Rightarrow P \leq \chi(Q) \] i.e. \( P \) is over-approximated by \( Q \) with meaning \( \chi(Q) \).
Properties of G.C.

- $\alpha$ is increasing
  - $\alpha(y) \subseteq \alpha(y)$
    \[ \Rightarrow y \subseteq \sigma \circ \alpha(y) \]
  - $x \in y$
    \[ \Rightarrow x \in \sigma \circ \alpha(y) \]
    \[ \Rightarrow \alpha(x) \subseteq \alpha(y) \]

(reflexivity)
(def. G.C.)
(hypothesis)
(def. G.C.)

(x $\in y \Rightarrow \sigma \circ \alpha(y)$ and transitivity)
Properties of G.C.

- Duality principle

\[ \langle L, \leq \rangle \iff \langle M, \leq \rangle \]
\[ \iff \alpha(x) \leq y \iff x \leq \tau(y) \]
\[ \iff \tau(y) \equiv x \iff y \geq \alpha(x) \]
\[ \iff \tau(x) \equiv y \iff x \geq \alpha(x) \]
\[ \iff \langle M, \gg \rangle \iff \langle L, \geq \rangle \]

I.e. if a theorem is true of \( L, \leq, M, \leq, \alpha, \tau \), then its dual for \( L, \geq, M, \geq, \tau, \alpha \), is also true.

E.g. \( \tau \) is unimodal.
Properties of G.C.

- $\alpha$ preserves Lubs.

  - Let $UX$ be the lub of $X$ in $L$.
  - Does $\alpha(X) = \{ \alpha(x) | x \in X \}$ have a lub $\bigvee \alpha(X) \cap M$?
  - Yes, this is $\alpha(UX) = \bigvee \alpha(X)$.

**Proof**

- $\forall x \in X : x \in UX$
- $\Rightarrow \forall x \in X : \alpha(x) \leq \alpha(UX)$ (\(\alpha\) increasing)
- $\Rightarrow \alpha(UX)$ is an upper bound of $\alpha(X)$
- Let $m$ be any upper bound of $\alpha(X)$
  - $\forall x \in X : \alpha(x) \leq m$ (def. upper bound)
  - $\Rightarrow \forall x \in X : \alpha(x) \leq \alpha(m)$ (def. G.C.)
  - $\Rightarrow UX \subseteq \alpha(m)$ (def. lub $U$)
  - $\Rightarrow \alpha(UX) \subseteq m$ (G.C.)
  - $\Rightarrow \alpha(UX)$ is the least upper bound of $\alpha(X)$ (by duality)

- $\delta$ preserves gLbs

SFLA 2016
Properties of G.C.

- one adjoint uniquely determines the other

Proof

\[ \alpha(x) = \cap \{ Y : \alpha(x) \leq Y \} \]

\[ = \cap \{ Y : Y \in \delta(y) \} \]

by duality

\[ \delta(x) = \cup \{ Y : \delta(x) \geq Y \} \]

\[ = \cup \{ Y : Y \leq \delta(x) \} \]
FIXPOINT ABSTRACTION
Fixpoint abstraction theorem

\[(L, \sqsubseteq, \bot, \top) \sqsubseteq \text{ complete lattices}\]

\[(M, \leq, V)\]

\[
\langle L, \sqsubseteq \rangle \leftrightarrow \langle M, \leq \rangle \quad \text{Galois connection}
\]

\[F \in L \rightarrow L \quad \{ \text{preserve lubs} \}\]

\[F \in M \rightarrow L \]

\[F \circ \alpha = \alpha \circ F \quad (\text{commutativity})\]

\[\Rightarrow \quad \langle \text{effp } F \rangle = \text{effp } F\]

Intuition: commutative abstracting preserve fixpoints

Numerous weaker versions.

JFLA 2016
Proof

- \( \alpha(F^0(\bot)) = \alpha(\bot) \) which is the supremum of \( M \)
  (since \( \bot \in \breve{x}(x) \) so \( \alpha(\bot) \leq x \), for all \( x \in M \))

- \( \alpha(F^n(\bot)) = \breve{F}^n(\alpha(\bot)) \) inductive hypothesis

\[ \Rightarrow \alpha(F^{n+1}(\bot)) = \breve{F}^{n+1}(\alpha(\bot)) \]

- \( \forall n: \alpha(F^n(\bot)) = \breve{F}^n(\alpha(\bot)) \)

\[ \Rightarrow \bigcup \alpha(F^n(\bot)) = \bigcup \breve{F}^n(\alpha(\bot)) \]

\[ \Rightarrow \alpha(\bigcup F^n(\bot)) = \bigcup \breve{F}^n(\alpha(\bot)) \]

\[ \Rightarrow \alpha(\text{epp } F) = \text{epp } \breve{F} \]

\[ \square \]
Example

\[ t^* = \varepsilon F \text{ where } F(x) = t \cup \delta x \sigma t \]
\[ \alpha(x) = \{ \lambda' \mid \exists e \in I : \langle s, \lambda' \rangle \in x \} \]

\[ \alpha(F(x)) \]
\[ = \alpha(t \cup \delta x \sigma t) \quad \text{(def. } F) \]
\[ = \alpha(t) \cup \alpha(x \sigma t) \quad \text{(} \alpha \text{ preserves } \cup \text{)} \]
\[ = \alpha(\{ \lambda' \mid \exists e \in I : s = \lambda' \}) \cup \alpha(x \sigma t) \]
\[ = I \cup \{ \lambda' \mid \exists e \in I : \exists s'' : \langle s', s'' \rangle \in x \land \langle s'', \lambda' \rangle \in t \} \]
\[ = I \cup \{ \lambda' \mid \exists e \in I : \exists s'' : \langle s', s'' \rangle \in x \land \langle s'', \lambda' \rangle \in t \} \]
\[ = I \cup \{ \lambda' \mid \exists e \in I : \alpha(x) = \langle s', \lambda' \rangle \in t \} \]
\[ = F(\alpha(x)) \quad \text{eureka!} \]

where \[ F(x) = I \cup \{ \lambda' \mid \exists s : \langle s', \lambda' \rangle \in t \} \]
and so \[ \alpha(t^*) = \alpha(\varepsilon F) = \varepsilon F \]

i.e. calculational design of the verification condition \[ F(x) \in \mathbb{X} \]
DESIGN OF AN INVARIANCE PROOF METHOD
Parallel Programs

\[ [P_1 \parallel \ldots \parallel P_n] \]

States:

\[ S = C_1 \times \ldots \times C_n \times M \]

\[ \uparrow \]

\[ \text{control parts of the processes} \]

\[ \uparrow \]

\[ \text{state of the variables in the shared memory} \]

Transition relation of processes

\[ t.S_i \in C_i \times M \]

\[ t_i \in \mathcal{G}(S_i \times S_i) \]

Transition relation of the parallel program

\[ t \in \mathcal{G}(S \times S) \]

\[ t_i = \{ \langle c_1, \ldots, c_{i-1}, c_i, c_{i+1}, \ldots, c_m \rangle, \langle c_1, \ldots, c_i', c_{i+1}, \ldots, c_m' \rangle \} \]

\[ t_i(\langle c_i, m \rangle, \langle c_i', m' \rangle) \]

\[ t = \bigvee_{i=1}^{n} t_i \]
**Principle of the design**

\[ R(s, i, t) \subseteq Q \quad \text{invariance} \]

\[ \xi(t^*) \subseteq Q \]

\[ \xi(t^r \diamond F) \subseteq Q \]

\[ \eta \diamond F \subseteq Q \]

\[ \exists P: F(P) \subseteq P \land P \subseteq Q \quad \text{Progress abstract} \]

\[ \exists P: S \cup \{ s' \in P : \langle s', a \rangle \in T \} \subseteq P \land P \subseteq Q \]

\[ \exists P: I \subseteq P \land \forall s' \in P: S \vdash s' \Rightarrow s \in P \land P \subseteq Q \]

Find an inductive invariant \( P \)

The inductive invariant is true for all initial states \( s \in I \)

Assuming the invariant true (\( s' \in P \)) prove that it remains true (\( s \in P \)) after a program step (\( s' \Rightarrow s \)) i.e. the invariant is inductive

The inductive invariant
Principle of the design

- This is the basic induction principle.
- Applying further fixpoint preserving abstractions we get
  - Numerous variants of the induction principle
    \[ \alpha(P) = -P \]
    \[ \alpha(t) = t^{-1} \]
  - Language specific invariance proof methods

Example: Twig / Naur / Floyd

\[ S = C \times M \]
\[ \text{control state} \]
\[ \text{memory state} \]

\[ \chi(P) = \prod_{c \in C} \{ m \mid <m> \in P \} \]

i.e. projection on the program control points to get local invariants on variables attached to program points.
APPLICATION TO PARALLEL PROCESSES (WITH SEQUENTIAL CONSISTENCY).
The Ascroft – Manna method

- Apply the further abstraction (which is also an isomorphism)

\[ \chi_{AM}(\rho) = \prod_{c_i \in C_1, c_2 \in C_2, ..., c_m \in C_m} \{ m \mid \langle c_1, ..., c_m \rangle \in \rho \} \]
Ascroft–Manna verification conditions

- obtained by the commutation condition of the fixpoint abstract theorem

- \( \forall c_1 \in C_1 : \forall c_2 \in C_2 : \ldots : \forall c_n \in C_n : \forall m \in M : \forall i \in [1,n] : \)
  \(<c_1 \ldots c_i-1 c_i c_{i+1} \ldots c_n \cdot m> \in P_{c_1 \ldots c_i \ldots c_n} \)
  \wedge <c_i, m> \xrightarrow{t} <c_i', m'>

  \Rightarrow <c_1 \ldots c_i-1 c_i' c_{i+1} \ldots c_n \cdot m'> \in P_{c_1 \ldots c'_i \ldots c_n}

- too many invariants \(|C_1| \times |C_2| \times \ldots \times |C_n|\)
The Lamport method.

- Apply the further isomorphic abstraction:

\[
\chi(P) = \prod_{i=1}^{n} \prod_{c \in C_i} \{ \langle c_1 \ldots c_{i-1} c_{i+1} \ldots c_n \rangle \mid \langle c_1 \ldots c_{i-1} c_i c_{i+1} \ldots c_n \rangle \in P \} 
\]

To each process \( P_i \), to each program pair of that process, attach an invariant:

- on the control points
- of the other processes
- on the shared memory state \( m \)
Lampert's verification conditions

- Obtained by the commutation conditions of the
  fixpoint abstraction for $\lambda$

- $\forall i \in [1, n] :$ 
  $\forall c_i \in C_i :$
    
    \[
    \langle c_1 \ldots c_{i-1} c_{i+1} \ldots c_n \ m \rangle \in P_{c_i} \\
    \wedge t_i (\langle c_i, m \rangle, \langle c'_i, m' \rangle) \\
    \Rightarrow \langle c_1 \ldots c_{i-1} c_{i+1} \ldots c_n m \rangle \in P_{c_i}
    \]

  $\wedge \forall j \in [1, n] \setminus \{i\} :$
    
    \[
    \langle c_1 \ldots c_{i-1} c_{i+1} \ldots c_j \ldots c_n m \rangle \in P_{c_i} \\
    \wedge t_j (\langle c_j, m \rangle, \langle c'_j, m' \rangle) \\
    \Rightarrow \langle c_1 \ldots c_{i-1} c_{i+1} \ldots c_j \ldots c_n m' \rangle \in P_{c_i}
    \]

- Note: The precondition can be strengthened e.g.

  \[
  \langle c_1 \ldots c_{i-1} c_{i+1} \ldots c_j c_{i-1} c_{i+1} \ldots c_n m \rangle \in P_{c_j}
  \]
Example

\[ \{ x = 0 \} \]

i.e. \( \{ m \mid m(x) = 0 \} \)

\[
\begin{align*}
1 & : c_2 = 3 \land x = 0 \\
   & \lor c_2 = 4 \land x = 1 \\
   & x := x + 1
\end{align*}
\]

\[
\begin{align*}
2 & : c_2 = 3 \land x = 1 \\
   & \lor c_2 = 4 \land x = 2
\end{align*}
\]

\[
\begin{align*}
3 & : c_4 = 1 \land x = 0 \\
   & \lor c_4 = 2 \land x = 1 \\
   & x := x + 1
\end{align*}
\]

\[
\begin{align*}
4 & : c_4 = 4 \land x = 1 \\
   & \lor c_4 = 2 \land x = 2
\end{align*}
\]

Initialisation: \( \{ x = 0 \} \land c_1 = 1 \land c_2 = 3 \implies \) \( P_3 \)

Sequential proof
Absence of interference proof

Finalisation: \( c_2 = 1 \land P_2 \land c_2 = 4 \land P_4 \implies x = 2 \).
The Owachi & Gries abstraction:

\[ \text{log } (P) = \prod_{i=1}^{n} \prod_{c_i \in C_i} \{ m \mid \exists \ c_1 \ldots \ c_{i-1} \ c_{i+1} \ldots \ c_n : \langle c_1 \ldots c_{i-1} \ c_i \ c_{i+1} \ldots c_n, m \rangle \in P \} \]

to each process \( P \).

to each program point \( c_i \) of process

i.e. same as Floyd for \( n = 1 \) but incomplete for \( n > 1 \).

attach an invariant on the shared memory state.
Proof of incompleteness

- To make the proof, we need an invariant.
- The strongest one is the LFP of the verification condition.
- Here is an example of strongest invariant:

\[
\begin{align*}
1: (x \geq 0) & \quad 3: (x \geq 0) \\
2: (x \geq 1) & \quad 4: (x \geq 1) \\
\{x=0\} \\
x := x + 1 & \quad x := x + 1
\end{align*}
\]

\[\Rightarrow \text{impossible to prove that } x=2 \text{ on exit.}\]
Auxiliary variables

- Add auxiliary variables to the program, prove the modified program, this implies the correctness of the original program

- Example

\[
\begin{align*}
1 : & \quad c_1 = 1 \\
& \quad x := x + 1 \\
2 : & \quad c_1 = 2 \\
& \quad x := x + 1 \\
3 : & \quad c_2 = 3 \\
& \quad x := x + 1 \\
4 : & \quad c_2 = 4
\end{align*}
\]

- Switch & criteria provide no clue on how to discover auxiliary variables
The completeness proof

- choose auxiliary variables that simulate the program counters
- show that the abstraction eliminating these auxiliary counters provides the semantics of the original method
- conclude by completeness of Lamport's method

see details in:

WHAT ABOUT JONES’ RELY/GUARANTEE?
Reachable states

\[ R = \mathcal{G}^+ F \]

\[ F(X) = \bigcup \{ s' \mid \exists s \in X : \overset{t}{\rightarrow} s' \} \]

\[ = \bigcup \{ s' \mid \exists s \in X : \bigvee_{i=1}^{m} \overset{E_i}{\rightarrow} s' \} \]

\[ = \bigcup_{i=1}^{m} \left( \bigcup \{ s' \mid \exists s \in X : \overset{E_i}{\rightarrow} s' \} \right) \]

\[ = R(G) X \]

where \[ G(X) = \bigcup_{j \neq i}^{\infty} \{ s' \mid \exists s \in X : \overset{E_j}{\rightarrow} s' \} \]

\[ R(G) X = \bigcup_{j \neq i}^{\infty} \{ s' \mid \exists s \in X : \overset{E_j}{\rightarrow} s' \} \]

\[ \leftarrow \text{guarantee} \]

\[ \overset{E_j}{\rightarrow} \]

\[ \leftarrow \text{rely (assuming guarantee)} \]
Reachable states

**Theorem**

\[ \text{eff } F = \text{eff } \int X. R(G(X)) \]

**proof**

The least fixed point of

\[ X = F(X) \]

is the same as the least fixed point of the system of equalities

\[ X = R(Y) \]

\[ Y = G(X) \]

by the theorem of asynchronous iterations with memory (Cousot & Cousot, 1977)

\[ \square \]
- Apply the proposer-vouched principle to
  \[ \text{eff} \{ (x, y) \cdot \langle R(y) \times G(x) \rangle \} \]
- up to the Lamport abstraction \( \alpha_L \), assigning
  to each control point an assertion on
  - the shared variable
  - the control point of the other process
  (or Owuchi & Gnes with auxiliary variables?)

---

- Cliff B. Jones:

- Joey W. Coleman, Cliff B. Jones:
APPLICATIONS
Astrée A

- Astrée : a static analyser of C for synchronous control-command embedded software
- Astrée A : idem, for parallel programs

⇒ a further abstraction of
- an invariant at each point of each process on the shared variables and program counter of other processes
+ rely-guaranteed fix-point computation
+ widening/narrowing convergence acceleration
CONCLUSION
Conclusion

- Too many computer scientists are tinker[wo]men (bricoleur[rs/sea])
- If you want to understand what you do go, to basic principles.
- For reasoning on program semantics this is A.I. =)

PS: this approach generalizes to termination (Cousot & Cousot, APL 20K)
THE END