

# 1 Introduction

Reasoning about programs bla bla bla

## 2 Programming Language

### 2.1 Syntax

We start by defining an imperative programming language with non deterministic choice and parametric on a set  $Base$  of base commands, common choices for the set of base commands usually contain a command for assignments and boolean guards.

The set of valid  $\mathbb{C}$  programs is defined by the following inductive definition:

**Definition 1** ( $\mathbb{C}$  language syntax)

$\mathbb{C} ::= \mathbb{0}$	<i>Always non terminating program</i>
$\mathbb{1}$	<i>Identity program (skip)</i>
$b$	<i>Base command</i>
$C_1 \mathbin{\circ} C_2$	<i>Program composition</i>
$C_1 + C_2$	<i>Non deterministic choice</i>
$C^*$	<i>Iteration</i>

Where  $C, C_1, C_2 \in \mathbb{C}$  and  $b \in Base$ .

### 2.2 Semantics

Given a set  $\mathbb{S}$  of states and a family of partial functions  $\llbracket b \rrbracket_{base} : \mathbb{S} \hookrightarrow \mathbb{S}$  we can define inductively the semantics of a program in  $\mathbb{C}$  by structural induction on the terms:

**Definition 2** ( $\mathbb{C}$  language semantics)

$$\begin{aligned}
\llbracket \cdot \rrbracket &: \mathcal{P}(\mathbb{S}) \rightarrow \mathcal{P}(\mathbb{S}) \\
\llbracket \mathbb{0} \rrbracket &= \text{const } \emptyset \\
\llbracket \mathbb{1} \rrbracket &= id \\
\llbracket b \rrbracket &= \lambda P \rightarrow \{x \mid \llbracket b \rrbracket_{base}(p) \downarrow = x \wedge p \in P\} \\
\llbracket C_1 \mathbin{\circ} C_2 \rrbracket &= \llbracket C_2 \rrbracket \circ \llbracket C_1 \rrbracket \\
\llbracket C_1 + C_2 \rrbracket &= \lambda P \rightarrow \llbracket C_1 \rrbracket P \cup \llbracket C_2 \rrbracket P \\
\llbracket C^* \rrbracket &= \lambda P \rightarrow lfp(\lambda P' \rightarrow P \cup \llbracket C \rrbracket P')
\end{aligned}$$

**Theorem 1** ( $\llbracket \cdot \rrbracket$  is monotone)

$$P \subseteq Q \implies \llbracket C \rrbracket(P) \subseteq \llbracket C \rrbracket(Q)$$

This framework is general enough to cover non deterministic imperative languages, for example if we include a base command for boolean guards  $e?$  where  $e$  is a boolean valued expression on  $\mathbb{S}$  we can define the usual control flow statements **if**  $b$  **then**  $C_1$  **else**  $C_2$  as  $(e? \circ C_1) + (\neg e? \circ C_2)$  and **while**  $e$  **do**  $C$  **done** as  $(e? \circ C)^* \circ \neg e?$ .

### 3 Best Abstract Inductive semantics

We can abstract the *Best Abstract Inductive Semantics* of a program in  $\mathbb{C}$  parametrically on a complete lattice  $A$  that has as elements collections of program states and a family of monotone functions  $\llbracket b \rrbracket_{base} : A \rightarrow A \quad \forall b \in Base$  by structural induction on the syntax of  $\mathbb{C}$ :

**Definition 3 (Best abstract inductive semantics)**

$$\begin{aligned} \llbracket \cdot \rrbracket_{bai}^A &: A \rightarrow A \\ \llbracket 0 \rrbracket_{bai}^A &= const \perp_A \\ \llbracket 1 \rrbracket_{bai}^A &= id_A \\ \llbracket b \rrbracket_{bai}^A &= \lambda P \rightarrow \llbracket b \rrbracket_{base}(P) \\ \llbracket C_1 \circ C_2 \rrbracket_{bai}^A &= \llbracket C_2 \rrbracket_{bai}^A \circ \llbracket C_1 \rrbracket_{bai}^A \\ \llbracket C_1 + C_2 \rrbracket_{bai}^A &= \lambda P \rightarrow \llbracket C_1 \rrbracket_{bai}^A(P) \vee_A \llbracket C_2 \rrbracket_{bai}^A(P) \\ \llbracket C^* \rrbracket_{bai}^A &= \lambda P \rightarrow lfp(\lambda P' \rightarrow P \vee_A \llbracket C \rrbracket_{bai}^A P') \end{aligned}$$

**Theorem 2 (Semantic equivalence)** *If we take as the lattice  $\mathcal{P}(\mathbb{S})$  and as  $\llbracket b \rrbracket_{bai}^A = \lambda P \rightarrow \{x \mid \llbracket b \rrbracket(p) \downarrow = x \wedge p \in P\}$  the two semantics are identical.*

$$\llbracket C \rrbracket_{bai}^{\mathcal{P}(\mathbb{S})}(P) = \llbracket C \rrbracket(P)$$

**Theorem 3 ( $\llbracket \cdot \rrbracket_{bai}^A$  is monotone)**

$$P \leq_A Q \implies \llbracket C \rrbracket_{bai}^A(P) \leq_A \llbracket C \rrbracket_{bai}^A(Q)$$

#### 3.1 Obtaining BAIs from other BAIs via Galois connections

Given a Galois connection  $\langle D, \leq_D \rangle \xrightleftharpoons[\alpha]{\gamma} \langle A, \leq_A \rangle$  if we have define a *Best Abstract Inductive Semantics* on  $D$  with the semantics of basic commands  $\llbracket b \rrbracket_{bai}^D$  we can define a *Best Abstract Inductive Semantics* on  $A$  with semantics of basic commands  $\llbracket b \rrbracket_{bai}^A = \alpha \circ \llbracket b \rrbracket_{bai}^D \circ \gamma$ .

**Theorem 4 (Soundness)**

$$\alpha(\llbracket C \rrbracket_{bai}^D(P) \leq_D \llbracket C \rrbracket_{bai}^A(\alpha(P)))$$

(Here soundness is intended in an abstract interpretation sense)

## 4 Abstract Hoare logic

In this section we will define when an Abstract Hoare triple is valid, give some inference rules for it **BLAH**.

**Definition 4 (Abstract Hoare triple)** *Fixed a complete lattice  $A$  and the semantics of the base commands  $\llbracket b \rrbracket_{bai}^A$ , an Abstract Hoare triple is valid if and only if executing the bai of a command  $C$  of some precondition captured by the element  $P$  of  $A$  is overapproximated by some element  $Q$  of  $A$ :*

$$\langle P \rangle_A C \langle Q \rangle \iff \llbracket C \rrbracket_{bai}^A(P) \leq_A Q$$

### 4.1 Inference rule

As in standard Hoare logic we can give a set of inference rules to derive valid triples from smaller ones:

**Definition 5 (Abstract Hoare inference rules)**  $\frac{}{\vdash \langle P \rangle_A \mathbf{0} \langle \perp \rangle} (\mathbf{0})$

$$\frac{}{\vdash \langle P \rangle_A \mathbf{1} \langle P \rangle} (1)$$

$$\frac{}{\vdash \langle P \rangle_A b \langle \llbracket b \rrbracket_{base}(P) \rangle} (b)$$

$$\frac{\vdash \langle P \rangle_A C_1 \langle Q \rangle \quad \vdash \langle Q \rangle_A C_2 \langle R \rangle}{\vdash \langle P \rangle_A C_1 ; C_2 \langle R \rangle} (s)$$

$$\frac{\vdash \langle P \rangle_A C_1 \langle Q \rangle \quad \vdash \langle P \rangle_A C_2 \langle Q \rangle}{\vdash \langle P \rangle_A C_1 + C_2 \langle Q \rangle} (+)$$

$$\frac{\vdash \langle P \rangle_A C \langle P \rangle}{\vdash \langle P \rangle_A C^* \langle P \rangle} (*)$$

$$\frac{P \leq P' \quad \vdash \langle P' \rangle_A C \langle Q' \rangle \quad Q' \leq Q}{\vdash \langle P \rangle_A C \langle Q \rangle} (\leq)$$

**Example 4.1 (Derivation in the integer interval domain)** *We will start by defining our running example the integer interval domain: The elements of the interval domain on one variable are defined by  $Int = \{[x, y] \mid x \leq y \quad x, y \in \mathbb{Z} \cup \{+\infty, -\infty\}\} \cup \{\perp\}$  where the order relation is given by  $\perp \leq_{Int} a \quad \forall a \in Int$  and  $[a, b] \leq_{Int} [c, d] \iff c \leq a \wedge b \leq d$  and clearly  $\top = [-\infty, +\infty]$ . The definition can be lifted pointwise for domains with an arbitrary number of variables.*

*As basic command we will add  $e?$  for boolean tests discarding all the states that don't satisfy the condition  $e$  and  $x := y$  assigning the result of evaluating expression  $y$  to the variable  $x$  and fix  $\llbracket b \rrbracket_{bai}^A = \alpha \circ \llbracket b \rrbracket \circ \gamma$ .*

Then the following is a valid derivation for program  $C = (x := 1 + x := 2) \circ x := x + 1$

$$\frac{\frac{\frac{\top \leq \top \quad \vdash \langle \top \rangle_A x := 1 \ (x \in [1, 1]) \quad x \in [1, 1] \leq x \in [1, 2]}{\vdash \langle \top \rangle_A x := 1 \ (x \in [1, 2])} \quad \frac{\top \leq \top \quad \vdash \langle \top \rangle_A x := 2 \ (x \in [2, 2]) \quad x \in [2, 2] \leq x \in [1, 2]}{\vdash \langle \top \rangle_A x := 2 \ (x \in [1, 2])}}{\vdash \langle \top \rangle_A x := 1 + x := 2 \ (x \in [1, 2])} \quad \frac{\vdash \langle \top \rangle_A x := 1 + x := 2 \ (x \in [1, 2]) \quad \vdash \langle x \in [1, 2] \rangle_A x := x + 1 \ (x \in [2, 3])}{\vdash \langle \top \rangle_A C \ (x \in [2, 3])}$$

And clearly  $\llbracket C \rrbracket_{bai}^A(\top) = x \in [2, 3]$ .

**Theorem 5 (The proofsystem is sound)**

$$\vdash \langle P \rangle_A C \langle Q \rangle \implies \langle P \rangle_A C \langle Q \rangle$$

**Proof 1** By structural induction on the last rule applied in the derivation of  $\vdash \langle P \rangle_A C \langle Q \rangle$ :

- (0): Then the last step in the derivation was:

$$\frac{}{\vdash \langle P \rangle_A \mathbf{0} \langle \perp \rangle} \text{ (0)}$$

The triple is valid since:

$$\llbracket \mathbf{0} \rrbracket_{bai}^A(P) = \perp_A \quad \text{By definition of } \llbracket \cdot \rrbracket_{bai}^A$$

- (1): Then the last step in the derivation was:

$$\frac{}{\vdash \langle P \rangle_A \mathbf{1} \langle P \rangle} \text{ (1)}$$

The triple is valid since:

$$\llbracket \mathbf{1} \rrbracket_{bai}^A(P) = P \quad \text{By definition of } \llbracket \cdot \rrbracket_{bai}^A$$

- (b): Then the last step in the derivation was:

$$\frac{}{\vdash \langle P \rangle_A b \langle \llbracket b \rrbracket_{base}(P) \rangle} \text{ (b)}$$

The triple is valid since:

$$\llbracket b \rrbracket_{bai}^A(P) = \llbracket b \rrbracket_{base}(P) \quad \text{By definition of } \llbracket \cdot \rrbracket_{bai}^A$$

- ( $\circ$ ): Then the last step in the derivation was:

$$\frac{\vdash \langle P \rangle_A C_1 \langle Q \rangle \quad \vdash \langle Q \rangle_A C_2 \langle R \rangle}{\vdash \langle P \rangle_A C_1 \circ C_2 \langle R \rangle} \text{ ( $\circ$ )}$$

By inductive hypothesis:  $\llbracket C_1 \rrbracket_{bai}^A(P) \leq_A Q$  and  $\llbracket C_2 \rrbracket_{bai}^A(Q) \leq_A R$ .

The triple is valid since:

$$\begin{aligned} \llbracket C_1 \circ C_2 \rrbracket_{bai}^A(P) &= \llbracket C_2 \rrbracket_{bai}^A(\llbracket C_1 \rrbracket_{bai}^A(P)) && \text{By definition of } \llbracket \cdot \rrbracket_{bai}^A \\ &\leq_A \llbracket C_2 \rrbracket_{bai}^A(Q) && \text{By monotonicity of } \llbracket \cdot \rrbracket_{bai}^A \\ &\leq_A R \end{aligned}$$

- (+): Then the last step in the derivation was:

$$\frac{\vdash \langle P \rangle_A C_1 \langle Q \rangle \quad \vdash \langle P \rangle_A C_2 \langle Q \rangle}{\vdash \langle P \rangle_A C_1 + C_2 \langle Q \rangle} (+)$$

By inductive hypothesis:  $\llbracket C_1 \rrbracket_{bai}^A(P) \leq Q$  and  $\llbracket C_2 \rrbracket_{bai}^A(P) \leq Q$ .

The triple is valid since:

$$\begin{aligned} \llbracket C_1 + C_2 \rrbracket_{bai}^A(P) &= \llbracket C_1 \rrbracket_{bai}^A(P) \vee \llbracket C_2 \rrbracket_{bai}^A(P) && \text{By definition of } \llbracket \cdot \rrbracket_{bai}^A \\ &\leq_A Q \vee Q \\ &= Q \end{aligned}$$

- (★): Then the last step in the derivation was:

$$\frac{\vdash \langle P \rangle_A C \langle P \rangle}{\vdash \langle P \rangle_A C^\star \langle P \rangle} (*)$$

By inductive hypothesis:  $\llbracket C \rrbracket_{bai}^A P \leq P$

$$\llbracket C^\star \rrbracket_{bai}^A(P) = \text{lf}p(\lambda P' \rightarrow P \vee_A \llbracket C \rrbracket_{bai}^A(P'))$$

$$\begin{aligned} (\lambda P' \rightarrow P \vee_A \llbracket C \rrbracket_{bai}^A(P'))(P) &= P \vee \llbracket C \rrbracket_{bai}^A(P) && \text{since } \llbracket C \rrbracket_{bai}^A(P) \leq P \\ &= P \end{aligned}$$

Hence  $P$  is a fixpoint for  $\lambda P' \rightarrow P \vee_A \llbracket C \rrbracket_{bai}^A(P')$

Thus  $\text{lf}p(\lambda P' \rightarrow P \vee_A \llbracket C \rrbracket_{bai}^A(P')) \leq_A P$

- ( $\leq$ ): Then the last step in the derivation was:

$$\frac{P \leq P' \quad \vdash \langle P' \rangle_A C \langle Q' \rangle \quad Q' \leq Q}{\vdash \langle P \rangle_A C \langle Q \rangle} (\leq)$$

By inductive hypothesis:  $\llbracket C \rrbracket_{bai}^A(P') \leq Q'$ .

$$\begin{aligned} \llbracket C \rrbracket_{bai}^A(P) \llbracket C \rrbracket_{bai}^A(P') &&& \text{By monotonicity of } \llbracket \cdot \rrbracket_{bai}^A \\ &\leq Q' && \text{By inductive hypothesis} \\ &\leq Q \end{aligned}$$

**Theorem 6 (Relative  $\llbracket \cdot \rrbracket_{bai}^A$ -completeness)**

$$\vdash \langle P \rangle_A C \langle \llbracket C \rrbracket_{bai}^A(P) \rangle$$

**Proof 2** *By structural induction on  $C$ :*

- $0$ : By definition  $\llbracket 0 \rrbracket_{bai}^A(P) = \perp$

$$\frac{}{\vdash \langle P \rangle_A 0 \langle \perp \rangle} (0)$$

- $1$ : By definition  $\llbracket 1 \rrbracket_{bai}^A(P) = P$

$$\frac{}{\vdash \langle P \rangle_A 1 \langle P \rangle} (1)$$

- $b$ : By definition  $\llbracket b \rrbracket_{bai}^A(P) = \llbracket b \rrbracket_{base}(P)$

$$\frac{}{\vdash \langle P \rangle_A b \langle \llbracket b \rrbracket_{base}(P) \rangle} (b)$$

- $C_1 \circ C_2$ : By definition  $\llbracket C_1 \circ C_2 \rrbracket_{bai}^A(P) = \llbracket C_2 \rrbracket_{bai}^A(\llbracket C_1 \rrbracket_{bai}^A(P))$

$$\frac{\begin{array}{c} (Inductive\ hypothesis) \\ \vdash \langle P \rangle_A C_1 \langle \llbracket C_1 \rrbracket_{bai}^A(P) \rangle \end{array} \quad \begin{array}{c} (Inductive\ hypothesis) \\ \vdash \langle \llbracket C_1 \rrbracket_{bai}^A(P) \rangle_A C_2 \langle \llbracket C_2 \rrbracket_{bai}^A(\llbracket C_1 \rrbracket_{bai}^A(P)) \rangle \end{array}}{\vdash \langle P \rangle_A C_1 \circ C_2 \langle \llbracket C_2 \rrbracket_{bai}^A(\llbracket C_1 \rrbracket_{bai}^A(P)) \rangle} (9)$$

- $C_1 + C_2$ : By definition  $\llbracket C_1 + C_2 \rrbracket_{bai}^A(P) = \llbracket C_1 \rrbracket_{bai}^A(P) \vee \llbracket C_2 \rrbracket_{bai}^A(P)$

$$\frac{\begin{array}{c} (Inductive\ hypothesis) \\ P \leq P \end{array} \quad \frac{\vdash \langle P \rangle_A C_1 \langle \llbracket C_1 \rrbracket_{bai}^A(P) \rangle \quad \llbracket C_1 \rrbracket_{bai}^A(P) \leq \llbracket C_1 \rrbracket_{bai}^A(P) \vee \llbracket C_2 \rrbracket_{bai}^A(P)}{\vdash \langle P \rangle_A C_1 \langle \llbracket C_1 \rrbracket_{bai}^A(P) \vee \llbracket C_2 \rrbracket_{bai}^A(P) \rangle} (\leq)}{\vdash \langle P \rangle_A C_1 + C_2 \langle \llbracket C_1 \rrbracket_{bai}^A(P) \vee \llbracket C_2 \rrbracket_{bai}^A(P) \rangle} \pi_1 (+)$$

Where  $\pi_1$ :

$$\frac{\begin{array}{c} (Inductive\ hypothesis) \\ P \leq P \end{array} \quad \frac{\vdash \langle P \rangle_A C_2 \langle \llbracket C_2 \rrbracket_{bai}^A(P) \rangle \quad \llbracket C_2 \rrbracket_{bai}^A(P) \leq \llbracket C_1 \rrbracket_{bai}^A(P) \vee \llbracket C_2 \rrbracket_{bai}^A(P)}{\vdash \langle P \rangle_A C_2 \langle \llbracket C_1 \rrbracket_{bai}^A(P) \vee \llbracket C_2 \rrbracket_{bai}^A(P) \rangle} (\leq)}{\vdash \langle P \rangle_A C_1 + C_2 \langle \llbracket C_1 \rrbracket_{bai}^A(P) \vee \llbracket C_2 \rrbracket_{bai}^A(P) \rangle} \pi_1 (+)$$

- $C^*$ : By definition  $\llbracket C^* \rrbracket_{bai}^A(P) = lfp(\lambda P'. P \vee \llbracket C \rrbracket_{bai}^A(S'))$  and let's call this value  $K$ , by  $K$  being a fixpoint the following fact is true  $K = P \vee \llbracket C \rrbracket_{bai}^A(K)$  hence the following facts are true:

- $\alpha_1: K \geq P$
- $\alpha_2: K \geq \llbracket C \rrbracket_{bai}^A(K)$

$$\begin{array}{c}
\text{(Inductive hypothesis)} \\
\frac{K \leq K \quad \vdash \langle K \rangle_A C \langle \llbracket C \rrbracket_{bai}^A(K) \rangle \quad \alpha_2}{\vdash \langle K \rangle_A C \langle K \rangle} \quad (\star) \\
\frac{\alpha_1 \quad \vdash \langle K \rangle_A C \langle K \rangle \quad K \leq K}{\vdash \langle P \rangle_A C \langle K \rangle} \quad (\leq)
\end{array}$$

**Theorem 7 (Relative completeness)**

$$\langle P \rangle_A C \langle Q \rangle \implies \vdash \langle P \rangle_A C \langle Q \rangle$$

**Proof 3** By definition of  $\langle P \rangle_A C \langle Q \rangle \iff Q \geq \llbracket C \rrbracket_{bai}^A(P)$

$$\begin{array}{c}
\text{By Theorem 6} \\
\frac{P \leq P \quad \vdash \langle P \rangle_A C \langle \llbracket C \rrbracket_{bai}^A(P) \rangle \quad Q \geq \llbracket C \rrbracket_{bai}^A(P)}{\vdash \langle P \rangle_A C \langle Q \rangle} \quad (\leq)
\end{array}$$

## 4.2 Merge rules

In standard Hoare logic the following rule is valid:

$$\frac{\vdash \{P_1\} C \{Q\} \quad \vdash \{P_2\} C \{Q\}}{\vdash \{P_1 \vee P_2\} C \{Q\}} \quad (merge)$$

And can be used to easily merge the results obtained by different analysis, but in general the same rule in the context of abstract Hoare logic:

$$\frac{\vdash \langle P_1 \rangle_A C \langle Q \rangle \quad \vdash \langle P_2 \rangle_A C \langle Q \rangle}{\vdash \langle P_1 \vee P_2 \rangle_A C \langle Q \rangle} \quad (merge)$$

is unsound.

**Example 4.2 (Unsoundness of the rule (merge))** Let  $A$  and  $\llbracket b \rrbracket_{bai}^A$  be the same as in 4.1 and  $C = (x = 3? \text{ } x := 400) + (x \neq 3? \text{ } x := x + 1)$

Then we can derive the following triples:

- $\vdash \langle x \in [1, 2] \rangle_A C \langle x \in [2, 3] \rangle$
- $\vdash \langle x \in [4, 5] \rangle_A C \langle x \in [5, 6] \rangle$

But applying the rule (merge) would allow to derive the following:  $\vdash \langle x \in [1, 5] \rangle_A C \langle x \in [2, 6] \rangle$ , that is unsound since  $\llbracket C \rrbracket_{bai}^A(x \in [1, 5]) = x \in [2, 400]$ .

The cause of the issue is caused by the non additivity of the base commands, in fact:

**Theorem 8 (Additivity of the semantics on attitive base commands)**  
 if  $\forall P_1, P_2 \in A$  and  $b \in \text{Base}$   $\llbracket B \rrbracket_{bai}^A(P_1 \vee P_2) = \llbracket B \rrbracket_{bai}^A(P_1) \vee \llbracket B \rrbracket_{bai}^A(P_2)$  then:  
 $\forall P_1, P_2 \in A \quad C \in \mathbb{C}$

$$\llbracket C \rrbracket_{bai}^A(P_1) \vee \llbracket C \rrbracket_{bai}^A(P_2) = \llbracket C \rrbracket_{bai}^A(P_1 \vee P_2)$$

**Proof 4** By structural induction on  $C$ :

- $\mathbb{0}$ : By definition  $\llbracket \mathbb{0} \rrbracket_{bai}^A(P_1 \vee P_2) = \perp$  and  $\llbracket \mathbb{0} \rrbracket_{bai}^A(P_i) = \perp$
- $\mathbb{1}$ : By definition  $i \in \{1, 2\}$   $\llbracket \mathbb{1} \rrbracket_{bai}^A(P_i) = P_i$  and  $\llbracket \mathbb{1} \rrbracket_{bai}^A(P_1 \vee P_2) = P_1 \vee P_2$
- $b$ :

$$\llbracket b \rrbracket_{bai}^A(P_1 \vee P_2) = \llbracket b \rrbracket_{bai}^A(P_1) \vee \llbracket b \rrbracket_{bai}^A(P_2) \quad \text{By additivity of } \llbracket b \rrbracket_{bai}^A$$

- $C_1 \circ C_2$ :

$$\begin{aligned} \llbracket C_1 \circ C_2 \rrbracket_{bai}^A(P_1 \vee P_2) &= \llbracket C_2 \rrbracket_{bai}^A(\llbracket C_1 \rrbracket_{bai}^A(P_1 \vee P_2)) \\ &= \llbracket C_2 \rrbracket_{bai}^A(\llbracket C_1 \rrbracket_{bai}^A(P_1) \vee \llbracket C_1 \rrbracket_{bai}^A(P_2)) \quad \text{By inductive hypothesis} \\ &= \llbracket C_1 \circ C_2 \rrbracket_{bai}^A(P_1) \vee \llbracket C_1 \circ C_2 \rrbracket_{bai}^A(P_2) \quad \text{By inductive hypothesis} \end{aligned}$$

- $C_1 + C_2$ :

$$\begin{aligned} \llbracket C_1 + C_2 \rrbracket_{bai}^A(P_1 \vee P_2) &= \llbracket C_1 \rrbracket_{bai}^A(P_1 \vee P_2) \vee \llbracket C_2 \rrbracket_{bai}^A(P_1 \vee P_2) \\ &= \llbracket C_1 \rrbracket_{bai}^A(P_1) \vee \llbracket C_2 \rrbracket_{bai}^A(P_2) \\ &\quad \vee \llbracket C_2 \rrbracket_{bai}^A(P_1) \vee \llbracket C_2 \rrbracket_{bai}^A(P_2) \quad \text{By inductive hypothesis} \\ &= \llbracket C_1 + C_2 \rrbracket_{bai}^A(P_1) \vee \llbracket C_1 + C_2 \rrbracket_{bai}^A(P_2) \quad \text{Rearranging } \vee \end{aligned}$$

- $C^*$ :

$$\llbracket C^* \rrbracket_{bai}^A(P_1 \vee P_2) = \text{lfp}(\lambda P' \rightarrow P_1 \vee P_2 \vee \llbracket C \rrbracket_{bai}^A(P'))$$

$$\text{Let } F_i = \text{lfp}(\lambda P' \rightarrow P_i \vee \llbracket C \rrbracket_{bai}^A(P'))$$

$$\begin{aligned} (\lambda P' \rightarrow P_1 \vee P_2 \vee \llbracket C \rrbracket_{bai}^A(P'))(F_1 \vee F_2) &= P_1 \vee P_2 \vee \llbracket C \rrbracket_{bai}^A(F_1 \vee F_2) \\ &\quad \text{By inductive hypothesis} \\ &= P_1 \vee P_2 \vee \llbracket C \rrbracket_{bai}^A(F_1) \vee \llbracket C \rrbracket_{bai}^A(F_2) \\ &= P_1 \vee \llbracket C \rrbracket_{bai}^A(F_1) \vee P_2 \vee \llbracket C \rrbracket_{bai}^A(F_2) \\ &= F_1 \vee F_2 \end{aligned}$$



Thus  $F_1 \vee F_2$  is a fixpoint for  $\lambda P' \rightarrow P_i \vee \llbracket C \rrbracket_{bai}^A(P')$ .

Following the same reasoning is also the least one by  $F_1 \vee F_2$  being the smallest  $K \geq F_1$  and  $K \geq F_2$

**Theorem 9 (Soundness of the rule merge on attitive base commands)** if  $\forall P_1, P_2 \in A$  and  $b \in Base$   $\llbracket B \rrbracket_{bai}^A(P_1 \vee P_2) = \llbracket B \rrbracket_{bai}^A(P_1) \vee \llbracket B \rrbracket_{bai}^A(P_2)$  then:  
 $\forall P_1, P_2, Q \in A \quad C \in \mathbb{C}$

$$\llbracket C \rrbracket_{bai}^A(P_1) \leq Q \text{ and } \llbracket C \rrbracket_{bai}^A(P_2) \leq Q \implies \llbracket C \rrbracket_{bai}^A(P_1 \vee P_2) \leq Q$$

**Proof 5** Let  $S_i = \llbracket C \rrbracket_{bai}^A(P_i)$  by Theorem 8  $\llbracket C \rrbracket_{bai}^A(P_1 \vee P_2) = \llbracket C \rrbracket_{bai}^A(P_1) \vee \llbracket C \rrbracket_{bai}^A(P_2) \leq Q \vee Q = Q$

But requiring all the base commands to be additive is a strong requirement with  $\mathcal{P}(\mathbb{S})$  the requirement is equivalent as requiring  $\gamma$  especially when the base commands are defined trough a Galois connection to be additive, a requiring satisfied by a very small portion of abstract domains utilized in practice.

**Theorem 10 (Base commands additivity)** Given a Galois connection  $\langle D, \leq_D \rangle \xrightleftharpoons[\alpha]{\gamma} \langle A, \leq_A \rangle$  and  $\llbracket b \rrbracket_{bai}^D$  additive.

$$\alpha \circ \llbracket b \rrbracket_{bai}^D \circ \gamma \text{ additive} \iff \gamma \text{ additive}$$

**Proof 6**

$$\begin{aligned} \alpha(\llbracket b \rrbracket_{bai}^A(\gamma(P_1 \vee P_2))) &= \alpha(\llbracket b \rrbracket_{bai}^A(\gamma(P_1) \vee \gamma(P_2))) \\ &= \alpha(\llbracket b \rrbracket_{bai}^A(\gamma(P_1)) \vee \llbracket b \rrbracket_{bai}^A(\gamma(P_2))) \\ &= \alpha(\llbracket b \rrbracket_{bai}^A(\gamma(P_1))) \vee \alpha(\llbracket b \rrbracket_{bai}^A(\gamma(P_2))) \end{aligned}$$

#### 4.2.1 Local condition

Given that additivity conditions on the domain  $A$  are too restrictive we will look at local conditions.

One could think that requiring  $\gamma$  to be additive on  $P_1 \vee P_2$  is enough giving a rule like this:

$$\frac{\vdash \langle P_1 \rangle_A C \langle Q \rangle \quad \vdash \langle P_2 \rangle_A C \langle Q \rangle \quad \gamma(P_1 \vee P_2) = \gamma(P_1) \vee \gamma(P_2)}{\vdash \langle P_1 \vee P_2 \rangle_A C \langle Q \rangle} \text{ (merge)}$$

**Example 4.3 ((merge) rule is unsound)** Let  $A$  and  $\llbracket b \rrbracket_{bai}^A$  be the same as in 4.1.

By picking  $P_1 = x \in [0, 1]$  and  $P_2 = x \in [2, 2]$  we can obtain the following judgments:

- $\langle P_1 \rangle_A C \langle \perp \rangle$

- $\langle P_2 \rangle_A C \langle \perp \rangle$

And by applying the (merge) rule we can obtain:  $\langle P_1 \vee P_2 \rangle_A C \langle Q \rangle$  but by running command  $C$  on  $P_1 \vee P_2$  we obtain  $x \in [1, 1]$ .

$$\begin{aligned}
\llbracket C \rrbracket_{bai}^A(P_1 \vee P_2) &= \llbracket C \rrbracket_{bai}^A(x \in [0, 2]) \\
&= \llbracket x = 1? \rrbracket_{bai}^A(\llbracket x = 0? + x = 2? \rrbracket_{bai}^A(x \in [0, 2])) \\
&= \llbracket x = 1? \rrbracket_{bai}^A(\llbracket x = 0? \rrbracket_{bai}^A(x \in [0, 2]) \vee \llbracket x = 2? \rrbracket_{bai}^A(x \in [0, 2])) \\
&= \llbracket x = 1? \rrbracket_{bai}^A(x \in [0, 0]) \vee (x \in [2, 2]) \\
&= \llbracket x = 1? \rrbracket_{bai}^A(x \in [0, 2]) \\
&= x \in [1, 1]
\end{aligned}$$

The issue is caused by the imprecision added by the  $\vee$  introduced by the non deterministic choice.

In fact the following equivalence between programs that is true on the concrete domain  $(C_1 + C_2) \circ C_3 \equiv (C_1 \circ C_3) + (C_2 \circ C_3)$  it's not true in the abstract, the program  $C' = (x = 0? \circ x = 1?) + (x = 2? \circ x = 1?)$  that should be equivalent to  $C$  it's not:  $\llbracket C' \rrbracket_{bai}^A(P_1 \vee P_2) = \perp$ .

The fact that programs  $C$  and  $C'$  are different when interpreted by  $\llbracket \cdot \rrbracket_{bai}^A$  means that we can't look at them as a KAT, in fact this would violate the axiom  $(p+q)r = pr+qr$ . It should also be noted that the dual axiom  $r(p+q) = rp+rq$  instead holds.

**Theorem 11**  $\forall C_1 C_2 C_3$

$$\llbracket C_1 \circ (C_2 + C_3) \rrbracket_{bai}^A = \llbracket (C_1 \circ C_2) + (C_1 \circ C_3) \rrbracket_{bai}^A$$

**Proof 7**

$$\begin{aligned}
\llbracket C_1 \circ (C_2 + C_3) \rrbracket_{bai}^A(P) &= \llbracket C_2 \rrbracket_{bai}^A(\llbracket C_1 \rrbracket_{bai}^A(P)) \vee \llbracket C_3 \rrbracket_{bai}^A(\llbracket C_1 \rrbracket_{bai}^A(P)) \\
&= \llbracket C_1 \circ C_2 \rrbracket_{bai}^A(P) \vee \llbracket C_1 \circ C_3 \rrbracket_{bai}^A(P) \\
&= \llbracket C_1 \circ C_2 + C_1 \circ C_3 \rrbracket_{bai}^A(P)
\end{aligned}$$

Requiring  $(C_1 + C_2) \circ C_3 \equiv (C_1 \circ C_3) + (C_2 \circ C_3)$  is equivalent to requiring the abstract semantics of every program to be additive:

**Theorem 12**

$$\begin{aligned}
\forall C_1 C_2 C_3 \quad \llbracket (C_1 + C_2) \circ C_3 \rrbracket_{bai}^A &= \llbracket (C_1 \circ C_3) + (C_2 \circ C_3) \rrbracket_{bai}^A \\
&\iff \\
\forall P_1 P_2 (decidable) C' \quad \llbracket C' \rrbracket_{bai}^A(P_1 \vee P_2) &= \llbracket C_1 \rrbracket_{bai}^A(P_1) \vee \llbracket C_2 \rrbracket_{bai}^A(P_2)
\end{aligned}$$

**Proof 8** • (  $\Leftarrow$  ):

$$\begin{aligned}
\llbracket (C_1 + C_2) \circ C_3 \rrbracket_{bai}^A(P) &= \llbracket C_3 \rrbracket_{bai}^A(\llbracket C_1 \rrbracket_{bai}^A(P) \vee \llbracket C_2 \rrbracket_{bai}^A(P)) \\
&= \llbracket C_3 \rrbracket_{bai}^A(\llbracket C_1 \rrbracket_{bai}^A(P)) \vee \llbracket C_3 \rrbracket_{bai}^A(\llbracket C_2 \rrbracket_{bai}^A(P)) \\
&= \llbracket (C_1 \circ C_3) + (C_2 \circ C_3) \rrbracket_{bai}^A(P)
\end{aligned}$$

- (  $\Rightarrow$  ): For every proposition  $P$  we can construct a program  $c(P)$  such that  $\llbracket c(P) \rrbracket_{bai}^A(Q) = P$  (we just need to provide a program for every join-irreducible element in  $A$ )

$$\begin{aligned}
\llbracket C \rrbracket_{bai}^A(P_1 \vee P_2) &= \llbracket C \rrbracket_{bai}^A(\llbracket c(P_1) \rrbracket_{bai}^A(Q) \vee \llbracket c(P_2) \rrbracket_{bai}^A(Q)) \\
&= \llbracket C \rrbracket_{bai}^A(\llbracket c(P_1) + c(P_2) \rrbracket_{bai}^A(Q)) \\
&= \llbracket (c(P_1) + c(P_2)) \circ C \rrbracket_{bai}^A(Q) \\
&= \llbracket (c(P_1) \circ C) + (c(P_2) \circ C) \rrbracket_{bai}^A(Q) \\
&= \llbracket C \rrbracket_{bai}^A(\llbracket c(P_1) \rrbracket_{bai}^A(Q)) \vee \llbracket C \rrbracket_{bai}^A(\llbracket c(P_2) \rrbracket_{bai}^A(Q)) \\
&= \llbracket C \rrbracket_{bai}^A(P_1) \vee \llbracket C \rrbracket_{bai}^A(P_2)
\end{aligned}$$