2.3 Concrete and abstract interpretation

To evaluate a program *call* in the concrete semantics, its meaning is the set of states reachable from the initial state $\varsigma_0 = (call, [], [], t_0)$:

$$\{\varsigma : \varsigma_0 \Rightarrow^* \varsigma\}.$$

A naïve abstract interpreter could behave similarly, exploring the states reachable from the initial state $\hat{\varsigma} = (call, [], \bot, \hat{t}_0)$:

$$\{\hat{\varsigma} : \hat{\varsigma}_0 \leadsto^* \hat{\varsigma}\}.$$

In practice, widening on value environments [5] accelerates convergence [16, 27].

2.4 Parameters for the analysis framework

Time-stamp incrementers and binding allocators serve as parameters:

$$\begin{array}{ll} alloc : \mathsf{Var} \times Time \to Bind & \widehat{alloc} : \mathsf{Var} \times \widehat{Time} \to \widehat{Bind} \\ tick : \mathsf{Call} \times Time \to Time & \widehat{tick} : \mathsf{Call} \times \widehat{Time} \to \widehat{Time}. \end{array}$$

Time-stamps are designed to encode context/history. Thus, the abstract time-stamp incrementer $\widehat{tick}$ and the abstraction map α^η decide how much history to retain in the abstraction. As a result, the function $\widehat{tick}$ determines the context-sensitivity of the analysis. Similarly, the abstract binding allocator chooses how to allocate abstract bindings to variables, and in doing so, it fixes the polyvariance of the analysis. Once the parameters are fixed, the semantics must obey a straightforward soundness theorem:

Theorem 1. *If $\alpha^\eta(\varsigma) \sqsubseteq \hat{\varsigma}$ and $\varsigma \Rightarrow \varsigma'$, then there exists a state $\hat{\varsigma}'$ such that $\hat{\varsigma} \leadsto \hat{\varsigma}'$ and $\alpha^\eta(\varsigma') \sqsubseteq \hat{\varsigma}'$.*

3 Analogy: Singleton abstraction to binding anodization

Focusing on our goal of solving the generalized environment problem—reasoning about the equality of individual bindings—we turn to singleton abstraction [4]. Singleton abstraction has been used in pointer and shape analyses to drive must-alias analysis; we extend singleton abstraction, and the framework of anodization, to determine the equivalence of bindings to the *same* variable. That is, we will be able to solve the environment problem with our singleton abstraction, but not the *generalized* environment problem. In Section 4, we will solve the generalized problem by bootstrapping binding invariants on top of anodization.

A Galois connection [6] $X \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} \hat{X}$ has a singleton abstraction iff there exists a subset $\hat{X}_1 \subseteq \hat{X}$ such that for all $\hat{x} \in \hat{X}_1$, $size(\gamma(\hat{x})) = 1$. The critical property of singleton abstractions is that equality of abstract representatives implies equality of their concrete constituents. Hence, when the set X contains addresses, singleton abstractions enable must-alias analysis. Analogously, when the set X contains bindings, singleton abstraction enables binding-equality testing.