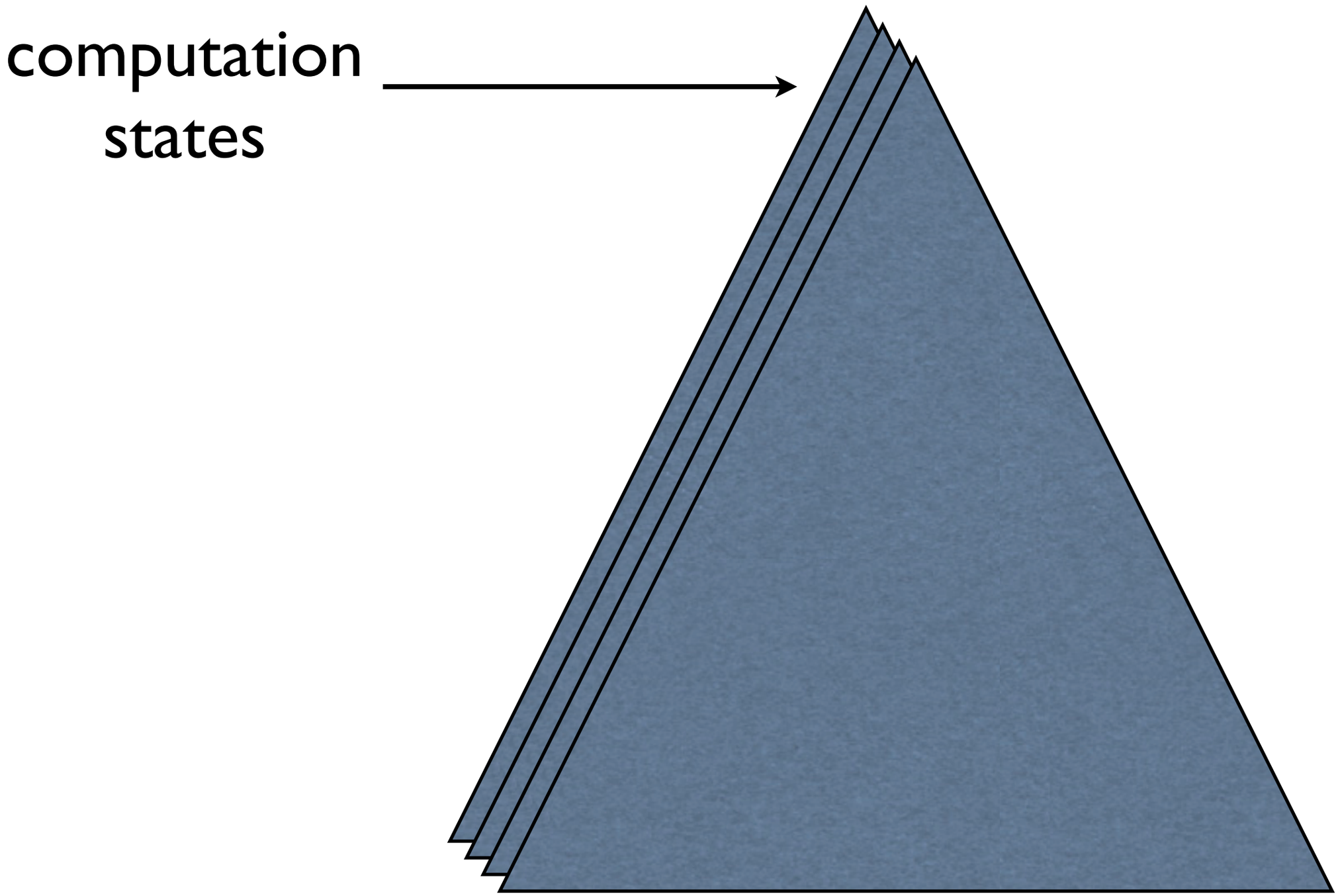
Using Typings as Types

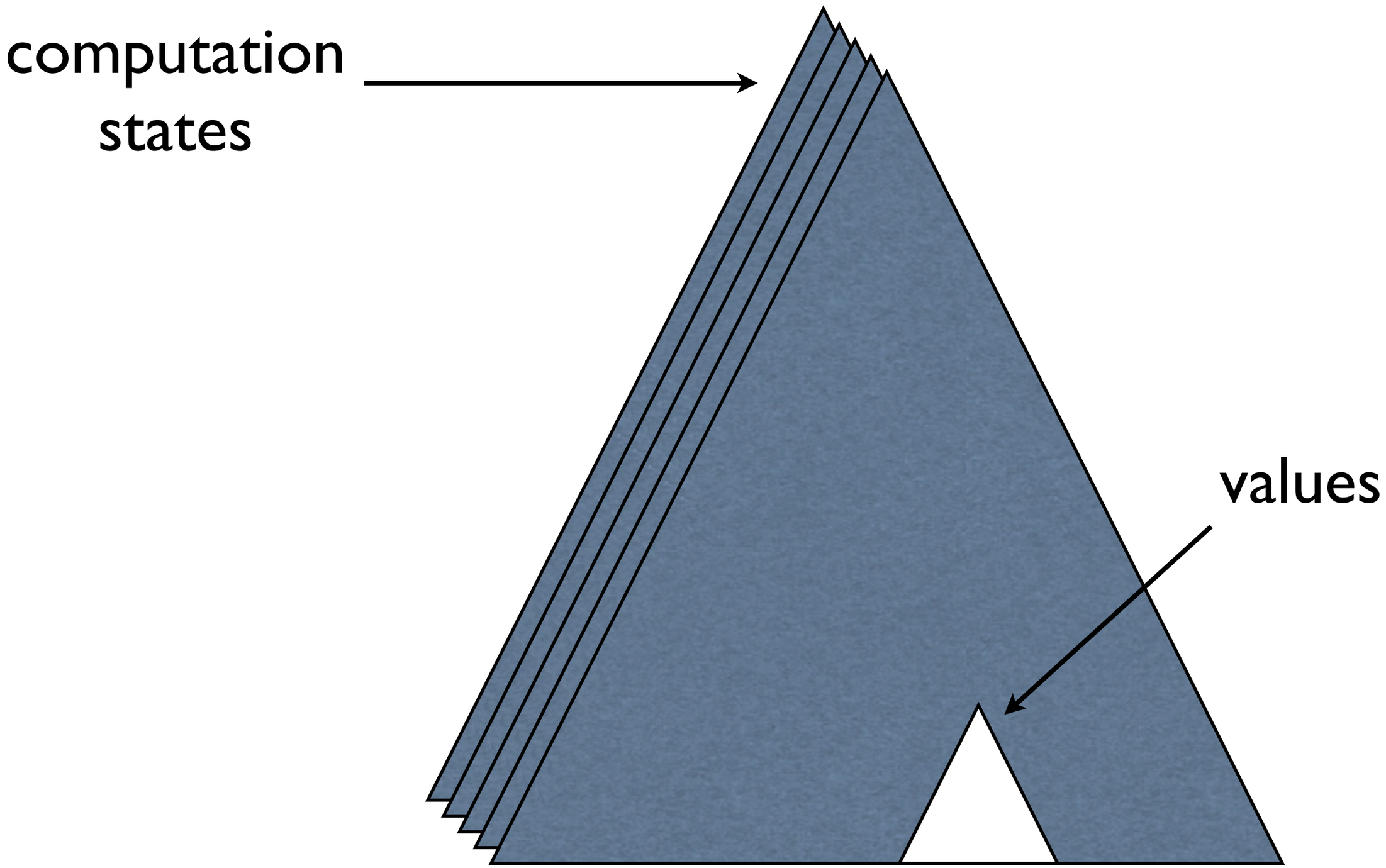
Casper Bach Poulsen
TU Delft, The Netherlands
Peter Mosses, Neil Sculthorpe
Swansea University, UK

NWPT 2015, Reykjavík, Iceland, October 2015

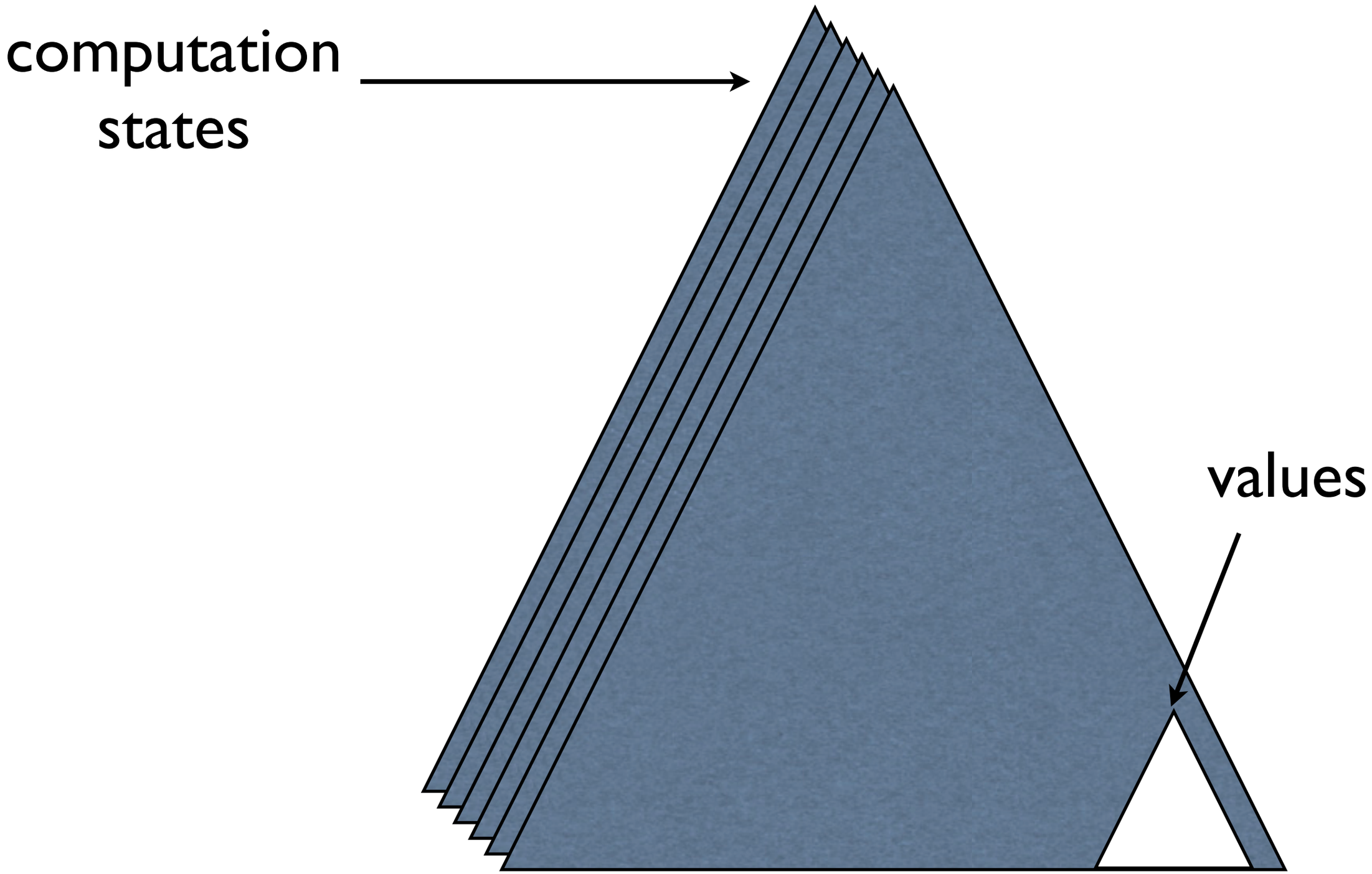
Computations and values



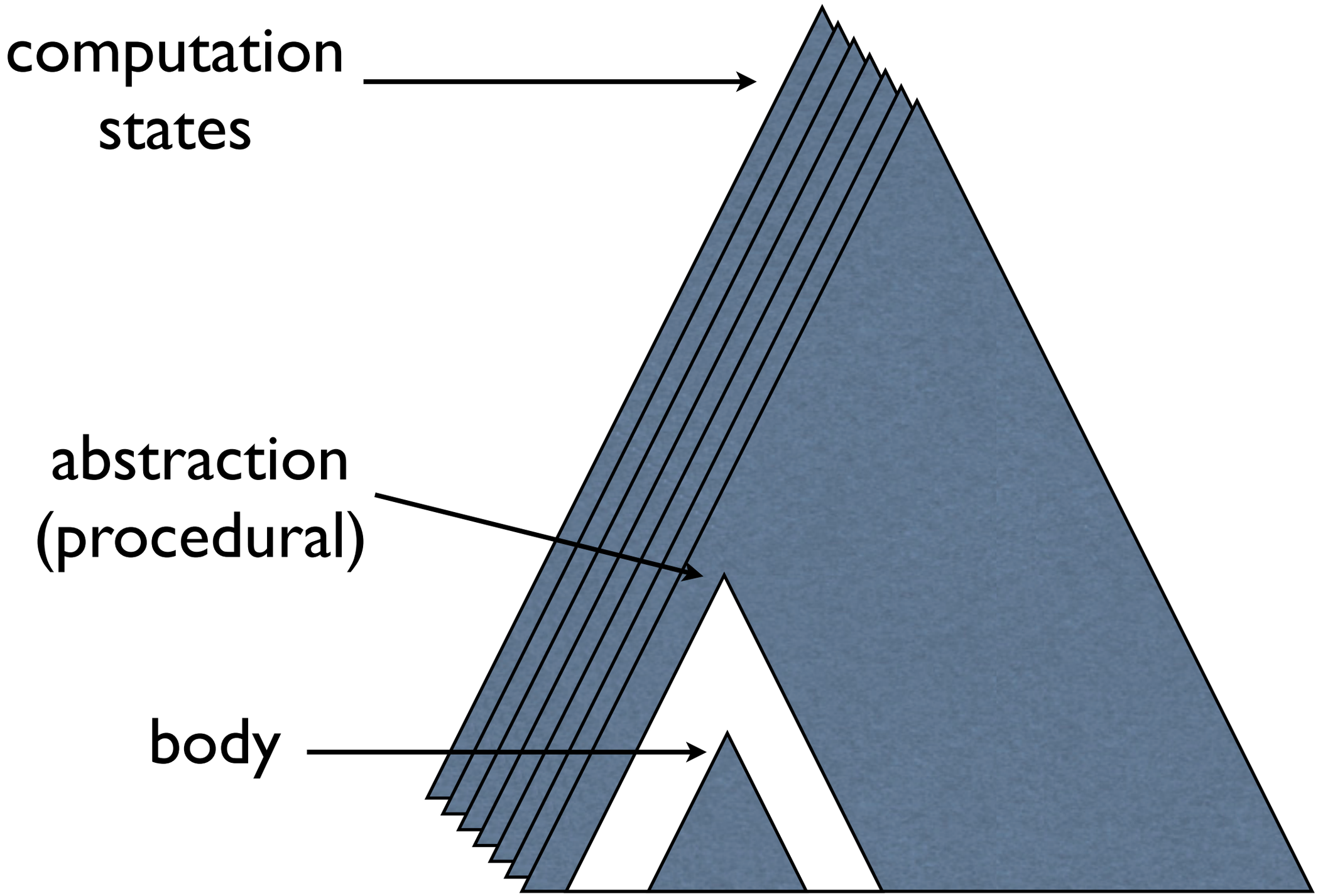
Computations and values



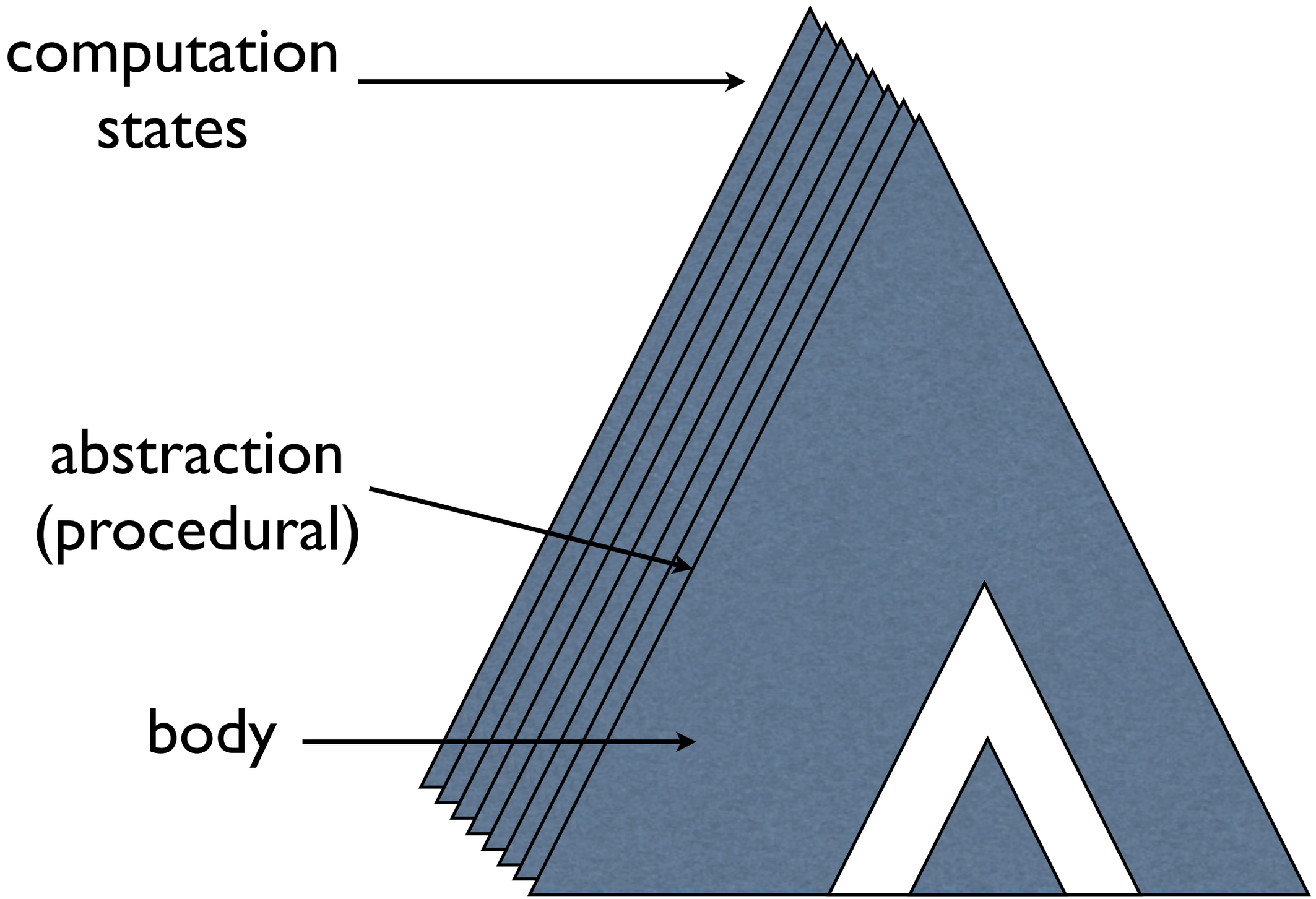
Computations and values



Computations and values



Computations and values



An example of dynamic scope

$$(\lambda f.\ (\lambda y.\ f\ 1)\ 2)\ (\lambda x.\ x + y)$$

$$\text{let } f = (\lambda x.\ x + y) \text{ in let } y = 2 \text{ in } f\ 1$$

$$(\{y : \text{int}\} \vdash (\text{int} \to \text{int})) \qquad \text{int}$$

Static scope

$$M, N ::= n \mid x \mid (M + N) \mid (\lambda x.M) \mid (M\ N)$$

$$\sigma, \tau ::= \mathsf{int} \mid t \mid (\sigma \to \tau)$$

$$\frac{A_x \cup \{x : \sigma\} \vdash M : \tau}{A \vdash (\lambda x.M) : (\sigma \to \tau)} \qquad \frac{A \vdash M : (\sigma \to \tau) \quad A \vdash N : \sigma}{A \vdash (M\ N) : \tau}$$

$$\frac{A \vdash M : \tau}{A' \vdash M : \tau}\ A \subseteq A'$$

Dynamic scope

$$M, N ::= n \mid x \mid (M + N) \mid (\lambda x.M) \mid (M\ N)$$

$$\sigma, \tau ::= \mathsf{int} \mid t \mid (\sigma \to \tau) \mid (A \vdash \tau) \mid (\sigma \wedge \tau)$$

$$\frac{A \vdash M : \tau}{\emptyset \vdash (\lambda x.M) : (A_x \vdash (\sigma \to \tau))}\ (A(x)=\sigma)$$

Dynamic scope

$$M, N ::= n \mid x \mid (M + N) \mid (\lambda x.M) \mid (M\ N)$$

$$\sigma, \tau ::= \mathsf{int} \mid t \mid (\sigma \to \tau) \mid (A \vdash \tau) \mid (\sigma \wedge \tau)$$

$$\frac{A_x \cup \{x : \sigma\} \vdash M : \tau}{\emptyset \vdash (\lambda x.M) : (A_x \vdash (\sigma \to \tau))}$$

$$\frac{A \vdash M : (A' \vdash (\sigma \to \tau)) \quad A \vdash N : \sigma}{(A \wedge A') \vdash (M\ N) : \tau}$$

$$\frac{A \vdash M : \sigma}{A' \vdash M : \tau}\ (A \vdash \sigma) <: (A' \vdash \tau)$$

Explicit closures

$$M ::= \cdots \mid \mathsf{close}(M)$$

$$\frac{A \vdash M : (A' \vdash (\sigma \to \tau))}{(A \wedge A') \vdash \mathsf{close}(M) : (\emptyset \vdash (\sigma \to \tau))}$$

$$\frac{A_x \cup \{x : \sigma\} \vdash M : \tau}{\dfrac{\emptyset \vdash (\lambda x.M) : (A_x \vdash (\sigma \to \tau))}{\emptyset \wedge A_x \vdash \mathsf{close}(\lambda x.M) : (\ \emptyset \vdash (\sigma \to \tau))}}$$

$$\frac{A \vdash M : (\emptyset \vdash (\sigma \to \tau)) \quad A \vdash N : \sigma}{(A \wedge \emptyset) \vdash (M\ N) : \tau}$$

Some subtyping rules

$$\frac{\sigma' <: \sigma \qquad \tau <: \tau'}{(\sigma \to \tau) <: (\sigma' \to \tau')}$$

$$(A_x \cup \{x : \tau\}) <: A_x$$

$$\frac{A' <: A \qquad \sigma <: \tau}{(A \vdash \sigma) <: (A' \vdash \tau)}$$

$$\frac{A_x <: A'_x \qquad \tau <: \tau'}{(A_x \cup \{x : \tau\}) <: (A'_x \cup \{x : \tau'\})}$$

Related work

Implicit Parameters: Dynamic Scoping with Static Types

Jeffrey R. Lewis* Mark B. Shields* Erik Meijer† John Launchbury*

[POPL 2000]

A Polymorphic Modal Type System for Lisp-Like Multi-Staged Languages *

Ik-Soon Kim Kwangkeun Yi Cristiano Calcagno

[POPL 2006]

ISOLATE: A Type System for Self-recursion

Ravi Chugh

[ESOP 2015]

Typings as types

$$A \vdash M : \tau \quad \Rightarrow \quad M : (A \vdash \tau)$$

$$\frac{A_x \cup \{x : \sigma\} \vdash M : \tau}{A \vdash (\lambda x.M) : (\sigma \to \tau)} \quad \Rightarrow \quad \frac{M : (A_x \cup \{x : \sigma\} \vdash \tau)}{(\lambda x.M) : (A \vdash (\sigma \to \tau))}$$

$$\frac{A \vdash M : (\sigma \to \tau) \quad A \vdash N : \sigma}{A \vdash (M\ N) : \tau} \quad \Rightarrow \quad \frac{M : (A \vdash (\sigma \to \tau)) \quad N : (A \vdash \sigma)}{(M\ N) : (A \vdash \tau)}$$

Conclusion

Novel type system

- motivated by considering abstractions as values
- applicable to general dynamic scope
- exploits typings and intersection types

Future work

- meta-theory (soundness, …)
- generalise to include stores, exceptions, …
- achieve modularity
  - *implicit propagation of omitted type features*

Appendix

Implicit parameters ... *[POPL 2000]*

λ-vars	x, y, z		
`let`-vars	p, q		
Implicit vars	$?x, ?y, ?z$		
Terms	t, u, v	$::=$	$x \mid p \mid ?x$
		$\mid$	$\lambda x.\ t \mid t\ u$
		$\mid$	$\texttt{let}\ p = u\ \texttt{in}\ t$
		$\mid$	$t\ \texttt{with}\ ?x = u$
Type vars	α, β		
Types	τ, υ	$::=$	α
		$\mid$	$\upsilon \to \tau$
Schemes	σ, φ	$::=$	$\forall\overline{\alpha}.\ C \Rightarrow \tau$
Contexts	C, D	$::=$	$?x_1 : \tau_1, \ldots, ?x_n : \tau_n$
			$?x_1, \ldots, ?x_n$ distinct
Type contexts	Γ	$::=$	$x_1 : \sigma_1, \ldots, x_n : \sigma_n$
			$x_1, \ldots, x_n$ distinct

Multi-stage ... *[POPL 2006]*

$$\begin{array}{lllll}
\text{Types} & A, B & \in & \mathit{Type} & \\
\text{Type Environments} & \Gamma & \in & \mathit{TyEnv} & = \quad \mathit{Var} \xrightarrow{\mathit{fin}} \mathit{Type} \\
\text{Store Typings} & \Sigma & \in & \mathit{ST} & = \quad \mathit{Loc} \xrightarrow{\mathit{fin}} \mathit{Type}
\end{array}$$

$$\text{Types} \qquad A, B \quad ::= \quad \iota \mid A \to B \mid \Box(\Gamma \triangleright A) \mid A\ \texttt{ref}$$

IsoLate ... self-recursion *[ESOP 2015]*

$$
\begin{array}{rrcl}
\textbf{Types} & R, S, T & ::= & \texttt{unit} \mid \texttt{\{}\ \overline{f}\texttt{:}\overline{T}\ \texttt{\}} \mid S \to T \mid A \mid \mu A.T \mid \forall A.\ T \\
& \textit{pre-type} & \mid & (A : S) \Rightarrow T \\
\textbf{Type Environments} & \Gamma & ::= & - \mid \Gamma, x{:}T \mid \Gamma, A \mid \Gamma, A <: T
\end{array}
$$