A SPECTRUM OF SOUNDNESS AND PERFORMANCE

Ben Greenman

Northeastern University

Expressions

$\mathbf{e}$

Types

$\boldsymbol{\tau}$

Dynamic Type

Dyn

Precision relation

$\sqsubseteq$

Typing judgment

$\vdash \mathbf{e} : \boldsymbol{\tau}$

Evaluation Syntax

$\mathbf{e}$

Semantics

$\longrightarrow$

Expressions

e

Types

τ

Dynamic Type

Dyn

Precision relation

$\sqsubseteq$

Typing judgment

$\vdash \mathbf{e} : \boldsymbol{\tau}$

Evaluation Syntax

e

Semantics

$\longrightarrow$

Pair of Languages

e_S = x
 | $e_S\, e_S$
 | λ(x:τ)e_S
 | ....
 | dyn τ e_D

e_D = x
 | $e_D\, e_D$
 | λ(x)e_D
 | ....
 | stat τ e_S

τ = Int
 | Nat
 | (τ x τ)
 | (τ -> τ)

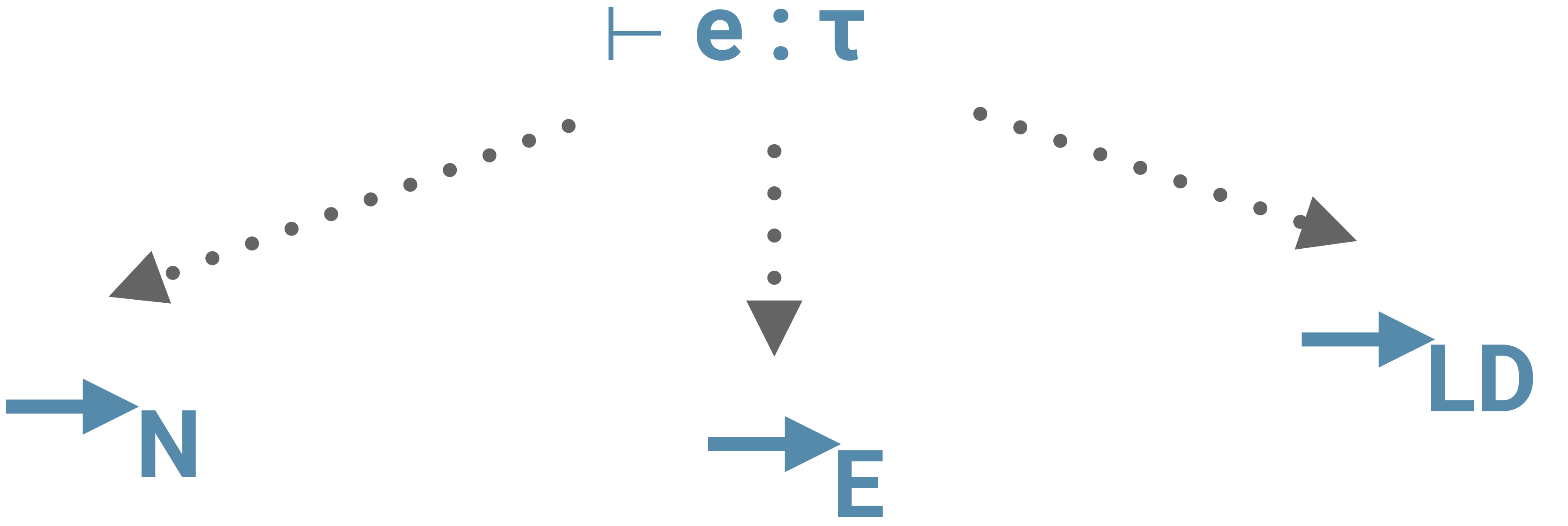
Typing judgment(s)

$\vdash e_S : \tau$

$\vdash e_D$

$$\frac{\vdash e_D}{\vdash \mathbf{dyn}\ \tau\ e_D : \tau}$$

$$\frac{\vdash e_S : \tau}{\vdash \mathbf{stat}\ \tau\ e_S}$$



$\vdash e : \tau$
N
E
LD

"types enforce levels of abstraction"

If $\mathbf{dyn}\ \boldsymbol{\tau}\ \mathbf{v_D} \rightarrow_{\mathbf{N}} \mathbf{v_S}$

then $\vdash \mathbf{v_S} : \boldsymbol{\tau}$

Boundary Terms

$\text{dyn Int } v_D \rightarrow_N v_D$ if v_D is an integer

$\text{dyn Nat } v_D \rightarrow_N v_D$ if v_D is a natural

$\text{dyn } (\tau \times \tau') \; v_D \rightarrow_N (\text{dyn } \tau \; v, \text{dyn } \tau' \; v')$

if $v_D = (v, v')$

$\text{dyn } (\tau \text{ -> } \tau') \; v_D \rightarrow_N (\text{mon } (\tau \text{ -> } \tau') \; v)$

if v_D is a function

Core Language

e_S = ...
| mon (τ -> τ') v_D

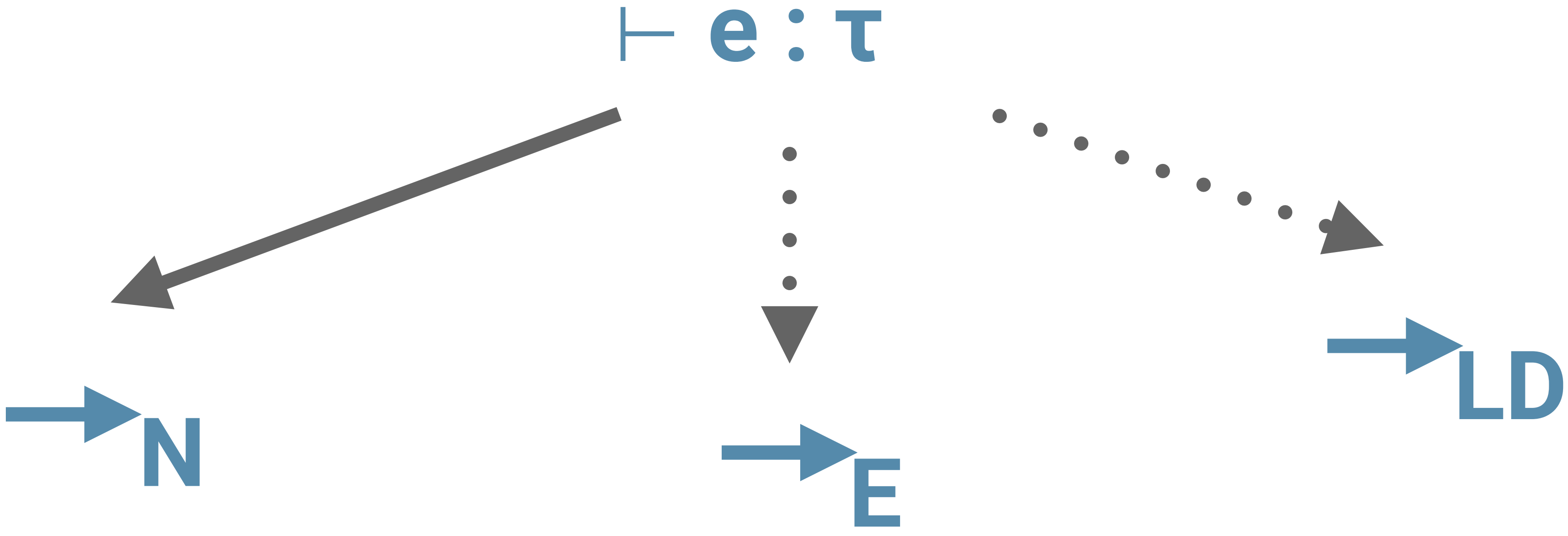
e_D = ...
| mon (τ -> τ') v_S

$$(\text{mon}\ (\tau \to \tau')\ v_D)\ v_S \longrightarrow_N (\text{stat}\ \tau'\ (v_D\ (\text{dyn}\ \tau\ v_S)))$$

Soundness (Natural Embedding)

If $\vdash \mathbf{e} : \tau$ then either:

- $\mathbf{e} \longrightarrow^{*}_{N} \mathbf{v_S}$ and $\vdash \mathbf{v_S} : \tau$
- $\mathbf{e} \longrightarrow^{*}_{N}$ **BoundaryErr**
- $\mathbf{e} \longrightarrow^{*}_{N} \mathbf{E[e_D]}$ and $\mathbf{e_D} \longrightarrow_{N}$ **DynErr**
- $\mathbf{e}$ diverges


⊢ e : τ
→N
→E
→LD

"types should not affect semantics"

If $\mathbf{dyn}\ \tau\ \mathbf{v}_D \rightarrow_E \mathbf{v}$

then $\vdash \mathbf{v}$

Boundary Terms

$\text{dyn Int } v_D \rightarrow_E v$

$\text{dyn Nat } v_D \rightarrow_E v$

$\text{dyn } (\tau \times \tau')\ v_D \rightarrow_E v$

$\text{dyn } (\tau \text{ -> } \tau')\ v_D \rightarrow_E v$

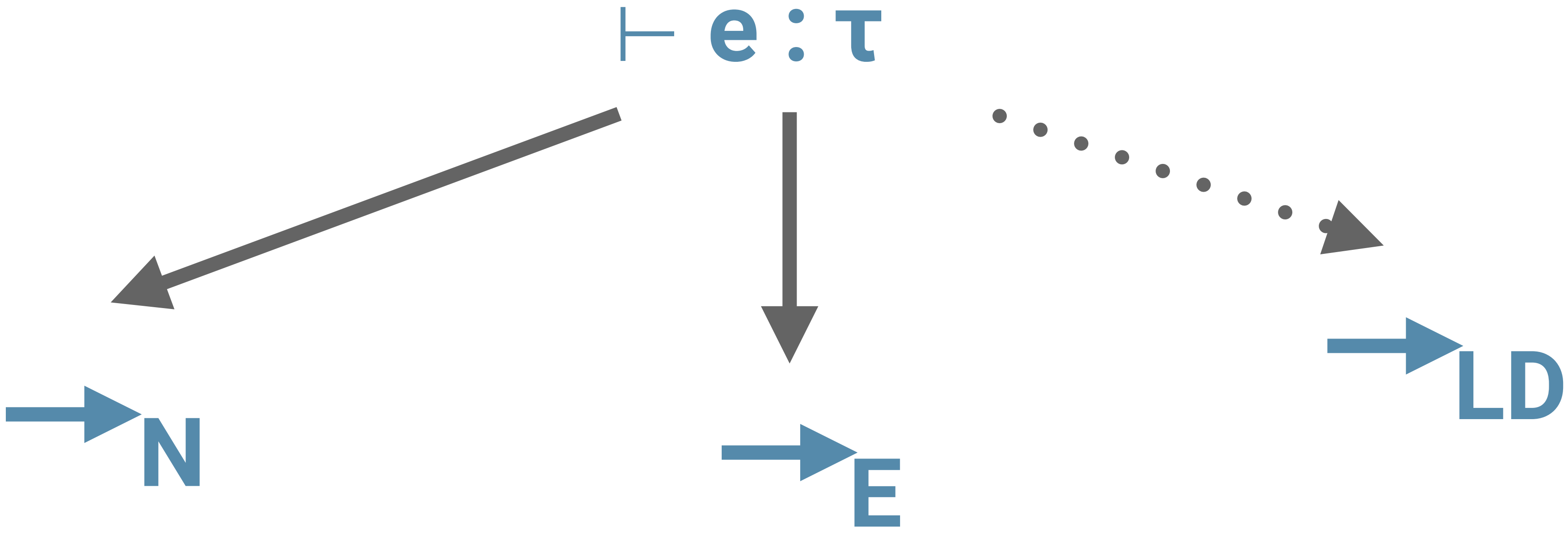
Core Language

Same as the surface language!

Soundness (Erasure Embedding)

If $\vdash \mathbf{e} : \boldsymbol{\tau}$ then either:

- $\mathbf{e} \rightarrow^{*}_{\mathbf{E}} \mathbf{v}$ and $\vdash \mathbf{v}$
- $\mathbf{e} \rightarrow^{*}_{\mathbf{E}}$ **BoundaryErr**
- $\mathbf{e} \rightarrow^{*}_{\mathbf{E}}$ **DynErr**
- $\mathbf{e}$ diverges


⊢ e : τ
N
E
LD

"types prevent undefined behavior"

"types prevent undefined behavior"

(1 2)

(fst #<fun>)

(#<fun> + #<fun>)

$\vdash$ **(-3 * -4) : Nat**

"types prevent undefined behavior"

"types prevent undefined behavior"

If $\mathbf{dyn}\ \tau\ v_D \rightarrow_{LD} v$

then $\vdash v : \lfloor \tau \rfloor$

where $\lfloor \mathbf{Int} \rfloor = \mathbf{Int}$ $\quad \lfloor \mathbf{Nat} \rfloor = \mathbf{Nat}$

$\lfloor (\tau \times \tau) \rfloor = \mathbf{Pair}$ $\quad \lfloor (\tau \to \tau) \rfloor = \mathbf{Fun}$

Boundary Terms

dyn Int v_D	$\to_{LD}$	v_D	if	v_D	is an integer
dyn Nat v_D	$\to_{LD}$	v_D	if	v_D	is a natural
dyn $(\tau \times \tau')$ v_D	$\to_{LD}$	v_D	if	v_D	is a pair
dyn $(\tau \text{ -> } \tau')$ v_D	$\to_{LD}$	v_D	if	v_D	is a function

✓ (2 + (dyn Int #<fun>))

✗ (2 + (fst (dyn (Int x Int) (#<fun>, #<fun>))))

✗ (2 + ((dyn (Int->Int) #<fun>) 3))

```
(2 + (dyn Int #<fun>))

(2 + (fst (dyn (Int x Int) (#<fun>, #<fun>))))
~~>  (2 + (check Int (fst ....)))

(2 + ((dyn (Int->Int) #<fun>) 3))
~~>  (2 + (check Int ((dyn ....) ....)))
```

$(\lambda(x : Nat)\ (x * x))$

~~>

```
(λ(x : Nat) (x * x))
~~> (λ(x : Nat) (check Nat x) ; (x * x))
```

Core Language

e_S = ...
    | check ⌊τ⌋ e_S

e_D = ...

check ⌊τ⌋ v $\longrightarrow_{LD}$ v

if v matches constructor ⌊τ⌋

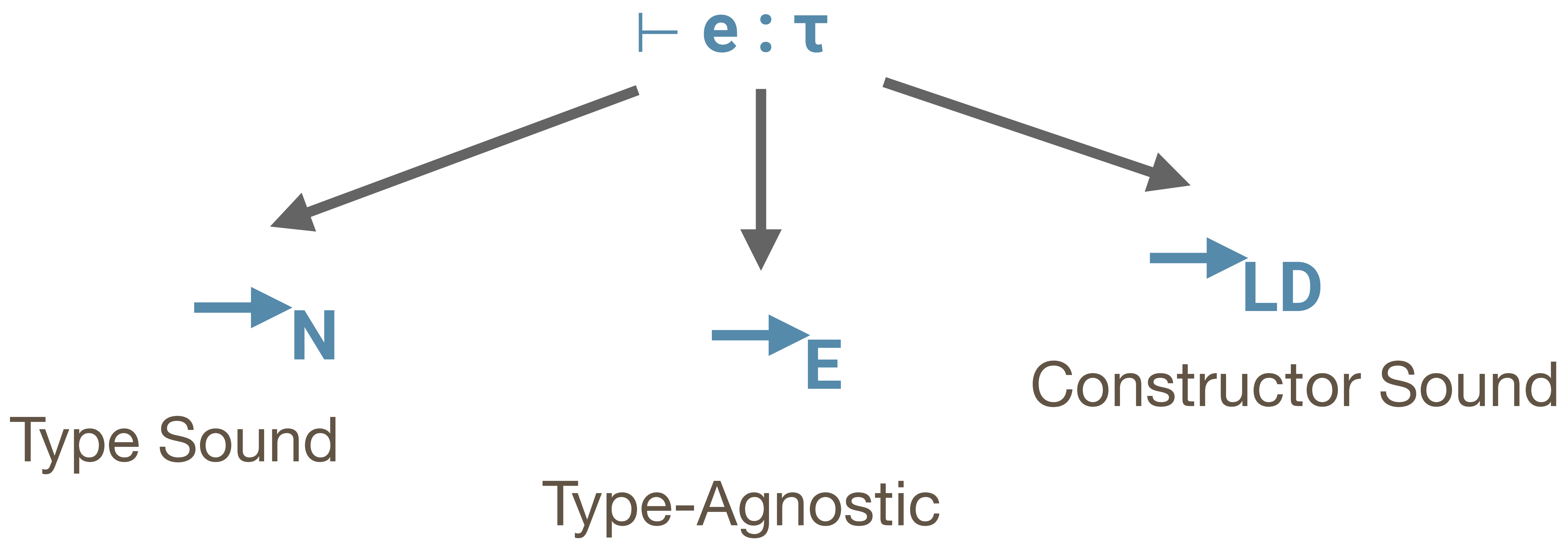
⊢ e : τ ~~> e'

⊢ e ~~> e'

Soundness (Locally-Defensive)

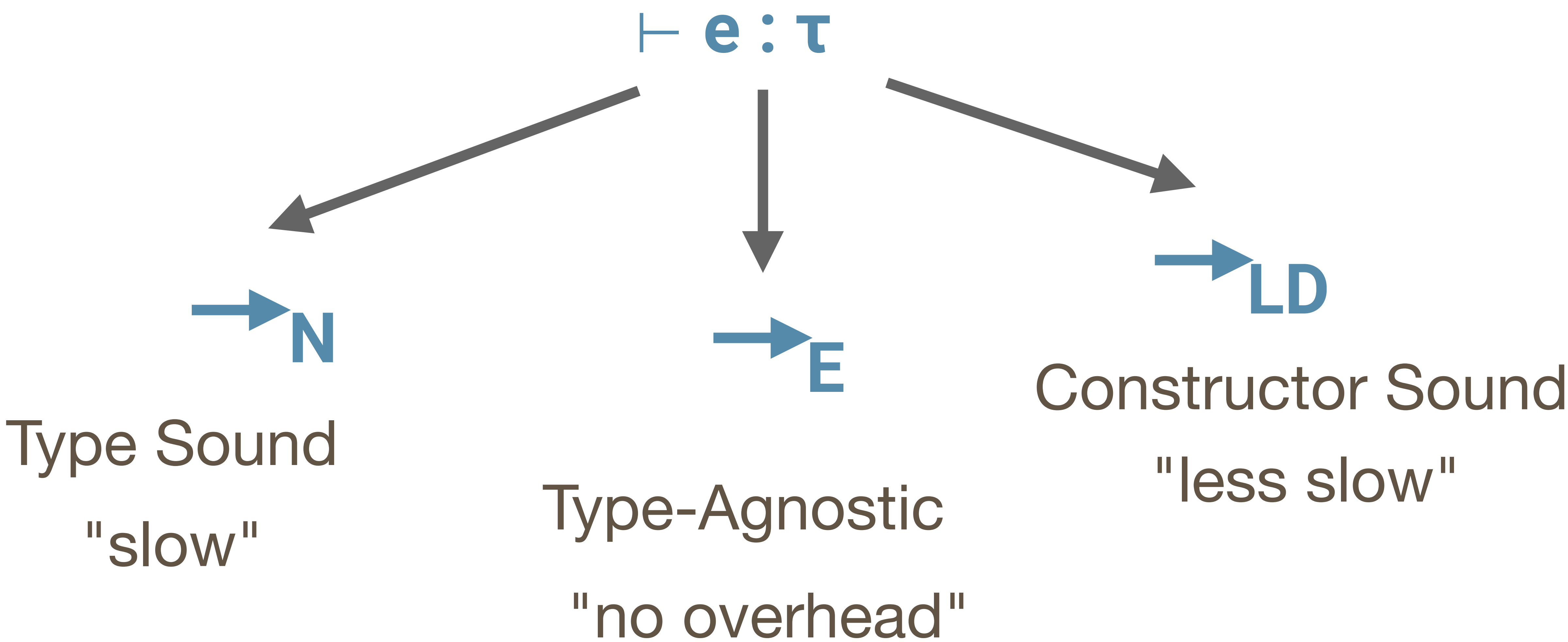
If $\vdash e : \tau$ then $\vdash e : \tau$ **~~>** e' and either:

- $e' \rightarrow^{*}_{LD} v_S$ and $\vdash v_S : \lfloor \tau \rfloor$
- $e' \rightarrow^{*}_{LD}$ **BoundaryErr**
- $e' \rightarrow^{*}_{LD} E[e_D]$ and $e_D \rightarrow_{LD}$ **DynErr**
- e' diverges

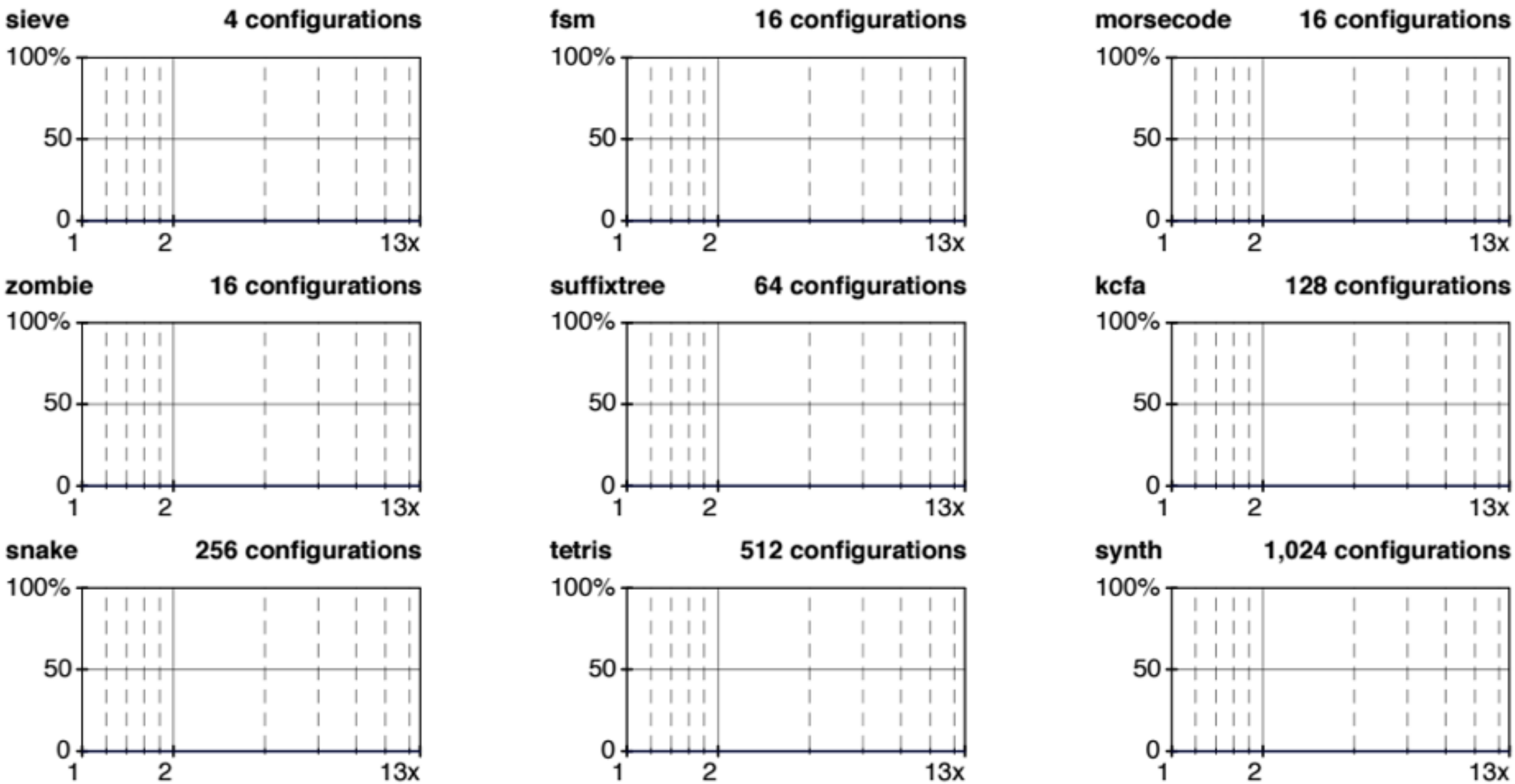


⊢ e : τ
→N
Type Sound
→E
Type-Agnostic
→LD
Constructor Sound

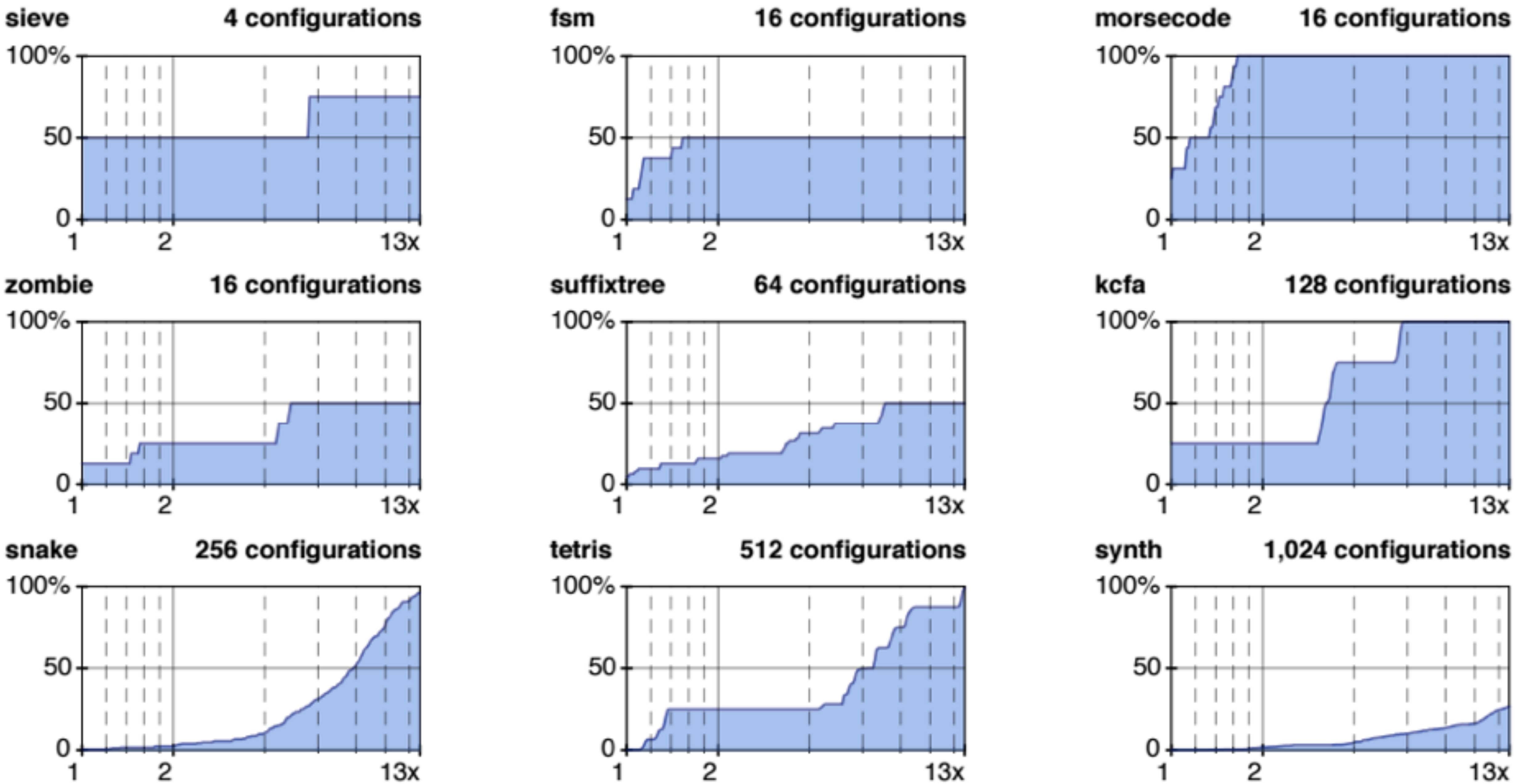
Performance?



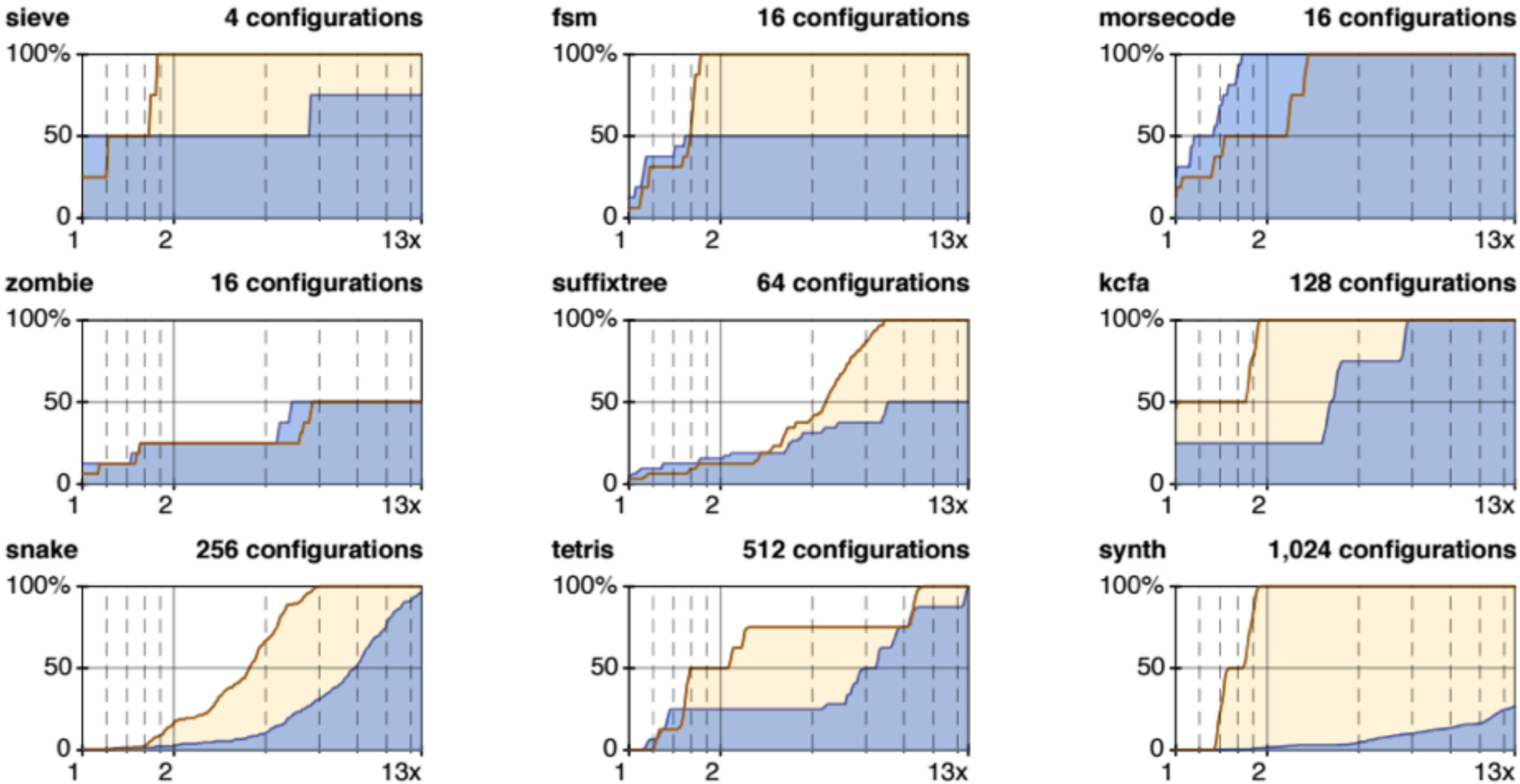
Performance



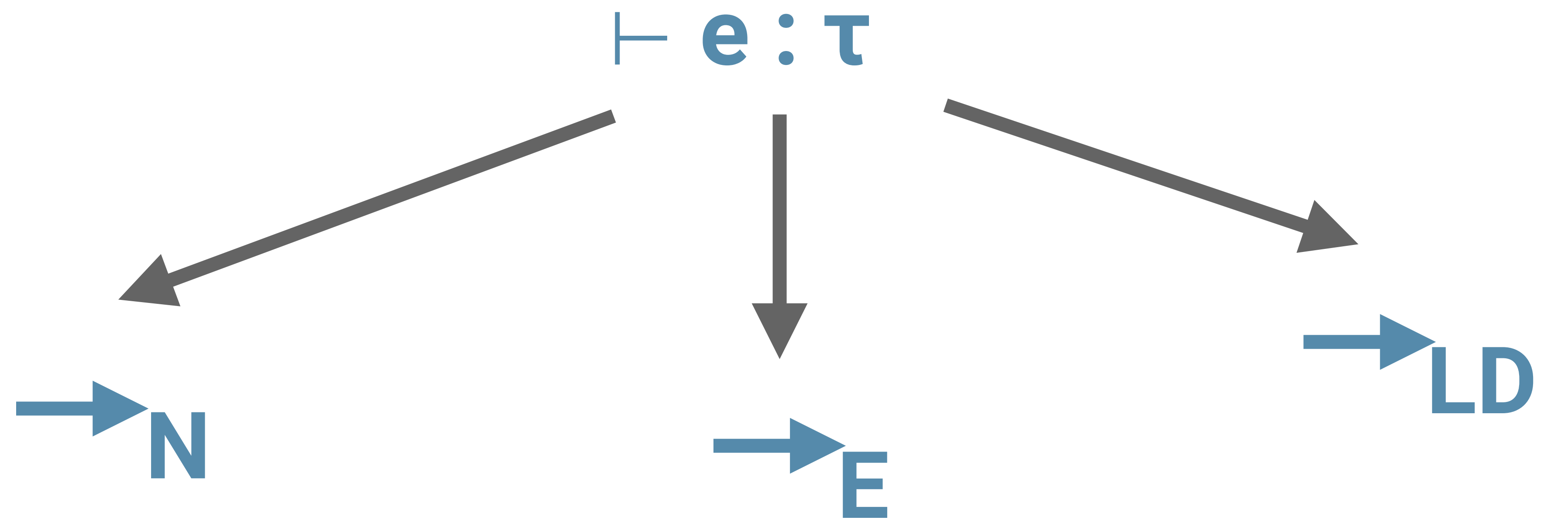
Performance



Performance

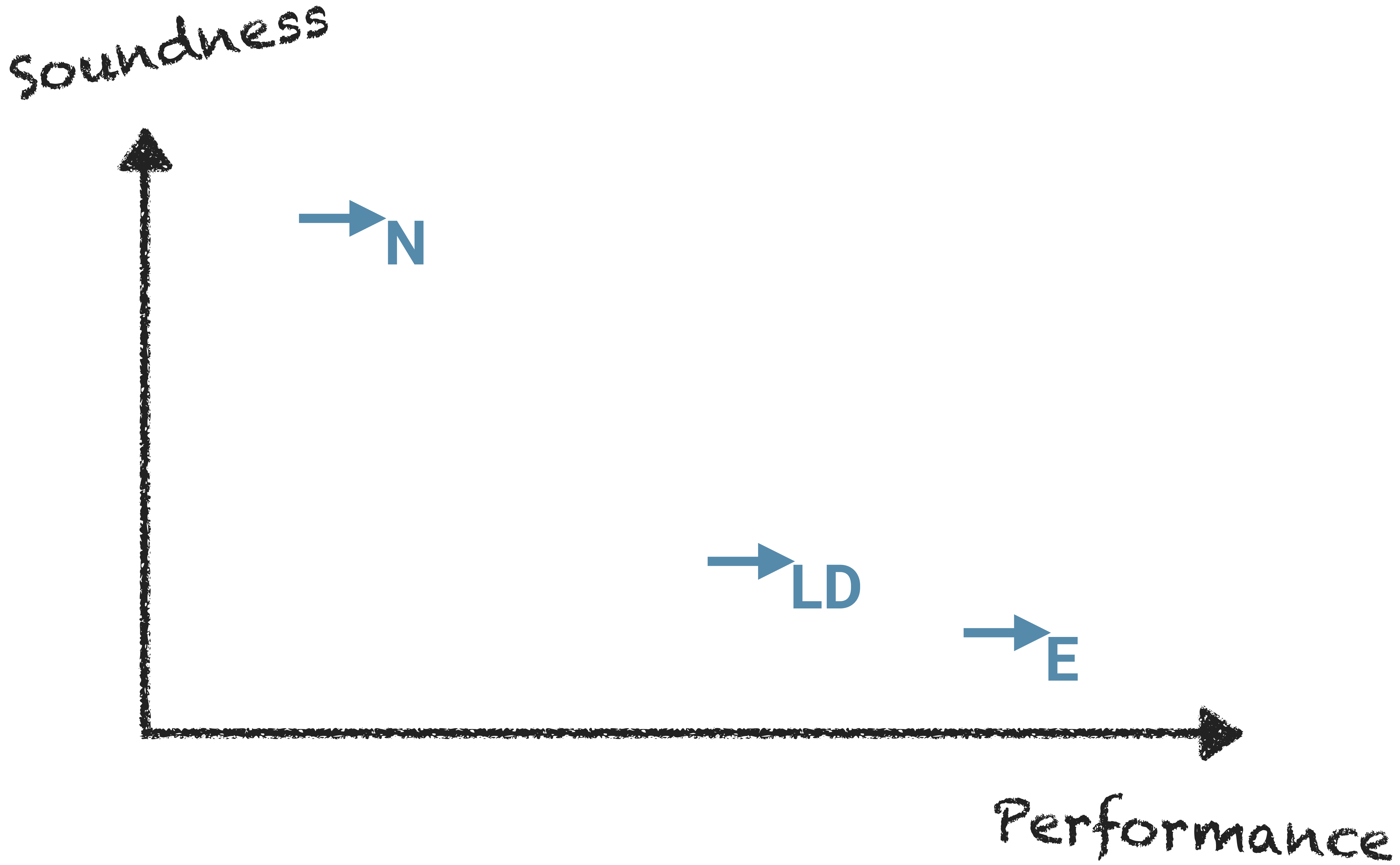


Metatheory

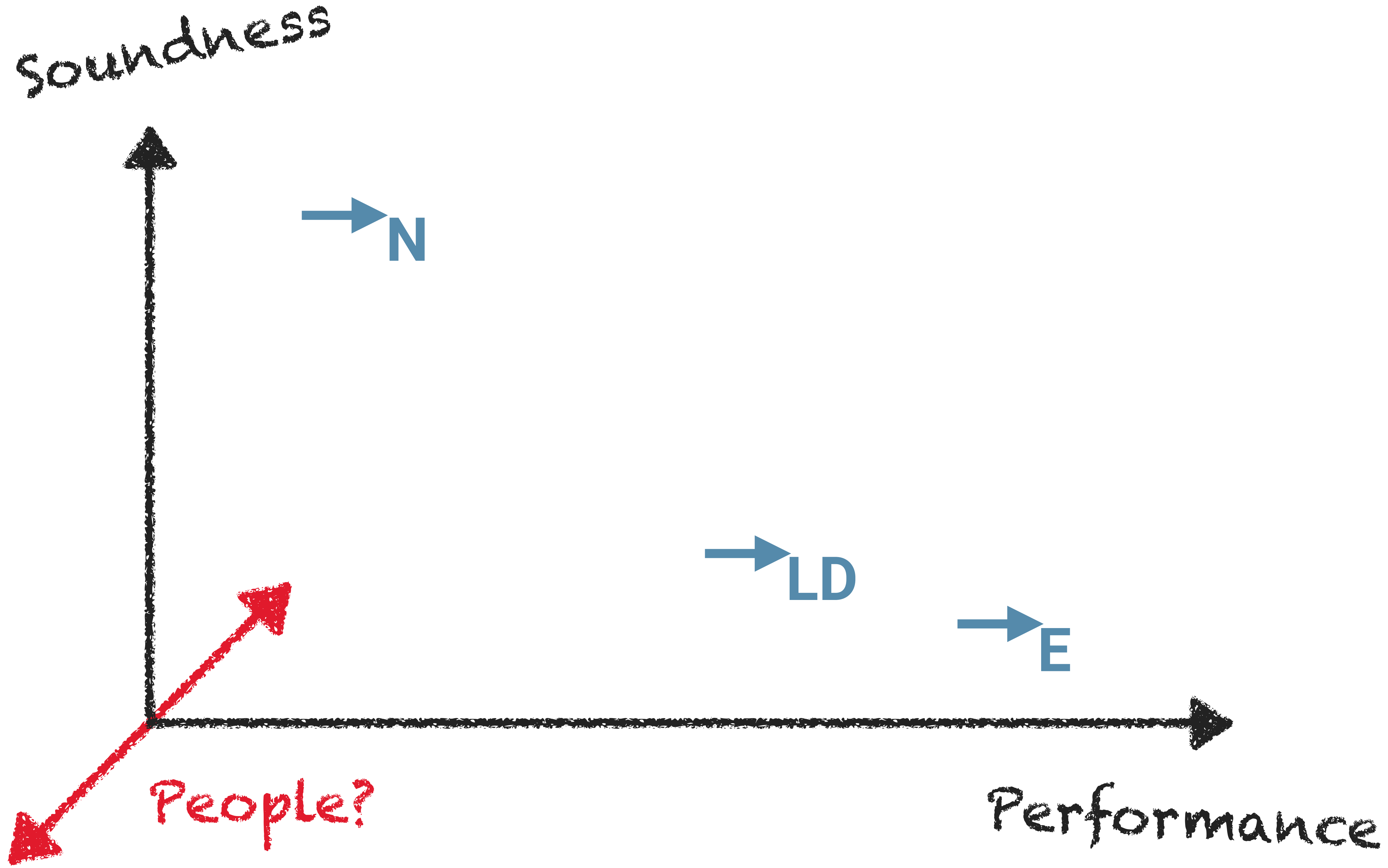


Metatheory

- Classic type soundness for boundary-free terms $\vdash \mathbf{e : \tau}$ $\rightarrow_{N}$ $\rightarrow_{E}$ $\rightarrow_{LD}$
- For boundaries of base type, $\rightarrow_{N}$ =~= $\rightarrow_{LD}$
- For boundaries of base type, $\rightarrow_{E}$ =/= $\rightarrow_{LD}$
- For errors, $\rightarrow_{N} \leq \rightarrow_{LD} \leq \rightarrow_{E}$



Soundness
N
LD
E
Performance


Soundness
N
LD
E
People?
Performance