

Scala Inline

No Author Given

No Institute Given

Abstract. [1]

Keywords: Partial Evaluation

1 Introduction

```
def dot[V: Numeric](v1: Vector[V], v2: Vector[V]): V =  
  (v1 zip v2).foldLeft(zero[V]){ case (prod, (c1, cr)) =>  
    prod + c1 * cr  
}
```

Fig. 1. Function for computing the dot product over generic `Numeric` values.

2 Formalization $F_{i<}$:

$t ::=$	Terms:
x, y	identifier
$(x : iT) \Rightarrow t$	function
$t(t)$	application
$\{x = t\}$	record
$t.x$	selection
$inline\ t$	inlining request
$[X <: iT] \Rightarrow t$	type abstraction
$t[iT]$	type application
$S, T, U ::=$	Types:
$iS \Rightarrow jT$	function type
$\{x : iS\}$	record type
$[X <: iS] \Rightarrow jT$	universal type
\top	top type
$iT, jT, kT, lT ::=$	Inlineable Types:
$T, dynamic\ T$	dynamic type
$static\ T$	static type
$inline\ T$	must inline type
$\Gamma ::=$	Contexts:
\emptyset	empty context
$\Gamma, x : iT$	term binding
$\Gamma, X <: iT$	type binding

Fig. 2. Syntax

$\frac{x : iT \in \Gamma}{\Gamma \vdash x : iT}$	(T-IDENT)
$\frac{\Gamma, x : iT_1 \vdash t : jT_2}{\Gamma \vdash (x : iT_1) \Rightarrow t : \text{static } iT_1 \Rightarrow jT_2}$	(T-FUNC)
$\frac{\Gamma \vdash t : iT}{\Gamma \vdash \{x = t\} : \text{static } \{x : iT\}}$	(T-REC)
$\frac{\Gamma \vdash t_1 : i(jT_1 \Rightarrow kT_2) \quad \Gamma \vdash t_2 : jT_2}{\Gamma \vdash t_1(t_2) : (i \wedge j \wedge k)T_2}$	(T-APP)
$\frac{\Gamma \vdash t : i\{x = jT_1, y = kT_2\}}{\Gamma \vdash t.x : (i \wedge j)T_1}$	(T-SEL)
$\frac{\Gamma \vdash t : \text{static } T}{\Gamma \vdash \text{inline } t : \text{inline } T}$	(T-INLINE)
$\frac{\Gamma, X <: iT_1 \vdash t_2 : jT_2}{\Gamma \vdash [X <: iT_1] \Rightarrow t_2 : \text{static } [X <: iT_1] \Rightarrow jT_2}$	(T-TABS)
$\frac{\Gamma \vdash t_1 : i([X <: jT_{11}] \Rightarrow kT_{12}) \quad \Gamma \vdash lT_2 <: jT_{11}}{\Gamma \vdash t_1[lT_2] : [X \mapsto lT_2](i \wedge k)T_{12}}$	(T-TAPP)
$\frac{\Gamma \vdash t : iS \quad \Gamma \vdash iS <: jT}{\Gamma \vdash t : jT}$	(T-SUB)

Fig. 3. typing $\Gamma \vdash t : iT$

$\frac{\forall i. i <: \text{dynamic}}{\forall i \in \{\text{static}, \text{inline}\}. i <: \text{static}}$	(IS-DYNAMIC)
$\frac{\forall i \in \{\text{static}, \text{inline}\}. i <: \text{static}}{\text{inline } <: \text{inline}}$	(IS-STATIC)
$\frac{\text{inline } <: \text{inline}}{\text{inline } <: \text{inline}}$	(IS-INLINE)

Fig. 4. Inlinity Subtyping $i <: j$

$$\frac{\forall i, \forall j, i <: j. i \wedge j = j}{\forall i, \forall j, i <: j. i \wedge j = j}$$

Fig. 5. Inlinity Intersection $i \wedge j$

$$\begin{array}{c}
\frac{}{\Gamma \vdash S <: S} \quad \text{(S-REFL)} \\
\frac{\Gamma \vdash S <: U \quad \Gamma \vdash U <: T}{\Gamma \vdash S <: T} \quad \text{(S-TRANS)} \\
\frac{}{\Gamma \vdash S <: \top} \quad \text{(S-TOP)} \\
\frac{X <: T \in \Gamma}{\Gamma \vdash X <: T} \quad \text{(S-TVAR)} \\
\frac{\Gamma \vdash kT_1 <: iS_1 \quad \Gamma \vdash jS_2 <: lT_2}{\Gamma \vdash iS_1 \Rightarrow jS_2 <: kT_1 \Rightarrow lT_2} \quad \text{(S-ARROW)} \\
\frac{\Gamma, X <: iU_1 \vdash jS_2 <: kT_2}{\Gamma \vdash [X <: iU_1] \Rightarrow jS_2 <: [X <: iU_1] \Rightarrow kT_2} \quad \text{(S-ALL)} \\
\frac{\{x_p : i_p T_p^{p \in 1..n+m}\} <: \{x_p : i_p T_p^{p \in 1..n}\}}{\forall p \in 1..n. i_p S_p <: j_p T_p} \quad \text{(S-RECDWIDTH)} \\
\frac{\{x_p : i_p S_p^{p \in 1..n}\} <: \{x_p : j_p T_p^{p \in 1..n}\}}{\{x_p : i_p S_p^{p \in 1..n}\} \text{ is permutation of } \{y_p : j_p T_p^{p \in 1..n}\}} \quad \text{(S-RECDDEPTH)} \\
\frac{\{x_p : i_p S_p^{p \in 1..n}\} \text{ is permutation of } \{y_p : j_p T_p^{p \in 1..n}\}}{\{x_p : i_p S_p^{p \in 1..n}\} <: \{y_p : j_p T_p^{p \in 1..n}\}} \quad \text{(S-RECPERM)} \\
\frac{i <: j \quad \Gamma \vdash S <: T}{\Gamma \vdash iS <: jT} \quad \text{(S-INLINE)}
\end{array}$$

Fig. 6. Subtyping $\Gamma \vdash iS <: jT$

3 Translating Scala to the Core Calculus

4 Case Studies

5 Related Work

6 Conclusion

References

1. Eugene Burmako and Martin Odersky. Scala Macros, a Technical Report. In *Third International Valentin Turchin Workshop on Metacomputation*, 2012.