# Scalar Actions in Lean's Mathlib Based on a paper by Eric Wieser[1]

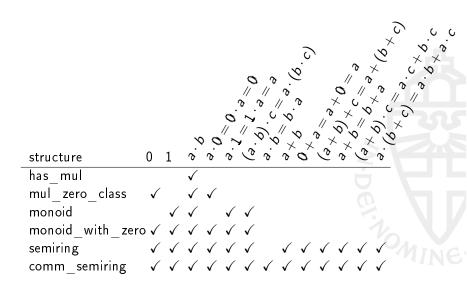
Steven Bronsveld Jelmer Firet

Radboud University Nijmegen

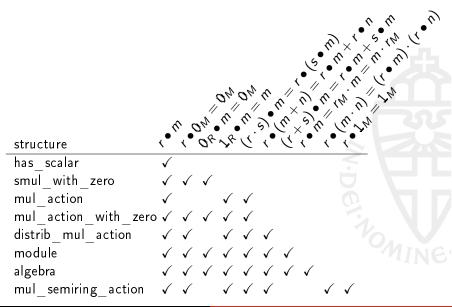
December 19, 2021



# Overview of structures with multiplication



# Overview of structures with scalar multiplication



## has scalar

## Structure (algebra.group.defs)

```
1 instance has_mul.to_has_scalar (\alpha : Type*) [has_mul \alpha]
2 : has_scalar \alpha \alpha := \langle (^*) \rangle
```

## has scalar

## Structure (algebra.group.defs)

```
1 instance has_mul.to_has_scalar (\alpha : Type*) [has_mul \alpha]
2 : has_scalar \alpha \alpha := \{ \text{ smul } := \text{ mul } \}
```

# function.has\_scalar

#### Instance

```
1 instance function.has_scalar (I \alpha : Type*) [has_mul \alpha]
2 : has_scalar \alpha (I \rightarrow \alpha) := { smul := \lambda r v, (\lambda i, r * v i)}
```

What about has\_scalar M (I  $\rightarrow \kappa \rightarrow \alpha$ )?

#### Instance

```
1 instance function.has_scalar (I M \alpha : Type*) [has_scalar M \alpha]
2 : has_scalar M (I \rightarrow \alpha) := { smul := \lambda r v, (\lambda i, r • v i) }
```

What about has\_scalar M ( $\Pi$  i : I, f i)?

```
instance pi.has_scalar (...) [|| i, has_scalar M (f i)|
: has_scalar M (Π i : I, f i) := { smul := λ r v, (λ i, r • v i) }
```

# function.has\_scalar

#### Instance

```
1 instance function.has_scalar (I \alpha : Type*) [has_mul \alpha]
2 : has_scalar \alpha (I \rightarrow \alpha) := { smul := \lambda r v, (\lambda i, r * v i)}
```

What about has\_scalar M (I  $\rightarrow \kappa \rightarrow \alpha$ )?

#### Instance

```
1 instance function.has_scalar (I M \alpha : Type*) [has_scalar M \alpha]
2 : has_scalar M (I \rightarrow \alpha) := { smul := \lambda r v, (\lambda i, r • v i) }
```

What about has\_scalar M ( $\Pi$  i : I, f i)?

```
1 instance pi.has_scalar (...) [\Pi i, has_scalar M (f i)]
2 : has_scalar M (\Pi i : I, f i) := { smul := \lambda r v, (\lambda i, r \cdot v i) }
```

# function.has scalar

#### Instance

```
1 instance function.has_scalar (I \alpha : Type*) [has_mul \alpha]
2 : has_scalar \alpha (I \rightarrow \alpha) := { smul := \lambda r v, (\lambda i, r * v i)}
```

What about has\_scalar M (I  $\rightarrow \kappa \rightarrow \alpha$ )?

#### Instance

```
1 instance function.has_scalar (I M \alpha : Type*) [has_scalar M \alpha]
2 : has_scalar M (I \rightarrow \alpha) := { smul := \lambda r v, (\lambda i, r • v i) }
```

What about has\_scalar M (П i : I, f i)?

```
1 instance pi.has_scalar (...) [\Pi i, has_scalar M (f i)]
2 : has_scalar M (\Pi i : I, f i) := { smul := \lambda r v, (\lambda i, r • v i) }
```

# function.has\_scalar

#### Instance

```
1 instance function.has_scalar (I \alpha : Type*) [has_mul \alpha]
2 : has_scalar \alpha (I \rightarrow \alpha) := { smul := \lambda r v, (\lambda i, r * v i)}
```

What about has\_scalar M (I  $\rightarrow \kappa \rightarrow \alpha$ )?

#### Instance

```
1 instance function.has_scalar (I M \alpha: Type*) [has_scalar M \alpha] 2 : has_scalar M (I \rightarrow \alpha) := { smul := \lambda r v, (\lambda i, r • v i) }
```

What about has\_scalar M ( $\Pi$  i : I, f i)?

```
1 instance pi.has_scalar (...) [\Pi i, has_scalar M (f i)]
2 : has_scalar M (\Pi i : I, f i) := { smul := \lambda r v, (\lambda i, r • v i) }
```

# function.has scalar

#### Instance

```
1 instance function.has_scalar (I \alpha : Type*) [has_mul \alpha]
2 : has_scalar \alpha (I \rightarrow \alpha) := { smul := \lambda r v, (\lambda i, r * v i)}
```

What about has\_scalar M (I  $\rightarrow \kappa \rightarrow \alpha$ )?

#### Instance

```
1 instance function.has_scalar (I M \alpha: Type*) [has_scalar M \alpha]
2 : has_scalar M (I \rightarrow \alpha) := { smul := \lambda r v, (\lambda i, r • v i) }
```

What about has\_scalar M ( $\Pi$  i : I, f i)?

```
1 instance pi.has_scalar (...) [\Pi i, has_scalar M (f i)]
2 : has_scalar M (\Pi i : I, f i) := { smul := \lambda r v, (\lambda i, r • v i) }
```

# Bibliography



Eric Wieser. "Scalar actions in Lean's mathlib". In: arXiv:2108.10700 [cs] (Aug. 10, 2021). arXiv: 2108.10700. URL: http://arxiv.org/abs/2108.10700 (visited on 12/09/2021).