Sikkel: Multimode Simple Type Theory as an Agda Library

Joris Ceulemans Andreas Nuyts Dominique Devriese

KU Leuven, Belgium

28th International Conference on Types for Proofs and Programs
Nantes, France
21 June 2022

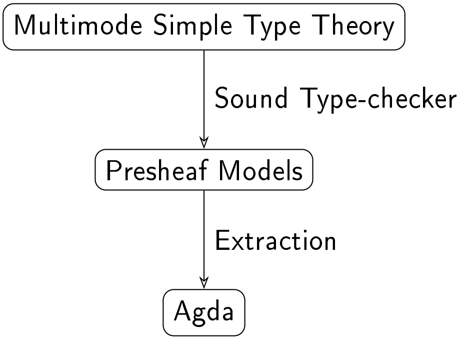
Motivation

Extensions of a standard type theory (MLTT, CIC, ...) with new primitives:

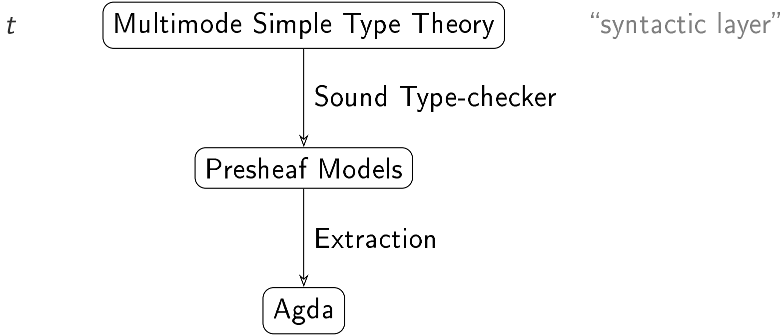
- guarded recursion,
- parametricity,
- univalence,
- directed type theory,
- nominal reasoning,
- ...

How to work in such extended theories within Agda, Coq, ...?

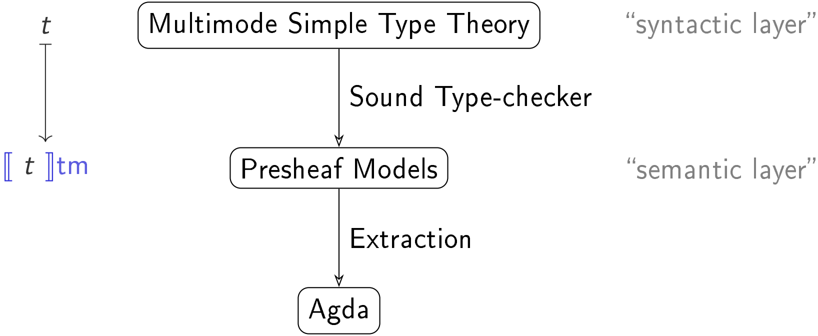
A Brief Overview of Sikkel



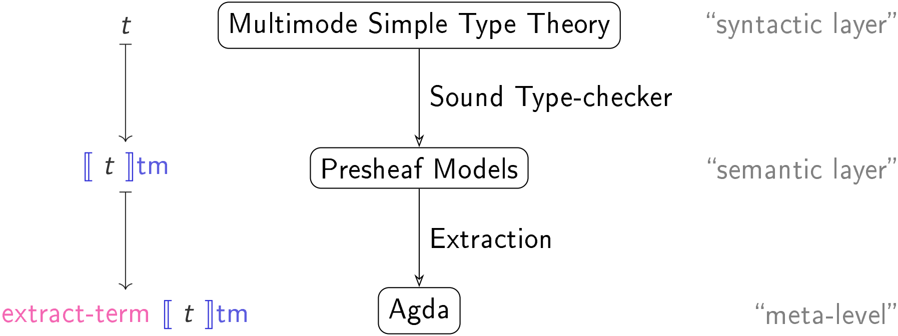
A Brief Overview of Sikkel



A Brief Overview of Sikkel



A Brief Overview of Sikkel



Multimode Simple Type Theory (MSTT)

$\approx$ MTT by Gratzer et al. (LICS20), restricted to simple types.

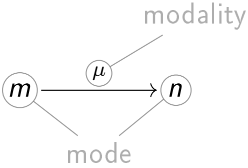
Parametrized by mode theory ($\approx$ small 2-category):

$$m \xrightarrow{\mu} n$$

Multimode Simple Type Theory (MSTT)

$\approx$ MTT by Gratzer et al. (LICS20), restricted to simple types.

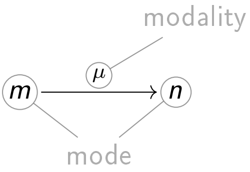
Parametrized by mode theory ($\approx$ small 2-category):



Multimode Simple Type Theory (MSTT)

≈ MTT by Gratzer et al. (LICS20), restricted to simple types.

Parametrized by mode theory (≈ small 2-category):



New primitive modal operations:

$$\begin{array}{ccc} \text{CtxExpr } m & \leftarrow & \text{CtxExpr } n \\ \Gamma \text{ ,lock}\langle\, \mu \,\rangle & \leftarrow\!\shortmid & \Gamma \end{array}$$

$$\begin{array}{ccc} \text{TyExpr } m & \rightarrow & \text{TyExpr } n \\ T & \mapsto & \langle\, \mu \mid T \,\rangle \end{array}$$

Programming in Sikkel

$$m \xrightarrow{\mu} n \xrightarrow{\kappa} o$$

Constructing a function of type

$$\langle\ \kappa \mid \langle\ \mu \mid A\ \rangle \Rrightarrow B\ \rangle \Rrightarrow \langle\ \kappa \circledcirc \mu \mid A\ \rangle \Rrightarrow \langle\ \kappa \mid B\ \rangle$$

```
mod-applicative : TmExpr o
mod-applicative =
    { }0
```

Hole	Mode	Context	Expected type
0	o	$\diamond$	$\langle\ \kappa \mid \langle\ \mu \mid A\ \rangle \Rrightarrow B\ \rangle \Rrightarrow \langle\ \kappa \circledcirc \mu \mid A\ \rangle \Rrightarrow \langle\ \kappa \mid B\ \rangle$

Programming in Sikkel

$$m \xrightarrow{\mu} n \xrightarrow{\kappa} o$$

Constructing a function of type

$$\langle\ \kappa \mid \langle\ \mu \mid A\ \rangle \Rrightarrow B\ \rangle \Rrightarrow \langle\ \kappa \textcircled{m} \mu \mid A\ \rangle \Rrightarrow \langle\ \kappa \mid B\ \rangle$$

```
mod-applicative : TmExpr o
mod-applicative =
    {lam[ κ | "f" ∈ ⟨ μ | A ⟩ ⇛ B ] ?}0
```

Hole	Mode	Context	Expected type
0	o	$\diamond$	$\langle\ \kappa \mid \langle\ \mu \mid A\ \rangle \Rrightarrow B\ \rangle \Rrightarrow \langle\ \kappa \textcircled{m} \mu \mid A\ \rangle \Rrightarrow \langle\ \kappa \mid B\ \rangle$

Programming in Sikkel

$$m \xrightarrow{\mu} n \xrightarrow{\kappa} o$$

Constructing a function of type

$$\langle\ \kappa \mid \langle\ \mu \mid A\ \rangle \Rrightarrow B\ \rangle \Rrightarrow \langle\ \kappa \ \textcircled{m}\ \mu \mid A\ \rangle \Rrightarrow \langle\ \kappa \mid B\ \rangle$$

mod-applicative : TmExpr o
mod-applicative =
    lam[κ | "f" $\in \langle\ \mu \mid A\ \rangle \Rrightarrow B$] { }0

Hole	Mode	Context	Expected type
0	o	$\diamond$, κ | "f" $\in \langle\ \mu \mid A\ \rangle \Rrightarrow B$	$\langle\ \kappa\ \textcircled{m}\ \mu \mid A\ \rangle \Rrightarrow \langle\ \kappa \mid B\ \rangle$

Programming in Sikkel

$$m \xrightarrow{\mu} n \xrightarrow{\kappa} o$$

Constructing a function of type

$$\langle\ \kappa \mid \langle\ \mu \mid A\ \rangle \Rrightarrow B\ \rangle \Rrightarrow \langle\ \kappa \circledcirc \mu \mid A\ \rangle \Rrightarrow \langle\ \kappa \mid B\ \rangle$$

mod-applicative : TmExpr o
mod-applicative =
    lam[κ | "f" $\in \langle\ \mu \mid A\ \rangle \Rrightarrow B$] {lam[κ ⓜ μ | "a" $\in A$]
    ?}0

Hole	Mode	Context	Expected type
0	o	$\diamond\ ,\ \kappa \mid$ "f" $\in \langle\ \mu \mid A\ \rangle \Rrightarrow B$	$\langle\ \kappa \circledcirc \mu \mid A\ \rangle \Rrightarrow \langle\ \kappa \mid B\ \rangle$

Programming in Sikkel

$$m \xrightarrow{\mu} n \xrightarrow{\kappa} o$$

Constructing a function of type

$$\langle\ \kappa \mid \langle\ \mu \mid A\ \rangle \Rrightarrow B\ \rangle \Rrightarrow \langle\ \kappa \ \textcircled{m}\ \mu \mid A\ \rangle \Rrightarrow \langle\ \kappa \mid B\ \rangle$$

```
mod-applicative : TmExpr o
mod-applicative =
    lam[ κ | "f" ∈ ⟨ μ | A ⟩ ⇛ B ] lam[ κ ⓜ μ | "a" ∈ A ]
    { }0
```

Hole	Mode	Context	Expected type
0	o	◇ , κ | "f" ∈ ⟨ μ | A ⟩ ⇛ B , κ ⓜ μ | "a" ∈ A	⟨ κ | B ⟩

Programming in Sikkel

$$m \xrightarrow{\mu} n \xrightarrow{\kappa} o$$

Constructing a function of type

$$\langle\, \kappa \mid \langle\, \mu \mid A \,\rangle \Rrightarrow B \,\rangle \Rrightarrow \langle\, \kappa \circledcirc \mu \mid A \,\rangle \Rrightarrow \langle\, \kappa \mid B \,\rangle$$

```
mod-applicative : TmExpr o
mod-applicative =
    lam[ κ | "f" ∈ ⟨ μ | A ⟩ ⇛ B ] lam[ κ ⓜ μ | "a" ∈ A ]
    {mod⟨ κ ⟩ ?}0
```

Hole	Mode	Context	Expected type
0	o	◇ , κ | "f" ∈ ⟨ μ | A ⟩ ⇛ B , κ ⓜ μ | "a" ∈ A	⟨ κ | B ⟩

Programming in Sikkel

$$m \xrightarrow{\mu} n \xrightarrow{\kappa} o$$

Constructing a function of type

$$\langle\ \kappa \mid \langle\ \mu \mid A\ \rangle \Rrightarrow B\ \rangle \Rrightarrow \langle\ \kappa \ \text{ⓜ}\ \mu \mid A\ \rangle \Rrightarrow \langle\ \kappa \mid B\ \rangle$$

```
mod-applicative : TmExpr o
mod-applicative =
    lam[ κ | "f" ∈ ⟨ μ | A ⟩ ⇛ B ] lam[ κ ⓜ μ | "a" ∈ A ]
    mod⟨ κ ⟩ { }0
```

Hole	Mode	Context	Expected type
0	n	◇ , κ | "f" ∈ ⟨ μ | A ⟩ ⇛ B , κ ⓜ μ | "a" ∈ A ,lock⟨ κ ⟩	B

Programming in Sikkel

$$m \xrightarrow{\mu} n \xrightarrow{\kappa} o$$

Constructing a function of type

$$\langle\, \kappa \mid \langle\, \mu \mid A \,\rangle \Rrightarrow B \,\rangle \Rrightarrow \langle\, \kappa \;ⓜ\; \mu \mid A \,\rangle \Rrightarrow \langle\, \kappa \mid B \,\rangle$$

```
mod-applicative : TmExpr o
mod-applicative =
    lam[ κ | "f" ∈ ⟨ μ | A ⟩ ⇛ B ] lam[ κ ⓜ μ | "a" ∈ A ]
    mod⟨ κ ⟩ {? ·⟨ μ ⟩ ?}0
```

Hole	Mode	Context	Expected type
0	n	◇ , κ | "f" ∈ ⟨ μ | A ⟩ ⇛ B , κ ⓜ μ | "a" ∈ A ,lock⟨ κ ⟩	B

Programming in Sikkel

$$m \xrightarrow{\mu} n \xrightarrow{\kappa} o$$

Constructing a function of type

$$\langle\ \kappa \mid \langle\ \mu \mid A\ \rangle \Rrightarrow B\ \rangle \Rrightarrow \langle\ \kappa \ ⓜ\ \mu \mid A\ \rangle \Rrightarrow \langle\ \kappa \mid B\ \rangle$$

```
mod-applicative : TmExpr o
mod-applicative =
    lam[ κ | "f" ∈ ⟨ μ | A ⟩ ⇛ B ] lam[ κ ⓜ μ | "a" ∈ A ]
    mod⟨ κ ⟩ ({ }0 ·⟨ μ ⟩ { }1)
```

Hole	Mode	Context	Expected type
0	n	◇ , κ | "f" ∈ ⟨ μ | A ⟩ ⇛ B , κ ⓜ μ | "a" ∈ A ,lock⟨ κ ⟩	⟨ μ | A ⟩ ⇛ B
1	m	◇ , κ | "f" ∈ ⟨ μ | A ⟩ ⇛ B , κ ⓜ μ | "a" ∈ A ,lock⟨ κ ⟩ ,lock⟨ μ ⟩	A

Programming in Sikkel

$$m \xrightarrow{\mu} n \xrightarrow{\kappa} o$$

Constructing a function of type

$$\langle\, \kappa \mid \langle\, \mu \mid A \,\rangle \Rrightarrow B \,\rangle \Rrightarrow \langle\, \kappa \; ⓜ \; \mu \mid A \,\rangle \Rrightarrow \langle\, \kappa \mid B \,\rangle$$

```
mod-applicative : TmExpr o
mod-applicative =
    lam[ κ | "f" ∈ ⟨ μ | A ⟩ ⇛ B ] lam[ κ ⓜ μ | "a" ∈ A ]
    mod⟨ κ ⟩ ( {svar "f"}0 ·⟨ μ ⟩ { }1 )
```

Hole	Mode	Context	Expected type
0	n	◇ , κ | "f" ∈ ⟨ μ | A ⟩ ⇛ B , κ ⓜ μ | "a" ∈ A ,lock⟨ κ ⟩	⟨ μ | A ⟩ ⇛ B
1	m	◇ , κ | "f" ∈ ⟨ μ | A ⟩ ⇛ B , κ ⓜ μ | "a" ∈ A ,lock⟨ κ ⟩ ,lock⟨ μ ⟩	A

Programming in Sikkel

$$m \xrightarrow{\mu} n \xrightarrow{\kappa} o$$

Constructing a function of type

$$\langle\, \kappa \mid \langle\, \mu \mid A \,\rangle \Rrightarrow B \,\rangle \Rrightarrow \langle\, \kappa \circledcirc \mu \mid A \,\rangle \Rrightarrow \langle\, \kappa \mid B \,\rangle$$

```
mod-applicative : TmExpr o
mod-applicative =
    lam[ κ | "f" ∈ ⟨ μ | A ⟩ ⇛ B ] lam[ κ ⓜ μ | "a" ∈ A ]
    mod⟨ κ ⟩ (svar "f" ·⟨ μ ⟩ { }1 )
```

Hole	Mode	Context	Expected type
1	m	$\diamond$, κ | "f" $\in \langle\, \mu \mid A \,\rangle \Rrightarrow B$, $\kappa \circledcirc \mu$ | "a" $\in A$,lock$\langle\, \kappa \,\rangle$,lock$\langle\, \mu \,\rangle$	A

Programming in Sikkel

$$m \xrightarrow{\mu} n \xrightarrow{\kappa} o$$

Constructing a function of type

$$\langle \kappa \mid \langle \mu \mid A \rangle \Rrightarrow B \rangle \Rrightarrow \langle \kappa \textcircled{m} \mu \mid A \rangle \Rrightarrow \langle \kappa \mid B \rangle$$

```
mod-applicative : TmExpr o
mod-applicative =
    lam[ κ | "f" ∈ ⟨ μ | A ⟩ ⇛ B ] lam[ κ ⓜ μ | "a" ∈ A ]
    mod⟨ κ ⟩ (svar "f" ·⟨ μ ⟩ {svar "a"}1 )
```

Hole	Mode	Context	Expected type
1	m	◇ , κ | "f" ∈ ⟨ μ | A ⟩ ⇛ B , κ ⓜ μ | "a" ∈ A ,lock⟨ κ ⟩ ,lock⟨ μ ⟩	A

Programming in Sikkel

$$m \xrightarrow{\mu} n \xrightarrow{\kappa} o$$

Constructing a function of type

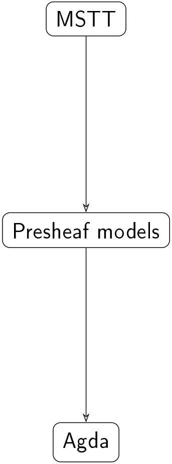
$$\langle\ \kappa \mid \langle\ \mu \mid A\ \rangle \Rrightarrow B\ \rangle \Rrightarrow \langle\ \kappa \textcircled{m}\ \mu \mid A\ \rangle \Rrightarrow \langle\ \kappa \mid B\ \rangle$$

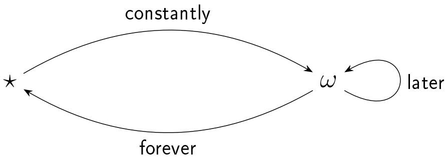
```
mod-applicative : TmExpr o
mod-applicative =
    lam[ κ | "f" ∈ ⟨ μ | A ⟩ ⇛ B ] lam[ κ ⓜ μ | "a" ∈ A ]
    mod⟨ κ ⟩ (svar "f" ·⟨ μ ⟩ svar "a")
```

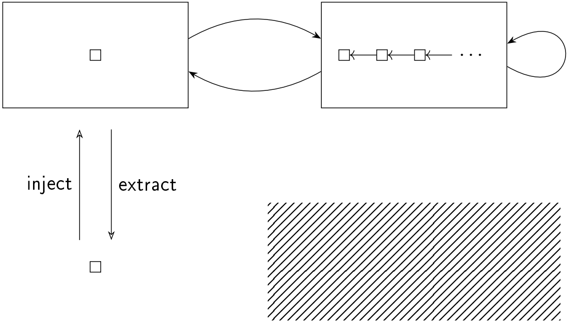
Hole	Mode	Context	Expected type

Presheaf Models & Extraction

Application: Guarded Recursion

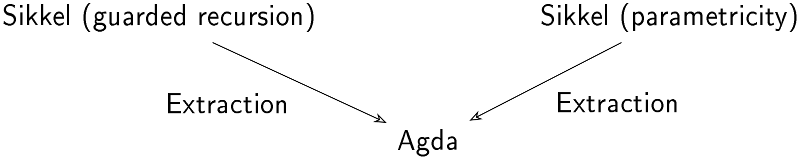






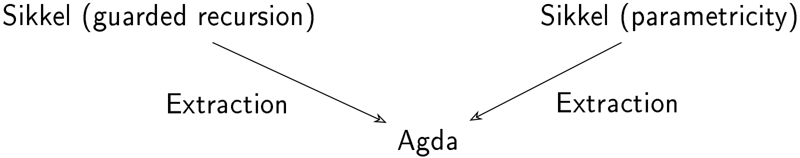
Why Use Sikkel?

- Completely general in mode theory/base category.
- Elegant extraction mechanism via modalities.
- Modular use of type theory extensions, e.g.



Why Use Sikkel?

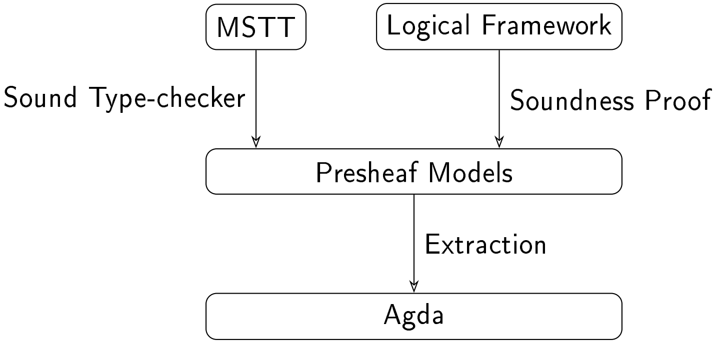
- Completely general in mode theory/base category.
- Elegant extraction mechanism via modalities.
- Modular use of type theory extensions, e.g.



But:

- Custom syntax, so no use of convenient Agda features (e.g. implicits).
- Restricted to MTT-style modal systems, no substructurality.

WIP: Reasoning about Sikkel Programs



Future Work

- Hofmann-Streicher universe at semantic layer.
  - Problems to implement in standard Agda.
  - A variant of Cubical Agda with UIP seems desirable.
- Extension of syntax to full dependent types.
  - Challenging to make interpretation pass termination checker.
- Inductive types (syntactic & semantic layer).
  - E.g. W-types with Sikkel types of constructors and arities.
- Other applications.
  - Ongoing exploration of nominal types.

Thank you for listening!
Questions?

https://github.com/JorisCeulemans/sikkel/releases/tag/v1.0