Type-logical grammar in Agda

Wen Kokke

Utrecht University

August 6th 2015

Take-home message

Machine-checked linguistic theories, which are directly embedded in a published work, are within reach.

(Sometimes a little bit of typesetting is nice, though.)

Example

ex_1 : ✓ mary finds a unicorn
$ex_1 = \_$

ex_2 : ✓ (a unicorn) finds mary
$ex_2 = \_$

ex_3 : * unicorn unicorn unicorn unicorn
$ex_3 = \_$

(The constructors of the parse tree have been omitted, as they are superfluous.)

Example

```
isPreorder : IsPreorder _≡_ _⊢ᴺᴸ_
isPreorder = record
  { isEquivalence = ≡.isEquivalence
  ; reflexive     = ax′
  ; trans         = cut′
  }
```

(I realise this doesn't say much at the moment, but we're getting there.)

Types and judgement

```
data Type : Set where
  el    : Atom → Type
  _⊗_   : Type → Type → Type
  _⍀_   : Type → Type → Type
  _⁄_   : Type → Type → Type

data Judgement : Set where
  _⊢_ : Type → Type → Judgement
```

The non-associative Lambek calculus

```
data NL : Judgement → Set where

  ax    : el A ⊢ el A

  m⊗    : A ⊢ B → C ⊢ D → A ⊗ C ⊢ B ⊗ D
  m⇐    : A ⊢ B → C ⊢ D → B ⇐ C ⊢ A ⇐ D
  m⇒    : A ⊢ B → C ⊢ D → A ⇒ D ⊢ B ⇒ C

  r⇐⊗   : B ⊢ A ⇐ C → A ⊗ B ⊢ C
  r⊗⇐   : A ⊗ B ⊢ C → B ⊢ A ⇐ C
  r⇒⊗   : A ⊢ C ⇒ B → A ⊗ B ⊢ C
  r⊗⇒   : A ⊗ B ⊢ C → A ⊢ C ⇒ B
```

(Each judgement should be prefixed with NL, but in the interest of readability we will use $A \vdash B$ = NL $A \vdash B$.)

Identity expansion

$$
\begin{array}{llll}
\mathsf{ax}' : A \vdash A & & & \\
\mathsf{ax}' \{A = \mathsf{el} & A\} = \mathsf{ax} & & \\
\mathsf{ax}' \{A = A \otimes & B\} = \mathsf{m}\otimes & \mathsf{ax}' \; \mathsf{ax}' & \\
\mathsf{ax}' \{A = A \,/\, & B\} = \mathsf{m}/ & \mathsf{ax}' \; \mathsf{ax}' & \\
\mathsf{ax}' \{A = A \setminus & B\} = \mathsf{m}\setminus & \mathsf{ax}' \; \mathsf{ax}' &
\end{array}
$$

($\{A{=}...\}$ is Agda syntax to match on implicit parameters.)

Cut elimination

- All connectives are introduced by monotonicity rules.
- Each connective can be affected by residuation at the top-level on only *one* side of the turnstile.
- In a cut, we have the top-level connective available on *both* sides.
- So: we can be sure to find the monotonicity rule which introduced the connective in one of the arguments.

Cut elimination

For instance, in the case of $\otimes$:

$$
\dfrac{\dfrac{\dfrac{\dfrac{E \vdash B \qquad F \vdash C}{E \otimes F \vdash B \otimes C}}{\vdots}}{A \vdash B \otimes C} \qquad B \otimes C \vdash D}{A \vdash D}
\quad \rightsquigarrow \quad
\dfrac{\dfrac{\dfrac{\dfrac{E \vdash B \qquad \dfrac{\dfrac{\dfrac{F \vdash C \qquad \dfrac{B \otimes C \vdash D}{C \vdash B \backslash D}}{F \vdash B \backslash D}}{B \otimes F \vdash D}}{B \vdash D \backslash F}}{E \vdash D \backslash F}}{E \otimes F \vdash D}}{\vdots}}{A \vdash D}
$$

Origins

$$\dfrac{\dfrac{\dfrac{E \vdash B \qquad F \vdash C}{E \otimes F \vdash B \otimes C}}{\vdots}}{A \vdash B \otimes C}$$

Origins

$$\dfrac{\dfrac{h_1 : E \vdash B \qquad h_2 : F \vdash C}{\mathsf{m}\otimes\ h_1\ h_2 : E \otimes F \vdash B \otimes C}}{\dfrac{f : \vdots}{f\ (\mathsf{m}\otimes\ h_1\ h_2) : A \vdash B \otimes C}}$$

Origins

$$
\begin{array}{l}
\mathsf{data}\ \ \mathsf{Origin'} \\
\quad (\ f :\ A \vdash B \otimes C\ ) \\
\quad :\ \mathsf{Set}\ \mathsf{where} \\
\\
\quad \mathsf{origin}\ :\ \ (h_1\ :\ E \vdash B) \\
\qquad\qquad \to\ (h_2\ :\ F \vdash C) \\
\qquad\qquad \to\ (f'\ :\ E \otimes F \vdash G \to A \vdash G) \\
\qquad\qquad \to\ (pr\ :\ f \equiv f'\ (\mathsf{m}\otimes\ h_1\ h_2)) \\
\qquad\qquad \to\ \mathsf{Origin'}\ f
\end{array}
$$

(Function f' should work for *any* type G.)

Origins

$$\dfrac{\dfrac{\dfrac{E \vdash B \qquad F \vdash C}{E \otimes F \vdash B \otimes C}}{\vdots}}{A \otimes D \vdash B \otimes C}$$

Origins

$$\cfrac{\cfrac{\cfrac{\cfrac{E \vdash B \qquad F \vdash C}{E \otimes F \vdash B \otimes C}}{\vdots}}{A \vdash (B \otimes C) / D}}{A \otimes D \vdash B \otimes C}$$

Contexts

```
data Polarity : Set where + − : Polarity

data Context (p : Polarity) : Polarity → Set where
  []     : Context p p
  _⊗>_   : Type         → Context p +  → Context p +
  _⇛>_   : Type         → Context p −  → Context p −
  _⇚>_   : Type         → Context p +  → Context p −
  _<⊗_   : Context p +  → Type         → Context p +
  _<⇛_   : Context p +  → Type         → Context p −
  _<⇚_   : Context p −  → Type         → Context p −
```

Contexts

$$
\begin{array}{llll}
\_[\_] : \text{Context}\ p_1\ p_2 \to \text{Type} \to \text{Type} \\
[] & [\,A\,] & = & A \\
B \otimes\!> C & [\,A\,] & = & B \otimes (C\,[\,A\,]) \\
B \setminus\!> C & [\,A\,] & = & B \setminus (C\,[\,A\,]) \\
B /\!> C & [\,A\,] & = & B / (C\,[\,A\,]) \\
C <\!\otimes B & [\,A\,] & = & (C\,[\,A\,]) \otimes B \\
C <\!\setminus B & [\,A\,] & = & (C\,[\,A\,]) \setminus B \\
C <\!/ B & [\,A\,] & = & (C\,[\,A\,]) / B
\end{array}
$$

Contexts

```
data Contextᴶ (p : Polarity) : Set where
  _<⊢_  : Context p +  → Type          → Contextᴶ p
  _⊢>_  : Type         → Context p -   → Contextᴶ p

_[_]ᴶ : Contextᴶ p → Type → Judgement
A <⊢ B [ C ]ᴶ = A [ C ] ⊢ B
A ⊢> B [ C ]ᴶ = A ⊢ B [ C ]
```

Origins (revisited)

```
data Origin
     ( J : Contextᴶ − )
     ( f : NL J [ B ⊗ C ]ᴶ )
     : Set where

     origin  :    (h₁  : E ⊢ B)
             →   (h₂  : F ⊢ C)
             →   (f′  : E ⊗ F ⊢ G → NL J [ G ]ᴶ)
             →   (pr  : f ≡ f′ (m⊗ h₁ h₂))
             →   Origin J f
```

Origins (revisited)

$$
\begin{array}{llll}
\mathsf{view} : (\ J : \mathsf{Context}^{\mathsf{J}}\ -\ )\ (f : \mathsf{NL}\ J\,[\ B \otimes C\ ]^{\mathsf{J}}) \to \mathsf{Origin}\ J\ f \\
\mathsf{view}\ (.\_\ \vdash>\ [])\ (\mathsf{m}\otimes\ f\ g) & = & \mathsf{origin}\ f\ g\ \mathsf{id}\ \mathsf{refl} \\
\mathsf{view}\ (.\_\ \vdash>\ [])\ (\mathsf{r}\diagdown\otimes\ f) & = & \mathsf{wrap}\ \mathsf{r}\diagdown\otimes\ f \\
\mathsf{view}\ (.\_\ \vdash>\ [])\ (\mathsf{r}\diagup\otimes\ f) & = & \mathsf{wrap}\ \mathsf{r}\diagup\otimes\ f \\
 & \vdots &
\end{array}
$$

Origins (revisited)

$$
\begin{aligned}
&\text{view} : (\ J : \text{Context}^{\text{J}}\ -\ )\ (f : \text{NL}\ J\,[\ B \otimes C\ ]^{\text{J}}) \to \text{Origin}\ J\ f\\
&\text{view}\ (.\_\ \vdash>\ [])\ (\text{m}\otimes\ f\ g) \;=\; \text{origin}\ f\ g\ \text{id}\ \text{refl}\\
&\text{view}\ (.\_\ \vdash>\ [])\ (\text{r}\diagdown\otimes\ f) \;=\; \text{wrap}\ \text{r}\diagdown\otimes\ f\\
&\text{view}\ (.\_\ \vdash>\ [])\ (\text{r}\diagup\otimes\ f) \;=\; \text{wrap}\ \text{r}\diagup\otimes\ f\\
&\qquad\vdots
\end{aligned}
$$

$$
\begin{aligned}
&\text{wrap} : (g : \text{NL}\ I\,[\ G\ ]^{\text{J}} \to \text{NL}\ J\,[\ G\ ]^{\text{J}})\ (f : \text{NL}\ I\,[\ B \otimes C\ ]^{\text{J}})\\
&\qquad \to \text{Origin}\ J\ (g\ f)\\
&\text{wrap}\ g\ f\ \text{with}\ \text{view}\ I\ f\\
&\text{wrap}\ g\ f \mid \text{origin}\ h_1\ h_2\ f'\ pr = \text{origin}\ h_1\ h_2\ (g \circ f')\ (\text{cong}\ g\ pr)
\end{aligned}
$$

(A with statement is a way to pattern match on the result of a function.)

Cut elimination (revisited)

$\mathsf{cut}' : A \vdash B \to B \vdash C \to A \vdash C$

$\mathsf{cut}' \; \{B = \mathsf{el} \; \_\} \; f \; g \; \mathsf{with} \; \mathsf{el.view} \; ([] <\vdash \_) \; g$
$\ldots \mid \mathsf{el.origin} \; g' \; \_ = g' \; f$

$\mathsf{cut}' \; \{B = \_ \otimes \_\} \; f \; g \; \mathsf{with} \; \otimes\mathsf{.view} \; (\_ \vdash> []) \; f$
$\ldots \mid \otimes\mathsf{.origin} \; h_1 \; h_2 \; f' \; \_ =$
$\quad f' \; (\mathsf{r}\!\oslash\!\otimes \; (\mathsf{cut}' \; h_1 \; (\mathsf{r}\!\otimes\!\oslash \; (\mathsf{r}\!\varobslash\!\otimes \; (\mathsf{cut}' \; h_2 \; (\mathsf{r}\!\otimes\!\varobslash \; g))))))$

$\vdots$

Cut elimination (revisited)

```
cut′ : A ⊢ B → B ⊢ C → A ⊢ C
cut′ {B = el _} f g with el.view ([] <⊢ _) g
... | el.origin g′ _ = g′ f
cut′ {B = _ ⊗ _} f g with ⊗.view (_ ⊢> []) f
... | ⊗.origin h₁ h₂ f′ _ =
  f′ (r⟋⊗ (cut′ h₁ (r⊗⟋ (r⟍⊗ (cut′ h₂ (r⊗⟍ g))))))
cut′ {B = _ ⟍ _} f g with ⟍.view ([] <⊢ _) g
... | ⟍.origin h₁ h₂ g′ _ =
  g′ (r⊗⟍ (r⟋⊗ (cut′ h₁ (r⊗⟋ (cut′ (r⟍⊗ f) h₂)))))
cut′ {B = _ ⟋ _} f g with ⟋.view ([] <⊢ _) g
... | ⟋.origin h₁ h₂ g′ _ =
  g′ (r⊗⟋ (r⟍⊗ (cut′ h₂ (r⊗⟍ (cut′ (r⟋⊗ f) h₁)))))
```

Utrechts
Universiteitsfonds

Universiteit Utrecht