# System Specification

November 20, 2020

This document is created directly from the definitions in the file `dqtt.ott`, with minor modifications as listed below.

This document is intended to specify, in a readable form, the subject of the proofs in Section 7.2 of the POPL paper "A Graded Dependent Type SYstem with a Usage-Aware Semantics" as well as explain the slight differences between the ott source file (`dqtt.ott`), this rendering, the POPL publication, and the generated Coq files `dqtt_ott.v` and `dqtt_inf.v`.

The reason for these slight differences is due to the restrictions of the Ott locally nameless backend and the LNgen theory generation tool.

1. All parts of the syntax must be defined concretely in the Ott source file.
2. All bound variables need to be explicitly determined
3. All syntactic forms must bind at most one variable at a time.

The first limitation is simple to accommodate through minor manual edits of the outputs of Ott and LNgen. These edits allow us to parameterize the development on an arbitrary semiring (see `usage_sig.v`) instead of working with a specific, concrete semiring.

The second limitation affects our generation of the typing rules for pattern matching elimination forms, i.e. T-UnitE, T-LetBox and T-Case. In these rules, we need to substitute in for the scrutinee $y$ the result type $B$. Therefore, we need to inform Ott that $y$ should be bound in $B$. We do so with an additional annotation (written $@\lambda y.B$) on the corresponding terms. For simplicity, in the paper, we elide this annotation.

The third limitation causes difficulty for the formalization of the elimination rule for tensor products. The usual pattern matching elimination syntactic form binds two variables, one for each component of the tuple. This is the form that is used in the POPL publication. To accommodate Ott, in this version we replace the pattern matching elimination form for $\Sigma$ types with a slightly more general, but less familiar, form.

We call this syntactic form "spread" and it has the following grammar.

$$\textbf{spread}\ a\ \textbf{to}\ x\ \textbf{in}\ b\ @\ B$$

This syntactic form binds the variable $x$ (corresponding to the first component of the product) in the body $b$. The body $b$ must itself be a function, where the

argument is the second component of the tuple. By refactoring in this way, we observe Ott's restriction to single binding.

Using this syntax, we can encode an elimination of an argument $a$ of type $\Sigma x{:}^q A.B$, that uses the usual pattern matching syntax

$$\mathbf{let}\,(x, y)\ =\ (a : \Sigma x{:}^q A.B)\ \mathbf{in}\ b$$

using the term

$$\mathbf{spread}\,a\,\mathbf{to}\,x\,\mathbf{in}\,\lambda y{:}^q\,A.b@\,\lambda y\,^q A.B$$

Even though **spread** solves the issue with single binding in the syntax of the term, there is still the issue of generating its appropriate elimination rule via Ott. Unfortunately, Ott Ott cannot express the correct typing rule for **spread** because the typing rule requires a substitution for both variables. Therefore we modify the generated Coq definition to include the appropriate substitution. For clarity, this document includes the corresponding change in the typeset rule T-Spread.

# 1 Grammar

| | | | |
|---|---|---|---|
| $usage,\ q,\ r,\ s$ | ::= | | |
| $tm,\ a,\ b,\ A,\ B,\ v,\ w$ | ::= | | terms and types |
| | \| | $\mathbf{Unit}$ | |
| | \| | $\mathbf{unit}$ | |
| | \| | $\mathbf{let\,unit}\ =\ a\ \mathbf{in}\ b \mathbin{@} B$ | |
| | \| | $\Pi x :^q A.B$ | |
| | \| | $\lambda x :^q A.a$ | |
| | \| | $a\ b$ | |
| | \| | $\Box_q A$ | |
| | \| | $\mathbf{let\,box}\ x\ =\ a\ \mathbf{in}\ b \mathbin{@} B$ | |
| | \| | $\mathbf{type}$ | |
| | \| | $x$ | |
| | \| | $\mathbf{box}_q\ a$ | |
| | \| | $\mathbf{let}\ x = a\ \mathsf{in}\ b$ | |
| | \| | $A_1 \oplus A_2$ | |
| | \| | $\mathbf{inj}_1\ a$ | |
| | \| | $\mathbf{inj}_2\ a$ | |
| | \| | $\mathbf{case}_q\ a\ \mathbf{of}\ b_1; b_2 \mathbin{@} B$ | |
| | \| | $\Sigma x{:}^q A.B$ | |
| | \| | $(a, b)$ | |
| | \| | $\mathbf{spread}\ a\ \mathbf{to}\ x\ \mathbf{in}\ b \mathbin{@} B$ | |
| $context,\ \Gamma$ | ::= | | contexts |
| | \| | $\varnothing$ | |
| | \| | $x{:}^q A$ | |
| $\Delta$ | ::= | | contexts |
| | \| | $\varnothing$ | |
| | \| | $x{:}A$ | |

# 2 Step relation

$\boxed{a \rightsquigarrow a'}$ *(small-step)*

$$\frac{a \rightsquigarrow a'}{a\ b \rightsquigarrow a'\ b}\ \text{S-AppCong} \qquad \frac{}{(\lambda x :^q A.a)\ b \rightsquigarrow a\{b/x\}}\ \text{S-Beta}$$

$$\frac{a \rightsquigarrow a'}{\mathbf{let\,unit}\ =\ a\ \mathbf{in}\ b \mathbin{@} B \rightsquigarrow \mathbf{let\,unit}\ =\ a'\ \mathbf{in}\ b \mathbin{@} B}\ \text{S-UnitCong}$$

S-UnitBeta

$$\frac{}{\mathbf{let\,unit}\ =\ \mathbf{unit}\ \mathbf{in}\ b \mathbin{@} B \rightsquigarrow b}$$

S-BoxCong

$$\frac{a \rightsquigarrow a'}{\mathbf{let\,box}\, x\ =\ a\ \mathbf{in}\ b \mathbin{@} B \rightsquigarrow \mathbf{let\,box}\, x\ =\ a'\ \mathbf{in}\ b \mathbin{@} B}$$

S-BoxBeta

$$\frac{}{\mathbf{let\,box}\, x\ =\ \mathbf{box}_q\ a\ \mathbf{in}\ b \mathbin{@} B \rightsquigarrow b\{a/x\}}$$

S-CaseCong

$$\frac{a \rightsquigarrow a'}{\mathbf{case}_q\ a\ \mathbf{of}\ b_1; b_2 \mathbin{@} B \rightsquigarrow \mathbf{case}_q\ a'\ \mathbf{of}\ b_1; b_2 \mathbin{@} B}$$

S-Case1Beta

$$\frac{}{\mathbf{case}_q\ (\mathbf{inj}_1\ a)\ \mathbf{of}\ b_1; b_2 \mathbin{@} B \rightsquigarrow b_1\ a}$$

S-Case2Beta

$$\frac{}{\mathbf{case}_q\ (\mathbf{inj}_2\ a)\ \mathbf{of}\ b_1; b_2 \mathbin{@} B \rightsquigarrow b_2\ a}$$

S-SpreadCong

$$\frac{a \rightsquigarrow a'}{\mathbf{spread}\ a\ \mathbf{to}\ x\ \mathbf{in}\ b \mathbin{@} B \rightsquigarrow \mathbf{spread}\ a'\ \mathbf{to}\ x\ \mathbf{in}\ b \mathbin{@} B}$$

S-SpreadBeta

$$\frac{}{\mathbf{spread}\ (a_0, a_1)\ \mathbf{to}\ x\ \mathbf{in}\ b \mathbin{@} B \rightsquigarrow b\{a_0/x\}\ a_1}$$

# 3 Typing relation

$\boxed{\Delta; \Gamma \vdash a : A}$ *(Typing)*

T-sub

$$\frac{\Delta; \Gamma_1 \vdash a : A \qquad \Delta; \Gamma_1 \leq \Gamma_2}{\Delta; \Gamma_2 \vdash a : A}$$

T-type

$$\frac{}{\varnothing; \varnothing \vdash \mathbf{type} : \mathbf{type}}$$

T-var

$$\frac{x \notin \mathsf{dom}\,\Delta \qquad \Delta; \Gamma \vdash A : \mathbf{type}}{\Delta, x{:}A; 0{\cdot}\Gamma, x{:}^1 A \vdash x : A}$$

T-weak

$$\frac{x \notin \mathsf{dom}\,\Delta \qquad \Delta; \Gamma_1 \vdash a : B \qquad \Delta; \Gamma_2 \vdash A : \mathbf{type}}{\Delta, x{:}A; \Gamma_1, x{:}^0 A \vdash a : B}$$

T-Pi

$$\frac{\Delta; \Gamma_1 \vdash A : \mathbf{type} \qquad \Delta, x{:}A; \Gamma_2, x{:}^r A \vdash B : \mathbf{type}}{\Delta; \Gamma_1 + \Gamma_2 \vdash \Pi x{:}^q A.B : \mathbf{type}}$$

T-lam

$$\frac{\Delta, x{:}A; \Gamma_1, x{:}^q A \vdash a : B \qquad \Delta; \Gamma_2 \vdash A : \mathbf{type}}{\Delta; \Gamma_1 \vdash \lambda x{:}^q A.a : \Pi x{:}^q A.B}$$

T-app

$$\frac{\Delta; \Gamma_1 \vdash a : \Pi x{:}^q A.B \qquad \Delta; \Gamma_2 \vdash b : A}{\Delta; \Gamma_1 + q{\cdot}\Gamma_2 \vdash a\ b : B\{b/x\}}$$

T-CONV
$$\frac{\Delta;\Gamma_1 \vdash a : A \quad \Delta;\Gamma_2 \vdash B : \mathbf{type} \quad A \equiv B}{\Delta;\Gamma_1 \vdash a : B}$$

T-UNIT
$$\frac{}{\varnothing;\varnothing \vdash \mathbf{unit} : \mathbf{Unit}}$$

T-UNIT
$$\frac{}{\varnothing;\varnothing \vdash \mathbf{Unit} : \mathbf{type}}$$

T-UNITE
$$\frac{\Delta;\Gamma_1 \vdash a : \mathbf{Unit} \quad B_1 = B\{\mathbf{unit}/y\} \quad \Delta;\Gamma_2 \vdash b : B_1 \quad \Delta, y{:}\mathbf{Unit};\Gamma_3, y{:}^r\mathbf{Unit} \vdash B : \mathbf{type}}{\Delta;\Gamma_1 + \Gamma_2 \vdash \mathbf{let\,unit} \;=\; a \;\mathbf{in}\; b \mathbin{@} (\lambda y{:}^r \mathbf{Unit}.B) : B\{a/y\}}$$

T-BOX
$$\frac{\Delta;\Gamma \vdash A : \mathbf{type}}{\Delta;\Gamma \vdash \Box_q A : \mathbf{type}}$$

T-BOX
$$\frac{\Delta;\Gamma \vdash a : A}{\Delta;\, q{\cdot}\Gamma \vdash \mathbf{box}_q\, a : \Box_q A}$$

T-LETBOX
$$\frac{\Delta;\Gamma_1 \vdash a : \Box_q A \quad \Delta, x{:}A;\Gamma_2, x{:}^q A \vdash b : B\{\mathbf{box}_q\, x/y\} \quad \Delta, y{:}\Box_q A;\Gamma_3, y{:}^r\Box_q A \vdash B : \mathbf{type}}{\Delta;\Gamma_1 + \Gamma_2 \vdash \mathbf{let\,box}\, x \;=\; a \;\mathbf{in}\; b \mathbin{@} (\lambda y{:}^r \Box_q A.B) : B\{a/y\}}$$

T-SUM
$$\frac{\Delta;\Gamma_1 \vdash A_1 : \mathbf{type} \quad \Delta;\Gamma_2 \vdash A_2 : \mathbf{type}}{\Delta;\Gamma_1 + \Gamma_2 \vdash A_1 \oplus A_2 : \mathbf{type}}$$

T-INJ1
$$\frac{\Delta;\Gamma \vdash a : A_1 \quad \Delta;\Gamma_1 \vdash A_2 : \mathbf{type}}{\Delta;\Gamma \vdash \mathbf{inj}_1\, a : A_1 \oplus A_2}$$

T-INJ2
$$\frac{\Delta;\Gamma \vdash a : A_2 \quad \Delta;\Gamma_1 \vdash A_1 : \mathbf{type}}{\Delta;\Gamma \vdash \mathbf{inj}_2\, a : A_1 \oplus A_2}$$

T-CASE
$$\frac{1 \le q \quad \Delta;\Gamma_1 \vdash a : A_1 \oplus A_2 \quad B_1 = B\{\mathbf{inj}_1\, x/y\} \quad B_2 = B\{\mathbf{inj}_2\, x/y\} \quad \Delta;\Gamma_2 \vdash b_1 : \Pi x{:}^q A_1.B_1 \quad \Delta;\Gamma_2 \vdash b_2 : \Pi x{:}^q A_2.B_2 \quad \Delta, y{:}A_1 \oplus A_2;\Gamma_3, y{:}^r A_1 \oplus A_2 \vdash B : \mathbf{type}}{\Delta;\, q{\cdot}\Gamma_1 + \Gamma_2 \vdash \mathbf{case}_q\, a\, \mathbf{of}\; b_1; b_2 \mathbin{@} (\lambda y{:}^r A_1 \oplus A_2.B) : B\{a/y\}}$$

T-Sigma

$$\frac{\Delta;\Gamma_1 \vdash A : \mathbf{type} \qquad \Delta, x{:}A;\Gamma_2, x{:}^r A \vdash B : \mathbf{type}}{\Delta;\Gamma_1 + \Gamma_2 \vdash \Sigma x{:}^q A.B : \mathbf{type}}$$

T-Tensor

$$\frac{\Delta;\Gamma_1 \vdash a : A \qquad \Delta;\Gamma_2 \vdash b : B\{a/x\} \qquad \Delta, x{:}A;\Gamma_3, x{:}^r A \vdash B : \mathbf{type}}{\Delta;\, q\cdot\Gamma_1 + \Gamma_2 \vdash (a, b) : \Sigma x{:}^q A.B}$$

T-Spread

$$\frac{\begin{array}{c} A = \Sigma x{:}^q A_1.A_2 \qquad \Delta;\Gamma_1 \vdash a : A \\ \Delta, x{:}A_1;\Gamma_2, x{:}^q A_1 \vdash b : \Pi y{:}^1 A_2.B\{(x, y)/z\} \qquad \Delta, z{:}A;\Gamma_3, z{:}^r A \vdash B : \mathbf{type} \end{array}}{\Delta;\Gamma_1 + \Gamma_2 \vdash \mathbf{spread}\, a\, \mathbf{to}\, x\, \mathbf{in}\, b : B\{a/z\}}$$