NJPLS @ Princeton University, 10 November 2023

Stratified Type Theory

Jonathan Chan and Stephanie Weirich
University of Pennsylvania

Type Universes

```
Inductive U : Type :=
| mkU : forall X : Type, (X -> U) -> U.
```

Type Universes

```
Inductive U : Type :=
| mkU : forall X : Type, (X -> U) -> U.
```

```
Definition UU : U := mkU U (fun x => x).
```

```
The term "U" has type "Type@{U.u0+1}" while it is expected to have type "Type@{U.u0}"
(universe inconsistency: Cannot enforce U.u0 < U.u0 because U.u0 = U.u0).
```

Universe Hierarchy

```
       j < k
─────────────────────────
⊢ Type@{j} : Type@{k}
```

Universe Hierarchy

$$\frac{j < k}{\vdash \star@\{j\} : \star@\{k\}}$$

Universe Hierarchy

$$\frac{\texttt{j < k}}{\vdash \star\texttt{@\{j\} : }\star\texttt{@\{k\}}}$$

$$\frac{}{\nvdash \star\texttt{@\{k\} : }\star\texttt{@\{k\}}}$$

inconsistent!

Logical Inconsistency

Inductive False : Type.

Definition (consistency): $\nexists$b s.t. $\vdash$ b : False

Girard's paradox: $\vdash \star : \star$ is inconsistent

Universe Level Polymorphism

```
Polymorphic Definition eq@{u v}
  (X : Type@{u}) (x : X) (y : X) : Type@{v} :=
  forall P : X -> Type@{u}, P x -> P y.
(* u v |= u < v *)
```

Levels, Polymorphism, Constraints

ProofView: hurkens.v

```
Error: In environment
p :
P@{hurkens.693 hurkens.693 hurkens.694} U@{hurkens.693 hurkens.689 hurkens.693
  hurkens.690 hurkens.693 hurkens.694 hurkens.693 hurkens.690 hurkens.693
  hurkens.694}
one :
forall
  x : U@{hurkens.678 hurkens.693 hurkens.693 hurkens.690 hurkens.693 hurkens.694
   hurkens.693 hurkens.690 hurkens.693 hurkens.694},
sigma@{hurkens.678 hurkens.693 hurkens.690 hurkens.693 hurkens.694 hurkens.693
  hurkens.689 hurkens.693 hurkens.690 hurkens.693 hurkens.694} x p →
p (fun x0 : Type@{hurkens.689} ⇒ x x0)
The term
 "Omega@{hurkens.689 hurkens.690 hurkens.693 hurkens.692 hurkens.693 hurkens.694
hurkens.695 hurkens.696}"
has type
 "U@{hurkens.692 hurkens.689 hurkens.693 hurkens.690 hurkens.693 hurkens.694
hurkens.692 hurkens.696 hurkens.693 hurkens.694}"
while it is expected to have type
 "U@{hurkens.693 hurkens.689 hurkens.693 hurkens.690 hurkens.693 hurkens.694
hurkens.693 hurkens.690 hurkens.693 hurkens.694}"
(universe inconsistency: Cannot enforce hurkens.696 ≤ hurkens.690 because
hurkens.690 < hurkens.669 < hurkens.696).
```

💡 stratify **judgements** instead of **universes**

Stratifying System F

———

$\boxed{\Gamma \vdash \tau\ \textbf{type}}$ *System F*

$$\frac{\Gamma, \alpha\ \textbf{type} \vdash \tau\ \textbf{type}}{\Gamma \vdash \forall \alpha.\ \tau\ \textbf{type}}$$

$$\frac{\Gamma \vdash \sigma\ \textbf{type} \qquad \Gamma \vdash \tau\ \textbf{type}}{\Gamma \vdash \sigma \to \tau\ \textbf{type}}$$

Stratifying System F

$\boxed{\Gamma \vdash \tau \ \textbf{type}}$ *System F*

$$\frac{\Gamma, \alpha\ \textbf{type} \vdash \tau\ \textbf{type}}{\Gamma \vdash \forall \alpha.\ \tau\ \textbf{type}}$$

$$\frac{\Gamma \vdash \sigma\ \textbf{type} \quad \Gamma \vdash \tau\ \textbf{type}}{\Gamma \vdash \sigma \to \tau\ \textbf{type}}$$

$\boxed{\Gamma \vdash \tau \ \textbf{type}\ @k}$ *Stratified System F*

$$\frac{\Gamma, \alpha\ \textbf{type}\ @j \vdash \tau\ \textbf{type}\ @k \quad j < k}{\Gamma \vdash \forall \alpha\ @j.\ \tau\ \textbf{type}\ @k}$$

$$\frac{\Gamma \vdash \sigma\ \textbf{type}\ @k \quad \Gamma \vdash \tau\ \textbf{type}\ @k}{\Gamma \vdash \sigma \to \tau\ \textbf{type}\ @k}$$

Stratifying Type Theory

$\boxed{\Gamma \vdash a : A}$ *TT*

$$\frac{\Gamma \vdash A : \star@\{k\} \qquad \Gamma, x : A \vdash B : \star@\{k\}}{\Gamma \vdash \Pi x : A.\ B : \star@\{k\}}$$

$$\frac{\Gamma \vdash A : \star@\{k\} \qquad \Gamma \vdash B : \star@\{k\}}{\Gamma \vdash A \to B : \star@\{k\}}$$

$\boxed{\Gamma \vdash a : A\ @k}$ *StraTT*

$$\frac{\Gamma \vdash A : \star\ @j \qquad \Gamma, x : A\ @j \vdash B : \star\ @k \qquad j < k}{\Gamma \vdash \Pi x : A\ @j.\ B : \star\ @k}$$

$$\frac{\Gamma \vdash A : \star\ @k \qquad \Gamma \vdash A : \star\ @k}{\Gamma \vdash A \to B : \star\ @k}$$

Stratifying Type Theory

$\boxed{\Gamma \vdash a : A}$ *TT*

$$\frac{\begin{array}{l}\Gamma \vdash A : \star@\{k\} \\ \Gamma, x : A \vdash B : \star@\{k\}\end{array}}{\Gamma \vdash \Pi x : A.\ B : \star@\{k\}}$$

$$\frac{\Gamma \vdash A : \star@\{k\} \qquad \Gamma \vdash B : \star@\{k\}}{\Gamma \vdash A \to B : \star@\{k\}}$$

$\boxed{\Gamma \vdash a : A\ @k}$ *StraTT*

$$\frac{}{\Gamma \vdash \star : \star\ @k}$$

$$\frac{\begin{array}{l}\Gamma \vdash A : \star\ @j \\ \Gamma, x : A\ @j \vdash B : \star\ @k \quad j < k\end{array}}{\Gamma \vdash \Pi x : A\ @j.\ B : \star\ @k}$$

$$\frac{\Gamma \vdash A : \star\ @k \qquad \Gamma \vdash A : \star\ @k}{\Gamma \vdash A \to B : \star\ @k}$$

Cumulativity

$$\frac{\Gamma \vdash a : A \;@j \qquad j \leq k}{\Gamma \vdash a : A \;@k}$$

Cumulativity

$$\frac{\Gamma \vdash a : A \;@j \qquad j \leq k}{\Gamma \vdash a : A \;@k}$$

$$\frac{j \leq k \qquad \Gamma \vdash f : \Pi x : A \;@i.\; B \;@j}{\Gamma \vdash f : \Pi x : A \;@i.\; B \;@k}$$

Cumulativity

$$\frac{\Gamma \vdash a : A \ @j \quad j \leq k}{\Gamma \vdash a : A \ @k}$$

$$\frac{j \leq k \qquad \Gamma \vdash f : \Pi x : A \ @i.\ B \ @j}{\Gamma \vdash f : \Pi x : A \ @i.\ B \ @k}$$

$$\frac{j \leq k \qquad \Gamma \vdash f : A \to B \ @j}{\Gamma \vdash f : A \to B \ @k}$$

↳ takes A @k, not A @j

Why Floating Functions?

Coq

```
Definition id X (x : X) : X := x.

Definition idid : forall X, X -> X :=
  id (forall X, X -> X) (fun X x => id X x).
```

StraTT

```
id : ΠX : ⋆ @1. X → X @2
id ≔ λX. λx. x

idid : ΠX : ⋆ @0. X → X @2
idid ≔ id (ΠX : ⋆ @0. X → X) (λX. λx. id X x)
```

Why Floating Functions?

Coq

```
Definition id X (x : X) : X := x.

Definition idid : forall X, X -> X :=
  id (forall X, X -> X) (fun X x => id X x).
```

StraTT

$$\mathtt{id} : \Pi X : \star\ @1.\ \underbrace{\Pi x : X\ @1.}_{@1}\ X\ @2$$

$$\mathtt{id} \coloneqq \lambda X.\ \lambda x.\ x$$

$$\mathtt{idid} : \Pi X : \star\ @0.\ \Pi x : X\ @0.\ X\ @2$$

$$\mathtt{idid} \coloneqq \mathtt{id}\ (\Pi X : \star\ @0.\ \Pi x : X\ @0.\ X)\ \overbrace{(\lambda X.\ \lambda x.\ \mathtt{id}\ X\ x)}^{@2}$$

Displacement

$$\frac{i+j \le k \qquad x : A \ @j \coloneqq a \in \Delta}{\Gamma \vdash x^{i} : A+i \ @k} \qquad \frac{x : A \ @j \coloneqq a \in \Delta}{\Delta \vdash x^{i} \rightsquigarrow a+i}$$

$$(\Pi x : A \ @j.\ B)+i = \Pi x : A+i \ @(i+j).\ B+i$$

$$(x^{j})+i = x^{i+j}$$

…

Why Displacement?

```
eq : ΠX : ★ @0. X → X → ★ @1
eq X x y ≔ ΠP : X → ★ @0. P x → P y
```

isSet X ≔

$$\Pi x,y : X\ @0.\ \Pi p,q : \mathrm{eq}\ X\ x\ y\ @1.\ \underbrace{\mathrm{eq}^1\ \overbrace{(\mathrm{eq}\ X\ x\ y)}^{@1}\ p\ q}_{@2}$$

is StraTT useable?

implementation with extensions:

- prototype type checker
  - + datatypes
  - + annotation inference
- small core library

Level + Displacement Annotation Inference

```
unannot. isSet : ΠX : ⋆ @█. ⋆ @█
   StraTT isSet X ≔ Πx,y : X @█. Πp,q : eq X x y @█.
             eq█ (eq X x y) p q

inferred isSet : ΠX : ⋆ @0. ⋆ @2
   StraTT isSet X ≔ Πx,y : X @0. Πp,q : eq X x y @1.
             eq¹ (eq X x y) p q
```

On Expressivity

```
              fixed     floating
data X (a : A @i) (b : B) : ⋆ @j where
  C : … X a b @k
```

On Expressivity

```
              fixed    floating
data X (a : A @i) (b : B) : ⋆ @j where
  C : … X a b @k
```

- cumulativity + displacement help avoid code duplication
  - ↪ except: functions over datatypes need to fix floating parameters

On Expressivity

```
              fixed    floating
data X (a : A @i) (b : B) : ★ @j where
  C : … X a b @k
```

- cumulativity + displacement help avoid code duplication
  - ↪ except: functions over datatypes need to fix floating parameters
- per-defn inference has fewer & more modular levels
  - ↪ but: still a lot of levels per definition

On Expressivity

```
               fixed     floating
data X (a : A @i) (b : B) : ⋆ @j where
  C : … X a b @k
```

- cumulativity + displacement help avoid code duplication
  - ↪ except: functions over datatypes need to fix floating parameters
- per-defn inference has fewer & more modular levels
  - ↪ but: still a lot of levels per definition
- ill-typed defns fail to check at the same points
  - ↪ e.g. all three paradoxes (in 3… 2… 1…)

is StraTT consistent?

we don’t know.

- type-theoretic paradoxes failing
- StraTT w/o floating functions (subStraTT)

Three Type-Theoretic Paradoxes

1. **Russell's paradox**: an inductive containing inductives that don't contain themselves
2. **Burali-Forti's paradox**: a well-founded inductive strictly greater than itself
3. **Hurkens' paradox**: simplification of Girard's paradox

all fail to type check!

- trying to use higher-level term at lower level
- trying to use displaced term at undisplaced type

Consistency of subStraTT

Theorem (logical consistency):
$\nexists$ closed, well-typed inhabitant of $\bot$

Proof sketch:

1. interpret well-typed terms as Agda terms
2. interpret empty type as Agda's empty type
3. show interpretation preserves well-typedness
4. closed subStraTT $\bot$ term implies closed Agda $\bot$ term

Consistency of StraTT?

Theoremn't (logical consistency):

∄ closed, well-typed inhabitant of ⊥

Proof sketch:

1. interpret well-typed terms as Agda terms
2. interpret empty type as Agda's empty type
3. ~~show interpretation preserves well-typedness~~
4. closed subStraTT ⊥ term implies closed Agda ⊥ term

   ↳ floating functions break preserving cumulativity

Model of subStraTT Universes of Types

```
-- universe containing codes of types
data U (k : Level) : Set

-- interpretation of codes of types as Agda types
el : ∀ k → U k → Set

-- codes are cumulative
liftU : ∀ j k → j < k → U j → U k

-- interpretations are cumulative
liftEl : ∀ j k ltjk u → el j u → el k (lift j k ltjk u)
```

Model of StraTT Universes of Types?

```
-- universe containing codes of types
data U (k : Level) : Set

-- interpretation of codes of types as Agda types
el : ∀ k → U k → Set

-- codes are cumulative
liftU : ∀ j k → j < k → U j → U k

-- interpretations are cumulative
liftEl : ∀ j k ltjk u → el j u → el k (lift j k ltjk u)
↳ fails to hold for nondependent function types!
```

Contributions

- StraTT: type theory w/ stratified judgements + displacement
- type safety (not presented)
- prototype implementation
  ↳ datatypes
  ↳ annotation inference
- subStraTT consistency sketch
- consistency evidence

Future Work

- consistency
- expressivity
  ↳ vs. TT + prenex lvl polymorphism
- inference sound-/completeness
- alternative designs
  ↳ only require
  $\Gamma \vdash a : A \ @j \Rightarrow \exists k \geq j.\ \Gamma \vdash A : \star \ @k$
  instead of
  $\Gamma \vdash a : A \ @j \Rightarrow \Gamma \vdash A : \star \ @j$
  ↳ real level polymorphism

thank you!

to find out more:

- preprint
  arxiv.org/abs/2309.12164
- proofs/implementation
  github.com/plclub/StraTT

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

$$\textsc{DT-Type}\ \frac{\Delta \vdash \Gamma}{\Delta; \Gamma \vdash \star :^k \star}$$

$$\textsc{DT-Pi}\ \frac{\Delta; \Gamma \vdash A :^j \star \quad \Delta; \Gamma, x :^j A \vdash B :^k \star \quad j < k}{\Delta; \Gamma \vdash \Pi x :^j A.\, B :^k \star}$$

$$\textsc{DT-Arrow}\ \frac{\Delta; \Gamma \vdash A :^k \star \qquad \Delta; \Gamma \vdash B :^k \star}{\Delta; \Gamma \vdash A \to B :^k \star}$$

$$\textsc{DT-AbsTy}\ \frac{\Delta; \Gamma \vdash A :^j \star \quad \Delta; \Gamma, x :^j A \vdash b :^k B \quad j < k}{\Delta; \Gamma \vdash \lambda x.\, b :^k \Pi x :^j A.\, B}$$

$$\textsc{DT-AppTy}\ \frac{\Delta; \Gamma \vdash b :^k \Pi x :^j A.\, B \quad \Delta; \Gamma \vdash a :^j A \quad j < k}{\Delta; \Gamma \vdash b\ a :^k B\{a/x\}}$$

$$\textsc{DT-AbsTm}\ \frac{\Delta; \Gamma \vdash A :^k \star \quad \Delta; \Gamma \vdash B :^k \star \quad \Delta; \Gamma, x :^k A \vdash b :^k B}{\Delta; \Gamma \vdash \lambda x.\, b :^k A \to B}$$

$$\textsc{DT-AppTm}\ \frac{\Delta; \Gamma \vdash b :^k A \to B \quad \Delta; \Gamma \vdash a :^k A}{\Delta; \Gamma \vdash b\ a :^k B}$$

$$\textsc{DT-Var}\ \frac{x :^j A \in \Gamma \quad \Delta \vdash \Gamma \qquad j \leq k}{\Delta; \Gamma \vdash x :^k A}$$

$$\textsc{DT-Const}\ \frac{x :^j A \coloneqq a \in \Delta \quad \Delta \vdash \Gamma \qquad \vdash \Delta \quad i + j \leq k}{\Delta; \Gamma \vdash x^i :^k A^{+i}}$$

$$\textsc{DT-Bottom}\ \frac{\Delta \vdash \Gamma}{\Delta; \Gamma \vdash \bot :^k \star}$$

$$\frac{\vdash \Delta \qquad \Delta; \varnothing \vdash A :^k \star \quad \Delta; \varnothing \vdash a :^k A \quad x \notin \operatorname{dom} \Delta}{\vdash \Delta, x :^k A \coloneqq a}$$

$$\frac{\Delta \vdash \Gamma \qquad \Delta; \Gamma \vdash A :^k \star \quad x \notin \operatorname{dom} \Gamma \quad x \notin \operatorname{dom} \Delta}{\Delta \vdash \Gamma, x :^k A}$$

$$\textsc{DT-Absurd}\ \frac{\Delta; \Gamma \vdash A :^k \star \quad \Delta; \Gamma \vdash b :^k \bot}{\Delta; \Gamma \vdash \mathsf{absurd}(b) :^k A}$$

$$\textsc{DT-Conv}\ \frac{\Delta; \Gamma \vdash a :^k A \quad \Delta; \Gamma \vdash B :^k \star \quad \Delta \vdash A \equiv B}{\Delta; \Gamma \vdash a :^k B}$$

hurkens.agda

examples > hurkens.agda

```agda
{-# OPTIONS --cumulativity #-}

open import Agda.Primitive

data ⊥ : Set where

U : ∀ ℓ ℓ₁ ℓ₂ → Set (lsuc (ℓ ⊔ ℓ₁ ⊔ ℓ₂))
U ℓ ℓ₁ ℓ₂ = ∀ (X : Set ℓ) → (((X → Set ℓ₁) → Se

τ : ∀ ℓ₁ ℓ₂ → ((U ℓ₁ ℓ₁ ℓ₂ → Set ℓ₁) → Set ℓ₂)
τ ℓ₁ ℓ₂ t = λ X f p → t (λ x → p (f (x X f)))

σ : ∀ ℓ₁ ℓ₂ → U (lsuc (ℓ₁ ⊔ ℓ₂)) ℓ₁ ℓ₂ → (U ℓ₁
σ ℓ₁ ℓ₂ s = s (U ℓ₁ ℓ₁ ℓ₂) (τ ℓ₁ ℓ₂)

Δ : ∀ {ℓ₁ ℓ₂} → U (lsuc (ℓ₁ ⊔ ℓ₂)) ℓ₁ ℓ₂ → Set
Δ {ℓ₁} {ℓ₂} y = (∀ p → σ ℓ₁ ℓ₂ y p → p (τ ℓ₁ ℓ

Ω : ∀ {ℓ} → U ℓ ℓ (lsuc (lsuc ℓ))
Ω {ℓ} = τ ℓ (lsuc (lsuc ℓ)) (λ p → (∀ x → σ ℓ

M : ∀ {ℓ} x → σ (lsuc ℓ) ℓ x (Δ {ℓ} {ℓ}) → Δ {
M {ℓ} _ 𝟚 𝟛 = 𝟛 Δ 𝟚 (λ p → 𝟛 (λ y → p (τ ℓ ℓ (

R : ∀ {ℓ} p → (∀ x → σ ℓ (lsuc (lsuc ℓ)) x p →
R {ℓ} _ 𝟙 = {! 𝟙 (Ω {ℓ}) (λ x → 𝟙 (τ ℓ ℓ (σ ℓ
-- Need Ω : U (lsuc (lsuc (lsuc ℓ))) ℓ (lsuc (
-- Have Ω : U ℓ ℓ (lsuc (lsuc ℓ))

L : ∀ {ℓ} → (∀ p → (∀ x → σ ℓ (lsuc (lsuc ℓ))
L {ℓ} 𝟘 = {! 𝟘 (Δ {ℓ} {ℓ}) M (λ p → 𝟘 (λ y → p
-- Need Δ : U ℓ ℓ (lsuc (lsuc ℓ)) → Set ℓ
-- Have Δ : U (lsuc ℓ) ℓ ℓ → Set (lsuc ℓ)
```

Lean Infoview

```
▼ hurkens.lean:19:55 (pinned)
 ▼ Expected type
  p : U → Type ℓ
  x : U
  ⊢ U
 ▼ Messages (1)
  ▼ hurkens.lean:19:55
  application type mismatch
    p x
  argument
    x
  has type
    U : Type (ℓ + 4)
  but is expected to have type
    U : Type (ℓ + 3)
▼ hurkens.lean:37:14
 ▼ Messages (2)
  ▼ hurkens.lean:37:2
  stuck at solving universe constraint
    ?u.622 =?= max (max (?u.622+1) (?u.625+1)) (?u.638+1)
  while trying to unify
    U.{?u.622, ?u.638, ?u.625, ?u.638,
      ?u.625} : Type (max (max (max (max (?u.622 + 1) (?u.638 + 1)) (?u.625 + 1)) (?u.638 + 1)) (?
  u.625 + 1))
  with
    U.{max (max (?u.625 + 1) (?u.638 + 1)) (?u.622 + 1), ?u.638, ?u.625, ?u.638,
      ?u.625} : Type
      (max
          (max (max (max ((max (max (?u.625 + 1) (?u.638 + 1)) (?u.622 + 1)) + 1) (?u.638 + 1)) (?
  u.625 + 1))
              (?u.638 + 1))
          (?u.625 + 1))
  ▼ hurkens.lean:37:2
  failed to solve universe constraint
    ?u.622 =?= max (max (?u.622+1) (?u.625+1)) (?u.638+1)
  while trying to unify
    U : Type (max (max (max (max (u_1 + 1) (u_2 + 1)) (u_3 + 1)) (u_2 + 1)) (u_3 + 1))
  with
    U : Type
      (max (max (max (max ((max (max (u_3 + 1) (u_2 + 1)) (u_1 + 1)) + 1) (u_2 + 1)) (u_3 + 1)) (u_2
  + 1)) (u_3 + 1))
▼ All Messages (0)
No messages.
```

ProofView: hurkens.v

```
Error: In environment
p :
P@{hurkens.693 hurkens.693 hurkens.694} U@{hurkens.693 hurkens.689 hurkens.693
  hurkens.690 hurkens.693 hurkens.694 hurkens.693 hurkens.690 hurkens.693
  hurkens.694}
one :
forall
  x : U@{hurkens.678 hurkens.693 hurkens.693 hurkens.690 hurkens.693 hurkens.694
   hurkens.693 hurkens.690 hurkens.693 hurkens.694},
sigma@{hurkens.678 hurkens.693 hurkens.690 hurkens.693 hurkens.694 hurkens.693
  hurkens.689 hurkens.693 hurkens.690 hurkens.693 hurkens.694} x p →
p (fun x0 : Type@{hurkens.689} ⇒ x x0)
The term
 "Omega@{hurkens.689 hurkens.690 hurkens.693 hurkens.692 hurkens.693 hurkens.694
hurkens.695 hurkens.696}"
has type
 "U@{hurkens.692 hurkens.689 hurkens.693 hurkens.690 hurkens.693 hurkens.694
hurkens.692 hurkens.696 hurkens.693 hurkens.694}"
while it is expected to have type
 "U@{hurkens.693 hurkens.689 hurkens.693 hurkens.690 hurkens.693 hurkens.694
hurkens.693 hurkens.690 hurkens.693 hurkens.694}"
(universe inconsistency: Cannot enforce hurkens.696 ≤ hurkens.690 because
hurkens.690 < hurkens.669 < hurkens.696).
```

StraTT.pi M ×

pi > StraTT.pi

```
170  tau : P (P UU) → UU
171  tau = \t X f p. t (\s. p (f (s X f)))
172
173  sig : UU → P (P UU)
174  sig = \s. s UU (\t X f p. t (\s. p (f (s X f))))
175
176  Delta : P UU
177  Delta = \y. ((p : P UU @ 1) → sig y p → p (tau (sig y))) → Void
178
179  Omega : UU
180  Omega = tau (\p. (x : UU) → sig x p → p (\X. x X))
181
182  M : (x : UU) → sig x Delta → Delta x
183  M = \x s d. d Delta s (\p. d (\y. p (tau (sig y))))
184
185  D : Type
186  D = (p : P UU) → ((x : UU) → sig x p → p x) → p Omega
```

PROBLEMS 4 OUTPUT PORTS TERMINAL

```
Constraints are
  pi/StraTT.pi:186:5:
  l1251 < l1247
  pi/StraTT.pi:186:19:
  dOmega859 = dUU820
  pi/StraTT.pi:186:19:
  dUU820 + 2 <= l1247
  pi/StraTT.pi:186:20:
  l1272 < l1247
  pi/StraTT.pi:186:32:
  dUU823 = dUU820
  pi/StraTT.pi:186:32:
  dUU820 = dsig825
  pi/StraTT.pi:186:32:
  l1251 <= l1247
  pi/StraTT.pi:186:32:
  dsig825 = dsig825 + 1
  pi/StraTT.pi:186:32:
  l1272 <= l1247
  pi/StraTT.pi:186:32:
  dsig825 + 1 <= l1247
  pi/StraTT.pi:186:25:
  dsig825 + 1 + 1 <= l1247
  pi/StraTT.pi:186:10:
  dsig825 + 1 + 1 <= l1247
  pi/StraTT.pi:186:10:
  dP818 <= l1247
  pi/StraTT.pi:185:5:
  0 <= iST816
  pi/StraTT.pi:185:5:
  l1247 <= iST816
```