Towards Resolving Type Inconsistencies in Transparent Intensional Logic

WG6 meeting in Genova

Samuel Novotný and Ján Perháč

Department of Computers and Informatics
Faculty of Electrical Engineering and Informatics
Technical University of Košice
Slovakia

April 18, 2025

Transparent intensional logic (TIL)

Theoretical foundations

- Author: Pavel Tichy and significant contributors: Pavel Materna, Marie Duzi
- Procedural-semantic **partial hyperintensional** typed λ-calculus
- Syntax – constructions, that is, abstract procedures (computations) constructing the denotation of a linguistic expression.

$$\begin{aligned} C &::= x \mid {}^0X \mid \lambda x \ldots x C \mid [C \ldots C] \mid {}^1C \mid {}^2C \\ X &::= O \mid C \end{aligned} \qquad (G_{\mathrm{TIL}})$$

- Main concept – distinguish between constructions and values constructed by them

Transparent intensional logic

Theoretical foundations

- Two level Type System – Tichý's TT = Church's Simple Type Theory + a modification of Russell's Ramified Type Theory
  - elementary types $B_O = o, \iota, \tau, \omega$, types of constructions $*_1, *_2 \ldots *_n$
  - compound types built on functions over B_O (first order types) and the set of constructions C (higher order types – types of constructions).
- Typing relations

$$\Gamma \mid \Delta \vdash X/\alpha \qquad\qquad \Gamma \mid \Delta \vdash C \to \alpha$$

  - Γ typing context for unbounded variables, Δ typing context for non-constructions

Transparent intensional logic

Typing rules

$$\frac{x \to \alpha \in \Gamma}{\Gamma \mid \Delta \vdash x \to \alpha} \text{ (T-var)} \qquad \frac{\Gamma \mid \Delta \vdash X/\alpha}{\Gamma \mid \Delta \vdash {}^0X \to \alpha} \text{ (T-triv)}$$

$$\frac{\Gamma \mid \Delta \vdash C \to \alpha}{\Gamma \mid \Delta \vdash C/*_n} \text{ (T-prog)} \qquad \frac{X/\alpha \in \Delta}{\Gamma \mid \Delta \vdash X/\alpha} \text{ (T-nconstr)}$$

$$\frac{\begin{array}{c}\Gamma \mid \Delta \vdash C_0 \to (\alpha_0\alpha_1 \ldots \alpha_n) \\ \Gamma \mid \Delta \vdash C_1 \to \alpha_1 \\ \ldots \\ \Gamma \mid \Delta \vdash C_n \to \alpha_n\end{array}}{\Gamma \mid \Delta \vdash [C_0C_1 \ldots C_n] \to \alpha_0} \text{ (T-comp)}$$

$$\frac{\Gamma, x_1 \to \alpha_1, x_2 \to \alpha_2, \ldots, x_n \to \alpha_n \mid \Delta \vdash C \to \alpha_0}{\Gamma \mid \Delta \vdash \lambda x_1 x_2 \ldots x_n C \to (\alpha_0\alpha_1\alpha_2 \ldots \alpha_n)} \text{ (T-clos)}$$

$$\frac{\Gamma \mid \Delta \vdash X \to \alpha}{\Gamma \mid \Delta \vdash {}^1X \to \alpha} \text{ (T-exec)} \qquad \frac{\Gamma \mid \Delta \vdash X \to *_n}{\Gamma \mid \Delta \vdash {}^2X \to \alpha} \text{ (T-2exec)}$$

Transparent intensional logic

Standard application domain

- Logical analysis of natural language.
- The main goal is to express the meaning (Frege's sense) of natural language expressions in mathematically precise terms.
- Three-step analysis:
  1. Type analysis,
  2. Synthesis,
  3. Type checking.

TIL Application Example

Example: Logical analysis of the language expression: *"2+2÷0"*

1. Type analysis
   $\Delta = \{2, 0/\tau;\ +, \div/(\tau\tau\tau)\}$
2. Synthesis
   $[^0+\ {}^02\ [^0\div\ {}^02\ {}^00]] \rightarrow_v$
3. Type checking

$$\dfrac{\dfrac{\dfrac{+/(\tau\tau\tau)\in\Delta}{\Delta\vdash +/(\tau\tau\tau)}}{\Delta\vdash {}^0+\rightarrow(\tau\tau\tau)} \quad \dfrac{\dfrac{2/\tau\in\Delta}{\Delta\vdash 2/\tau}}{\Delta\vdash {}^02\rightarrow\tau} \quad \dfrac{\dfrac{\dfrac{\div/(\tau\tau\tau)\in\Delta}{\Delta\vdash \div/(\tau\tau\tau)}}{\Delta\vdash {}^0\div\rightarrow(\tau\tau\tau)} \quad \dfrac{\dfrac{2/\tau\in\Delta}{\Delta\vdash 2/\tau}}{\Delta\vdash {}^02\rightarrow\tau} \quad \dfrac{\dfrac{0/\tau\in\Delta}{\Delta\vdash 0/\tau}}{\Delta\vdash {}^00\rightarrow\tau}}{\Delta\vdash [^0\div\ {}^02\ {}^00]}}{\Delta\vdash [^0+\ {}^02\ [^0\div\ {}^02\ {}^00]]}$$

Figure: Type Checking Tree

Question

Should improperness always be strictly propagated ↑ ?

TIL Application Example

Example: Logical analysis of the language expression: *"2+2÷0 is undefined"*

1. Type analysis
   $\Delta = \{2, 0/\tau;\ +, \div/(\tau\tau\tau);\ Undefined/(o\tau)\}$
2. Synthesis
   $[^0 Undefined\ [^0 +\ {}^0 2\ [^0 \div\ {}^0 2\ {}^0 0]]] \rightarrow_v$
3. Type checking

$$\dfrac{\dfrac{Undefined/(o\tau) \in \Delta}{\Delta \vdash Undefined/(o\tau)}}{\Delta \vdash {}^0 Undefined \rightarrow (o\tau)} \qquad \dfrac{\vdots}{\Delta \vdash [^0 +\ {}^0 2\ [^0 \div\ {}^0 2\ {}^0 0]}$$
$$\Delta \vdash [^0 Undefined\ [^0 +\ {}^0 2\ [^0 \div\ {}^0 2\ {}^0 0]]]$$

Figure: Type Checking Tree

Observation

That is not correct analysis of this natural language expression.

TIL Application Example

Example: Correct logical analysis of the language expression: *"2+2÷0 is not defined"*

1. Type analysis
   $\Delta = \{2, 0/\tau;\ +, \div/(\tau\tau\tau); Improper/(o{*}_n)\}$
2. Synthesis
   $[^0 Improper\ ^0[^0 +\ \ ^0 2\ [^0 \div\ \ ^0 2\ ^0 0]]] \rightarrow_v T$
3. Type checking

$$\dfrac{\dfrac{\dfrac{Improper/(o{*}_n) \in \Delta}{\Delta \vdash Improper/(o{*}_n)}}{\Delta \vdash {}^0 Improper \rightarrow (o{*}_n)} \qquad \Delta \vdash {}^0[^0 +\ {}^0 2\ [^0 \div\ {}^0 2\ {}^0 0]]}{\Delta \vdash [^0 Improper\ ^0[^0 +\ {}^0 2\ [^0 \div\ {}^0 2\ {}^0 0]]]}$$

Figure: Type Checking Tree

β-reduction

Strategies

- In TIL β-reduction strategy *call by name* does not guarantee **procedural equivalency**, which plays a crucial role in inferring from propositional attitudes (propositions about agent belief, knowledge).

Example

$$[\lambda x\ \lambda y\ [^0+\ x\ y]\ [^0\div\ {}^0\mathbf{2}\ {}^0\mathbf{0}]] \to_v$$

- *improper construction* – it does not construct anything, because there is no value as a result of division 2 by 0
- **but** by its β-reduction we gain *proper construction* constructing a degenerate function f that is not defined on any argument

$$\lambda y\ [^0+\ [^0\div\ {}^0\mathbf{2}\ {}^0\mathbf{0}]\ y] \to_v f$$

Substitution method

Principle

- Function Sub operates on constructions $Sub/(*_n *_n *_n*_n)$ in the following way: It substitutes construction C_2 for construction x in construction $C_1(x)$

$$[\lambda x C_1(x)\ C_2] = {}^2[{}^0Sub\ {}^0C_2\ {}^0x\ {}^0C_1(x)]$$

- Application of Sub function corresponds to meta-programming technics
- Usage: logical analysis of anaphorical expressions in natural language, unbinding of a trivialization-bounded variable and so on

Construction typing problem

Example of construction with type error

$$[\lambda x[^0 + \ x \ ^0 1] \ ^0 John] \rightarrow_v$$

$$\Delta = \{John/\iota, +/(\tau\tau\tau), 1/\tau\}$$

- Type error is detected during type checking

Application of substitution method

$$^2[^0 Sub \ ^{00} John \ ^0 x \ ^0[^0 + \ x \ ^0 1]] \rightarrow_v$$

$$\Delta = \{Sub/(*_n \ *_n \ *_n *_n), John/\iota, +/(\tau\tau\tau), 1/\tau\}$$

- Type error is not detected during type checking

Construction typing problem

Type checking failure

$$\dfrac{\dfrac{\dfrac{Sub/(*_n\ *_n\ *_n *_n) \in \Delta}{\Gamma \vdash Sub/(*_n\ *_n\ *_n *_n)}}{\Gamma \vdash {}^0Sub \to (*_n\ *_n\ *_n *_n)} \quad \dfrac{\dfrac{\dfrac{John/\iota \in \Delta}{\Gamma \vdash John/\iota}}{\Gamma \vdash {}^0John \to \iota}}{\Gamma \vdash {}^0John/*_n} \quad \dfrac{\dfrac{x \to \tau \in \Gamma}{\Gamma \vdash x \to \tau}}{\Gamma \vdash x/*_n} \quad T}{\dfrac{\Gamma \vdash [{}^0Sub\ {}^{00}John\ {}^0x\ {}^0[{}^0+\ x\ {}^01]] \to *_n}{\Gamma \vdash {}^2[{}^0Sub\ {}^{00}John\ {}^0x\ {}^0[{}^0+\ x\ {}^01]] \to \tau}\ \text{(T-2exec)}}$$

Figure: Type Checking Tree

$$\dfrac{\dfrac{\dfrac{+/(\tau\tau\tau) \in \Delta}{\Gamma \vdash +/(\tau\tau\tau)}}{\Gamma \vdash {}^0+ \to (\tau\tau\tau)} \quad \dfrac{x \to \tau \in \Gamma}{\Gamma \vdash x \to \tau} \quad \dfrac{\dfrac{1/\tau \in \Delta}{\Gamma \vdash 1/\tau}}{\Gamma \vdash {}^01 \to \tau}}{\dfrac{\dfrac{\Gamma \vdash [{}^0+\ x\ {}^01] \to \tau}{\Gamma \vdash [{}^0+\ x\ {}^01]/*_n}\ \text{(T-prog)}}{\Gamma \vdash {}^0[{}^0+\ x\ {}^01] \to *_n}}$$

Figure: Type Checking Subtree T

Construction typing problem

The problem

- Loss of the type information about the object constructed by the construction, which was trivialized, caused by the combination of typing rules (T-prog) and (T-triv)

$$\dfrac{\dfrac{\Gamma \mid \Delta \vdash C \to \alpha}{\Gamma \mid \Delta \vdash C/*_n}\,\text{(T-prog)}}{\Gamma \mid \Delta \vdash {}^0C \to *_n}\,\text{(T-triv)}$$

- The need to restore this type information in case of using the double execution construction

$$\dfrac{\Gamma \mid \Delta \vdash C \to *_n}{\Gamma \mid \Delta \vdash {}^2C \to \alpha}\,\text{(T-2exec)}$$

- Problem: types of constructions $*_n$ do not involve types of values constructed by them

Solution

Redefinition of the type of the construction $*_n$

- Inspired by the typing mechanism of references
- Type of the construction will no longer only carry information about its order, but also type information about the object constructed by this construction
- $C/(*_n\alpha)$ means that C is a construction of order n constructing object of type α
- Redefinition of Sub function type $Sub/((*_n\alpha)(*_n\beta)(*_n\beta)(*_n\alpha))$

Redefinition of typing rules

$$\frac{\Gamma \mid \Delta \vdash C \to \alpha}{\Gamma \mid \Delta \vdash C/(\alpha *_n)}\ \text{(T-prog)} \qquad \frac{\Gamma \mid \Delta \vdash X \to (\alpha *_n)}{\Gamma \mid \Delta \vdash {}^2X \to \alpha}\ \text{(T-2exec)}$$

Solution

Problematic construction

$$^{2}[^{0}Sub\ ^{00}John\ ^{0}x\ ^{0}[^{0}+\ \ x\ ^{0}1]] \rightarrow_{v}$$

$$\Delta = \{Sub/((*_{n}\tau)(*_{n}\tau)(*_{n}\tau)(*_{n}\tau)), John/\iota, +/(\tau\tau\tau), 1/\tau\}$$

- Type error is detected during type checking

Solution

Type checking

$$\dfrac{\dfrac{\dfrac{Sub/((*_n\tau)(*_n\tau)(*_n\tau)(*_n\tau)) \in \Delta}{\Gamma \vdash Sub/((*_n\tau)(*_n\tau)(*_n\tau)(*_n\tau))}}{\Gamma \vdash {}^0Sub \rightarrow ((*_n\tau)(*_n\tau)(*_n\tau)(*_n\tau))} \quad \dfrac{\dfrac{Error}{\Gamma \vdash {}^0John/(*_n\tau)}}{\Gamma \vdash {}^{00}John \rightarrow (*_n\tau)} \quad \dfrac{\dfrac{x \rightarrow \tau \in \Gamma}{\Gamma \vdash x \rightarrow \tau}}{\Gamma \vdash x/(*_n\tau)} \quad T}{\dfrac{\Gamma \vdash [{}^0Sub\ {}^{00}John\ {}^0x\ {}^0[{}^0+\ x\ {}^01]] \rightarrow (*_n\tau)}{\Gamma \vdash {}^2[{}^0Sub\ {}^{00}John\ {}^0x\ {}^0[{}^0+\ x\ {}^01]] \rightarrow \tau}\ \text{(T-2exec)}}$$

Figure: Type Checking Process

$$\dfrac{\dfrac{\dfrac{+/(\tau\tau\tau) \in \Delta}{\Gamma \vdash +/(\tau\tau\tau)}}{\Gamma \vdash {}^0+ \rightarrow (\tau\tau\tau)} \quad \dfrac{x \rightarrow \tau \in \Gamma}{\Gamma \vdash x \rightarrow \tau} \quad \dfrac{\dfrac{1/\tau \in \Delta}{\Gamma \vdash 1/\tau}}{\Gamma \vdash {}^01 \rightarrow \tau}}{\dfrac{\dfrac{\Gamma \vdash [{}^0+\ x\ {}^01] \rightarrow \tau}{\Gamma \vdash [{}^0+\ x\ {}^01]/(*_n\tau)}\ \text{(T-prog)}}{\Gamma \vdash {}^0[{}^0+\ x\ {}^01] \rightarrow (*_n\tau)}}$$

Figure: Sub Type Checking Process T

Results

TIL as a functional programming language

- Publications:
  - Towards Resolving Type Inconsistencies in Transparent Intensional Logic;[1]
  - Implementation of TIL-framework in Haskell.[1] – inspired by the redefinition of construction type
  - Another Evidence of Transparent Intensional Logic's Expressive Power in the Field of Temporal Logical Systems[2]

[1] 35th International Conference on Information Modelling and Knowledge Bases EJC 2025, accepted

[2] IEEE 17th International Scientific Conference on Informatics (Informatics) 2024, published

Questions