Logic and **L**ogical **P**hilosophy
Volume 26 (2017), 473–508
DOI: 10.12775/LLP.2017.010

Ivo Pezlar

ALGORITHMIC THEORIES OF PROBLEMS
A constructive and a non-constructive approach

Abstract. In this paper we examine two approaches to the formal treatment of the notion of problem in the paradigm of algorithmic semantics. Namely, we will explore an approach based on Martin-Löf's Constructive Type Theory (CTT), which can be seen as a direct continuation of Kolmogorov's original calculus of problems, and an approach utilizing Tichý's Transparent Intensional Logic (TIL), which can be viewed as a non-constructive attempt of interpreting Kolmogorov's logic of problems. In the last section we propose Kolmogorov and CTT-inspired modifications to TIL-based approach. The focus will be on non-empirical (i.e., mathematical and logical) problems only.

Keywords: logic of problems; algorithmic semantics; procedural semantics; Constructive Type Theory; Transparent Intensional Logic

Introduction

In [17] Kolmogorov proposed his famous explanation of intuitionistic propositions in terms of problems that later became known only as a part of the so-called Brouwer-Heyting-Kolmogorov (BHK) interpretation of intuitionistic logic. More specifically, he proposed to view intuitionistic propositions as problems to which we try to find solutions (proofs, constructions).

To this day, Kolmogorov constructive approach can be still viewed as the prevalent way of logical explication of the notion of problem. This dominance was also fuelled by the discovery of the so-called Curry-Howard-de Bruijn (CHdB) correspondence (see [3, 13, 4], also [42, 9, 41]) between Gentzen's intuitionistic Natural Deduction (ND, [8]) and

Received February 16, 2017. *Revised* April 13, 2017. *Published online* April 28, 2017

Church's simply-typed λ-calculus ([1]) and the cordial endorsement from computer science that followed (see e.g. [35]). In hindsight, it was a natural continuation of Kolmogorov's original line of thought because seeing some proof a as a general method (solution) to a certain problem A is only a short step away from interpreting the method as a *program* or an algorithm (giving answer to the question "How?") and the problem as its *specification* or type (expectation of what it should do, i.e., delivering answer to the question "What?"). This all culminated into Constructive Type Theory (CTT), developed by Martin-Löf (see [20, 21, 19, 22, 23]), which championed interpreted formal syntax and algorithmic semantics (meaning as computation). Thus, from this vantage point, CTT can be considered to be the current leading theory of logical explication of the notion of problem (see e.g. [33, 34, 45, 36, 10], this position was lately also strengthened via Homotopy Type Theory and *The Univalent Foundations Program* [51]).

There is, however, another non-constructivist tradition with different roots that puts forward its own take on interpreted formal syntax and algorithmic semantics and that is Transparent Intensional Logic (TIL) developed by Tichý [49, 50] and based on λ-calculus and ramified many-sorted type theory in the style of Russell and Church (see [55, 2]). TIL also amounted considerable following (see e.g. [7, 5, 14, 37, 39] or [29, 38, 6] in this journal), although not as numerous nor influential as Martin-Löf's CTT. Up until recently TIL, however, offered no explication of the notion of problem that would challenge the status quo of CTT as the dominant explicational framework for non-empirical problems. This gap tried to fill Materna with series of articles [26, 27, 28, 29] and a chapter in a book [25].

Examination and comparison of these two theories will constitute the main subject matter of this paper. More specifically, we will examine two explications of the notion of problem: Martin-Löf's *propositions-as-problems* explication and rivalling Materna's *algorithms-as-problems* explication. The comparison itself will be of practical nature and it will be accomplished via analyses of two case studies, one aimed at mathematical problems, the other focused on logical problems. In other words, we won't be directly comparing CTT and TIL from a formal point of view, rather we will be interested in the overall accuracy of analyses they can provide. The results gained from the comparison will be then utilized together with Kolmogorov's idea of dual system for both problems and proposi-

tions in the final section, where we sketch a modification of TIL-based approach alleviating some of the issues of the original Materna's approach.

Why conduct this comparison if we already have what seems to be a well-developed theory of non-empirical problems at our disposal in the form of CTT? There are two reasons, one theoretical and one practical. The theoretical reason is motivated by the following Materna's question:

> Can we have a procedural theory of problems such that it would not be necessarily connected with intuitionism? [27, p. 297]

and we want to offer our own (positive) answer. The practical reason stems from the fact that while CTT offers extensive framework for dealing with non-empirical problems, it still lacks in terms of analysis of empirical problems, or more generally, analysis of empirical discourse and natural language.[1] Contrary to this, TIL can provide a mature framework for analysing natural language.[2] Our hope is that the interaction between these two theories can enrich them both. First such fusion will be demonstrated in this paper where we emulate in TIL the CHdB correspondence style proof-tracking of ND proofs native to CTT. Thus, although this paper is limited to non-empirical problems, our long-term interest lies in developing a general theory of problems that can encompass both non-empirical and empirical problems. From this perspective, the present paper represents only the first step in that direction.

Structure of the paper. The paper is structured as follows: In the first Section (1), we introduce two problems — one mathematical, one logical — that will serve as a basis for our case studies and consider, informally, what should generally count as a satisfactory solution to non-empirical problems. In the second Section (2), we introduce CTT and try to replicate in its framework as closely as possible the informal analysis from the first section. In the third Section (3), we do the same as in the previous section but from the TIL perspective. In the fourth Section (4), we sketch a modification of TIL inspired by CTT and Kolmogorov.

[1] Some exceptions are e.g. [40, 56]. For more recent developments, see also [43].

[2] Aside from the already mentioned references, see also e.g. [12, 18].

1. Non-Empirical Problems: Preliminary Considerations

1.1. Case Study A: Mathematical Problems

Imagine that you are taking a math test where you are faced with the following single assignment:

(**P0**) Find the root(s) of the equation $x^2 + 3x - 4 = 0$.

Let's imagine further that you simply write down the correct solution, in this case the roots $\langle -4, 1 \rangle$, and hand over the test. Despite giving the correct answer the chances are that examiner would not be very pleased with you. Of course, there is always the possibility that the examiner would trust your mathematical genius and that she would require no further proof of the result and how did you acquire it. It is, however, much more likely that the examiner would also demand from you some sort of mathematical construction that would lead to these two numbers. In other words, she would also want to know how did you get to the solution.

But why is the examiner dissatisfied? After all, we have found the solution, didn't we? Well no, not precisely. What we have found was rather the *result* of the solution, not the solution itself. Recall that in the previous section we have said — in accordance with Kolmogorov — that a *solution* should provide some kind of a method for solving the problem. And it is difficult to see how could a couple of numbers, i.e., $\langle -4, 1 \rangle$, provide any guidance to solving the problem $x^2 + 3x - 4 = 0$.

Of course, the issue here is that the examiner is not really interested in the two numbers themselves, she rather wants to know *how* did we find them. In other words, the examiner wants us to present the proper solution of the problem, not just the result of it. Thus, in order to appease the examiner in the case of problem **(P0)**, we would also have to write down something like:

a.

$$\begin{aligned} & x^2 + 3x - 4 = 0 \\ & (x+4)(x-1) = 0 \\ & x+4 = 0 \text{ or } x-1 = 0 \\ & x = -4 \text{ or } x = 1 \end{aligned}$$

or

b.

$$\begin{aligned} x &= \frac{-3 \pm \sqrt{3^2 - 4 \times (1 \times -4)}}{2 \times 1} \\ &= \frac{-3 \pm \sqrt{9+16}}{2} = \frac{-3 \pm \sqrt{25}}{2} \\ &= \frac{-3 \pm 5}{2} = \frac{-3-5}{2} \text{ or } \frac{-3+5}{2} \\ &= \frac{-8}{2}, \frac{2}{2} = -4 \text{ or } 1 \end{aligned}$$

To frame this issue in slightly different terms, we might say that the examiner wants to see not only the solution-*object*, in this case the couple $\langle -4, 1\rangle$ but also the solution-*process*, in this case represented by **a** or **b** that led to the discovery of this particular solution-object.[3] Note that this distinction is applicable even to the most basic mathematical problems. For example, simple case of addition $5 + 7$ can be framed as the following problem:

(**P1**) Find the natural number x such that $5 + 7 = x$.

The solution-object of **(P1)** would be the number 12 and the solution-process would consist of a series of steps, informally e.g. take 5, take 7 and then add them together.

1.2. Case Study B: Logical problems

Assume similar scenario as in the previous case but now you are given the following assignment:

(**P2**) Find a proof that $A \supset ((A \supset B) \supset B)$ is a theorem.

Notice that this assignment differs from the mathematical problem **(P1)** in one key aspect — you are given the result, in this case that $A \supset ((A \supset B) \supset B)$ is a theorem but you are explicitly asked to present the process that establishes it. In other words, we know from the very beginning that **(P2)** is a theorem but we have to show why it is so. Thus, we have to write down something like:

c.

$$\dfrac{\dfrac{\dfrac{A \supset B \qquad A}{B}}{(A \supset B) \supset B}}{A \supset ((A \supset B) \supset B)}$$

or

d.

$$\dfrac{\dfrac{\dfrac{\dfrac{A \supset B \qquad A}{B}}{A \supset B} \qquad A}{B}}{\dfrac{(A \supset B) \supset B}{A \supset ((A \supset B) \supset B)}}$$

So in the case of logical problem **(P2)** we might say that the examiner, analogously to **(P1)**, is not interested in the result (i.e., acquiring the

[3] We borrow the object/process terminology from [44]. Although it was originally devised for proofs, it is clear that solutions can be thought of as plagued with the same ambiguity.

theorem $A \supset ((A \supset B) \supset B))$ but in the process (i.e. **c** or **d**) that leads to it. This fact is even more evident in the logical case, since we are given the result from the beginning.

1.3. Summary of Preliminary Considerations

Results of our preliminary informal discussion can be summed up as follows: a problem is something *to be done*, a solution is something telling us *how to do it*, i.e., a certain algorithm, method or a set of instructions. We have also encountered a certain ambiguity in the term 'solution': (i.) solution as a process, i.e., an act of determining the solution result, and (ii.) solution as an object, i.e., the result determined by a solution process. Thus, we can informally distinguish solution-*object* (e.g. $\langle -4, 1 \rangle$ in the mathematical case and $A \supset ((A \supset B) \supset B)$ in the logical case) and solution-*process* (e.g. as represented by **a** or **b** in the mathematical case and **c** or **d** in the logical case).

This concludes our introductory examination of the involved concepts. In the upcoming sections, we try to analyse our problems **(P1)** and **(P2)** and their solutions in both senses (i.) and (ii.) in the frameworks of CTT and TIL, respectively.[4]

2. Martin-Löf's Constructive Type Theory

Per Martin-Löf's Constructive Type Theory (see e.g. [20, 21, 22, 23]) is based on the BHK interpretation of logical connectives and to full extent utilizes the CHdB correspondence between propositions (or in our case, problems) and types. The main motivating idea behind CTT is essentially that logic and mathematics are not explicit enough, that they leave too much information outside of their rules. For example, the Implication Introduction Rule of Natural Deduction (ND):

$$\frac{A \vdash B}{A \supset B}$$

keeps hidden away the information about the fact that A, B and $A \supset B$ are all propositions/problems and that we have to assume that A is

[4] Both CTT and TIL will be presented only to the extent that is required by our case studies. In other words, we omit those aspects of the respective theories that will not be necessary for analysing **(P1)** and **(P2)**.

true/solvable (i.e., that we have a proof of it) to derive that B is true, and hence that $A \supset B$ is derivable and true as well.[5] So in its fully exposed version the rule should look something more like this:

$$\frac{A \text{ problem} \quad B \text{ problem} \quad A \supset B \text{ problem} \quad A \text{ solvable} \vdash B \text{ solvable}}{A \supset B \text{ solvable}}$$

and CTT offers a framework that makes these general ideas more precise.

2.1. Problems and Judgements

CTT distinguishes between *problems* and *judgements*. Logical operations (e.g. $\supset, \forall, \ldots$) operate on problems, while logical inferences (e.g. those carried out via $\supset$-Introduction, $\forall$-Introduction,...) operate on judgements.

A problem is defined by laying down what counts as a solution of the problem and we say that problem is solvable if it has a solution. For simplicity, we will consider implication, denoted '$\supset$', to be our only logical connective as our logical problem **(P2)** demands no other. Solution to the problem of the form $A \supset B$ will consist of a method which takes any solution of A into a solution of B (this is where the BHK interpretation comes in). To put it differently, the solution of $A \supset B$ is a function $\lambda x.b(x)$, where $b(a)$ is a solution of B provided a is a solution of A, i.e., it is a function that takes any solution of A as an argument and returns a solution of B (this is where the CHdB correspondence comes in). More specifically, to obtain a solution of B, given solutions a and $\lambda x.b(x)$ of A and $A \supset B$, respectively, all we have to do is apply the latter to the former to get $b(a)$ ($= \beta$-reduction).

There are four basic *categorical* judgements in CTT:

Judgement	*notation*
declaring that A is a problem (proposition, type, ...)	A *prob*
declaring identity of problems A, B	$A = B$
declaring that a is a solution of the problem A	$a : A$
declaring identity of solutions a, b	$a = b : A$

These four categorical judgements can be generalized into *hypothetical* judgements, i.e., judgements depending on a certain assumption. For

[5] We use the symbol '$\vdash$' to separate assumptions from the consequent.

example, the first one $A\ prob$ can be generalized into $x : A \vdash B(x)\ prob$, which says that $B(x)$ is a problem under the assumption that we have a solution x for the problem A, i.e., $x : A$. Hypothetical judgements can be extended to an arbitrary number n of assumptions, e.g. $A\ prob$ can be generalized into $x_1 : A_1, x_2 : A_2(x_1), \ldots, x_n : A_n(x_1, \ldots, x_{n-1}) \vdash A(x_1, \ldots, x_n)\ prob$.[6]

2.2. FIEC Rules

CTT relies on four kinds of rules: *Formation* rules, which tell us how to form new problems, *Introduction* rules, that tell us what are the (canonical) solutions of those problems, *Elimination* rules, which tell us what we can do with solutions, i.e., they show us how to define functions operating on them, and *Computation/Equality* rules, which tell us how functions operate on solutions, i.e., they relate *Intro*-rules and *Elim*-rules.[7]

Given a problem A and a family of problems $B(x)$ over the problem A, we can form the problem $\Pi_{x:A}B(x)$ utilizing the FIEC-style rules in the following way:

$$\Pi\text{-Form}\ \frac{A\ prob \qquad x : A \vdash B(x)\ prob}{\prod\limits_{x:A} B(x)\ prob} \qquad \Pi\text{-Intro}\ \frac{x : A \vdash b(x) : B(x)}{\lambda x.b(x) : \prod\limits_{x:A} B(x)}$$

$$\Pi\text{-Elim}\ \frac{c : \prod\limits_{x:A} B(x) \qquad a : A}{\mathsf{Ap}(c, a) : B(a)} \qquad \Pi\text{-Comp}\ \frac{a : A \qquad x : A \vdash b(x) : B(x)}{\mathsf{Ap}(\lambda x.b(x), a) = b(a) : B(a)}$$

The Π-Form rule specifies the conditions under which we can judge $\Pi_{x:A}B(x)$ to be a problem. The Π-Intro rule states the form of canonical solution to the problem $\Pi_{x:A}B(x)$ and the rules Π-Elim and Π-Comp specify how can we use/compute with these canonical solutions (Ap is a constant for a binary application operation).

With Cartesian product ready, we can define implication as $A \supset B \equiv \Pi_{x:A}B$, where B does not depend on x.[8] This should come as no surprise given our earlier explanation of the connective $\supset$. And with the

[6] For more on CTT, see e.g. [23, 33, 10].

[7] We presuppose that substitution and equality rules are defined in the usual manner, see e.g. [23]. To further save some space, we also omit the identity rules associated with FIEC-rules (see [23]).

[8] The symbol '$\equiv$' represents so-called definitional equality, i.e., equality between linguistic expressions.

dependency gone, we can tailor new rules from the Π-rules specifically for implication (recall that A *solvable* means $a : A$):

$$\frac{A\ prob \quad A\ solvable \vdash B\ prob}{A \supset B\ prob}\ \supset\text{-Form} \qquad \frac{A\ solvable \vdash B\ solvable}{A \supset B\ solvable}\ \supset\text{-Intro}$$

$$\frac{A \supset B\ solvable \quad A\ solvable}{B\ solvable}\ \supset\text{-Elim}$$

Now, we have not only everything necessary for capturing our motivational example from the beginning of this section (specifically, judgements of the form A *prob*, A *solvable*, implication and rules $\supset$-Form, $\supset$-Intro) but also all the essential tools required for the analysis of our case studies A and B.

2.3. CTT-Driven Analysis: Case Study A (*Mathematical Problems*)

Our first goal is to capture the problem:

(**P1**) Find natural number x such that $5 + 7 = x$.

together with its solution in the framework of CTT. Clearly, we are dealing here with natural numbers, so we start by introducing a new type of problems $\mathbb{N}$ *prob* via the already familiar FIEC-style rules:

$$\mathbb{N}\text{-Form}\ \frac{}{\mathbb{N}\ prob} \qquad \mathbb{N}\text{-Intro}\ \frac{}{0 : \mathbb{N}} \qquad \frac{a : \mathbb{N}}{succ(a) : \mathbb{N}}$$

$\mathbb{N}$-Elim

$$\frac{c : \mathbb{N} \qquad d : C(0) \qquad x : \mathbb{N}, y : C(x) \vdash e(x, y) : C(succ(x))}{\mathsf{R}(c, d, (x, y)e(x, y)) : C(c)}$$

$\mathbb{N}$-Comp

$$\frac{d : C(0) \qquad x : A, y : C(x) \vdash e(x, y) : C(succ(x))}{\mathsf{R}(0, d, (x, y)e(x, y)) = d : C(0)}$$

$$\frac{a : \mathbb{N} \qquad d : C(0) \qquad x : \mathbb{N}, y : C(x) \vdash e(x, y) : C(succ(x))}{\mathsf{R}(succ(a), d, (x, y)e(x, y)) = e(a, \mathsf{R}(a, d, (x, y)e(x, y))) : C(succ(a))}$$

Short commentary is in order: the rule $\mathbb{N}$-Form introduces new type of problems $\mathbb{N}$ and the rule $\mathbb{N}$-Intro tells us what are the canonical solutions of these problems, specifically 0 and $succ(a)$. These rules will generate $0 : \mathbb{N}$, $succ(0) : \mathbb{N}$, $succ(succ(0)) : \mathbb{N}$, ..., for which we will use the

notation 0, 1, 2, ... The rule $\mathbb{N}$-Elim defines the function R, which is a primitive recursion/induction over natural numbers. To better explain, first assume that C is a family of problems over $\mathbb{N}$, which we can interpret as some property of natural numbers. Then d is the solution of the base case when c is 0 and e is a solution of the induction step, i.e., a solution from $C(x)$ to $C(succ(x))$. The conclusion then tells us that for any $a : \mathbb{N}$ we have a solution for $C(a)$, which is the $\mathsf{R}(c, d, (x, y)e(x, y))$. Thus, the $\mathbb{N}$-Elim rule essentially gives us tools to define functions on $\mathbb{N}$ by cases. And finally, the rule $\mathbb{N}$-Comp tells us how the function R operates on the canonical solutions introduced by $\mathbb{N}$-Intro, i.e., the first rule is for the case when c is 0, the second one for the case when c is $succ(a)$.

Notice that under this approach the "non-canonical numbers" such as e.g. $1, 2, 3, \ldots$ can be considered to represent the most basic mathematical problems, since they are neither 0, nor have the form $succ(a)$ (see $\mathbb{N}$-Intro rule above). Thus, e.g. 12 represents the problem (e.g. "Find out a construction that constructs number 12.") whose solution is $succ(11)$.[9]

With function R ready, we can now define addition, which is also required for the analysis of our initial problem **(P1)**:

$$a + b \equiv \mathsf{R}(b, a, (x, y)succ(y)) : \mathbb{N}.$$

Intuitively, this definition tells us that to perform the addition $a + b$ is the same as to apply the successor operation b times on a. Consequently, familiar rules can be derived:

$$\frac{a : \mathbb{N} \qquad b : \mathbb{N}}{a + b : \mathbb{N}} \qquad \frac{a : \mathbb{N}}{a + 0 = a : \mathbb{N}} \qquad \frac{a : \mathbb{N} \qquad b : \mathbb{N}}{a + succ(b) = succ(a + b) : \mathbb{N}}$$

Now, we finally approach the analysis of the problem:

(**P1**) Find natural number x such that $5 + 7 = x$.

This problem can be formalized simply as $5+7 : \mathbb{N}$ since we already know that $5 : \mathbb{N}$ and $7 : \mathbb{N}$ (see the first rule for $+$ above). And thus we can conclude that its solution as a process will be $succ(5+6) : \mathbb{N}$ (see the third

[9] The reader might be wondering why don't we evaluate the number 12 fully, i.e., bring it to the form $succ(\ldots succ(0))$ (so-called eager evaluation). We could, of course, do that but there is no need for it, since (by $\mathbb{N}$-Intro) everything in the form $succ(a)$ is already considered to be a canonical solution (so-called lazy evaluation). More on this topic can be found in [10].

rule for + above) with the natural number 12, i.e., $succ(\ldots succ(0))$, as the solution as an object.[10]

2.4. CTT-Driven Analysis: Case Study B (*Logical Problems*)

Our second goal is to capture the problem:

(**P2**) Find a proof that $A \supset ((A \supset B) \supset B)$ is a theorem.

together with its solution. In this case, our task will be much easier: since we are dealing with simple "propositional" problems, we do not need to introduce additional types as in the previous case. The theorem itself can be captured in CTT in the following manner:

$$\prod_{x:A} \prod_{y:\prod_{x:A} B} B$$

and the way we have defined implication allows us to write it simply as: $A \supset ((A \supset B) \supset B)$. Now, we get to the more interesting part, i.e., the analysis of the solutions **c** and **d**:

$$\mathbf{c}'.\quad \dfrac{\dfrac{\dfrac{x : A \supset B \qquad y : A}{\mathsf{Ap}(x,y) : B}\,\Pi\text{-Elim}}{\lambda x.\mathsf{Ap}(x,y) : (A \supset B) \supset B}\,\Pi\text{-Intro}}{\lambda y\lambda x.\mathsf{Ap}(x,y) : A \supset ((A \supset B) \supset B)}\,\Pi\text{-Intro}$$

$$\mathbf{d}'.\quad \dfrac{\dfrac{\dfrac{\dfrac{\dfrac{x : A \supset B \qquad y : A}{\mathsf{Ap}(x,y) : B}\,\Pi\text{-Elim}}{\lambda y.\mathsf{Ap}(x,y) : A \supset B}\,\Pi\text{-Intro} \qquad y : A}{\mathsf{Ap}(\lambda y.\mathsf{Ap}(x,y), y) : B}\,\Pi\text{-Elim}}{\lambda x.\mathsf{Ap}(\lambda y.\mathsf{Ap}(x,y), y) : (A \supset B) \supset B}\,\Pi\text{-Intro}}{\lambda y\lambda x.\mathsf{Ap}(\lambda y.\mathsf{Ap}(x,y), y) : A \supset ((A \supset B) \supset B)}\,\Pi\text{-Intro}$$

Thus, we conclude that the problem $A \supset ((A \supset B) \supset B)$ has a solution $\lambda y.\lambda x.\mathsf{Ap}(x,y)$ which can be understood both as a process (encoding of

[10] Note that $5+7 : \mathbb{N}$ corresponds, strictly speaking, rather to the problem "Find the canonical form of non-canonical natural number $5+7$" than to the original **(P1)**. However, by solving the former problem we get the solution for the latter problem as well since the canonical form $succ(5+6) : \mathbb{N}$ also expresses the desired number 12.

the solution $\mathbf{c}'$) and as an object (λ-term). Note that solutions of $\mathbf{c}'$ and $\mathbf{d}'$ are equivalent.

2.5. Summary of CTT-Driven Analysis

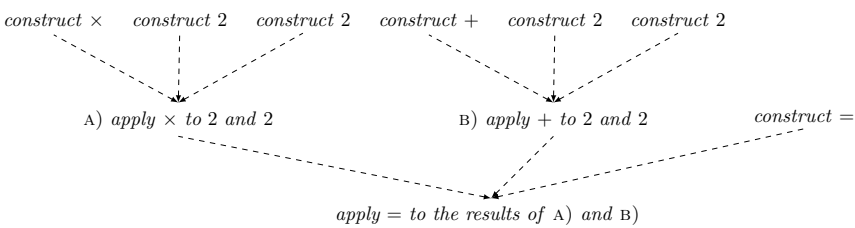
We have shown that CTT can deal with both of our case study problems **(P1)** and **(P2)** without any significant issues. Now, we turn our attention to TIL.

3. Tichý's Transparent Intensional Logic

Pavel Tichý's Transparent Intensional Logic (see e.g. [49, 47, 46]) is based on λ-calculus and ramified many-sorted type theory. Similarly to CTT, the main motivating idea behind TIL is that mathematical languages are, in general, not explicit and rigorous enough. For example, simple equality statement:

$$2 \times 2 = 2 + 2$$

conceals the fact that the number 2 is a certain construction, e.g. two successive applications of the successor function. The similar can be also said about $+$, $\times$, $=$ and about the process of putting them all together. Under this reading, it follows that the statement above expresses equality between results of certain constructions as well. Thus, in the properly "disclosed" version the above statement should look e.g. more like this:



So it is a certain algorithmically structured process that can be read as stating that the results of the two of its subprocedures (displayed as the left and the right branch in the tree-scheme above) are equal. In other words, it tells us that the calculations "multiplying two by two" and "adding two and two together" produce the same result. TIL offers a framework for making these basic insights more precise.

3.1. TIL-Constructions and Matches

In TIL, we will discern between *TIL-constructions*[11] and *matches*, which tell us what is constructed. Logical operations (e.g. $\supset, \forall, \ldots$) appear as functions constructed by corresponding TIL-constructions, while logical inferences (e.g. $\supset$-Introduction, $\forall$-Introduction,...) operate on matches.

There are six fundamental *TIL-constructions*:

TIL-construction	*notation*
Variable	$x, y, \ldots$
Composition	$[K\, L_1 \ldots L_m]$
Closure	$\lambda x_1 \ldots x_m \,.\, K$
Trivialization	$\mathbf{K}, \mathbf{L}, \ldots$
Single execution	$\downarrow K, \downarrow L, \ldots$
Double execution	$\Downarrow K, \Downarrow L, \ldots$

where K and L_i are TIL-constructions. The first four TIL-constructions should be recognizable to anybody familiar with applied/impure λ-calculus: Variables, Compositions, Closures and Trivializations roughly correspond to variables, function applications, function abstractions and constants, respectively. However, it is important to keep in mind that this resemblance is mostly just a cosmetic matter. Although in many contexts they can behave similarly (e.g. both λ-terms and TIL-constructions are amenable to α-, β-, η-conversions), there are non-trivial philosophical as well as technical differences. Most importantly, TIL-constructions are not terms but abstract objects that can be carried out to construct some other objects. This brings us to the last two TIL-constructions, Single and Double execution, which can be thought of as procedures of executing corresponding TIL-constructions once or twice in a row, respectively. What particular object TIL-constructions construct may further depend on a *valuation* v, i.e., an assignment of values to free Variables. If that is the case, we say that they v-construct that object.[12] Furthermore, if a TIL-construction v-constructs nothing at all, we will say that it is

[11] In TIL, by "construction" is meant something different than what intuitionists/constructivists usually do. Most importantly, TIL-constructions are not effective algorithmic computations [47] and partial functions can appear [46].

[12] More precisely, TIL-constructions *always* construct with respect to some valuation v. However, since it is not always the case that the valuation effects the result

a *v-improper* TIL-construction. Otherwise, we say that it is a *v-proper* TIL-construction. If we have two TIL-constructions K, L that both v-construct the same object, or they are both v-improper, we will say that K and L are *v-congruent* TIL-constructions, denoted as $K \cong L$. And if they are v-congruent for all valuations v, we will say that K and L are *equivalent*, denoted as $K \Leftrightarrow L$.

We will symbolize Trivialization by **boldface** font, with the exception of standard logical and mathematical symbols such as '+', '=', '$\forall$', '$\supset$', etc. which will be left in normal font. For example, **1** is a TIL-construction that constructs the number 1. The TIL-construction [**succ 0**] also constructs the number 1 (TIL-constructions **succ** and **0**, intuitively, construct successor function and 0, respectively). TIL-construction [+ **5 7**] constructs the number 12 (where + constructs the addition function) and **[** + **5 7]** constructs [+ **5 7**] (notice the appearance of bold square brackets that represent Trivialization of Composition). Thus, the basic idea behind TIL-constructions is that they are algorithms for computing denotations of the corresponding expressions.[13] Probably the easiest way to digest TIL-constructions is to think of them as explications of Frege's *Sinn*.[14] To cast it in different terms, TIL-constructions are embodying the conception of algorithm-as-proposition.

A problem (TIL-construction) is considered done once we reach a solution. For example, **12** can be considered a basic mathematical problem, whose solution is the number 12. Note that the informal tree-scheme from the beginning of this section can be captured as the following problem: [= [×**2 2**] [+ **2 2**]].

We use the notion of a *match* for stating what objects TIL-constructions v-construct (see [46, 48]). It has the general form $K : k$ where K is any TIL-construction and k is either Variable or Trivialization v-constructing an object of the same type as does K. A match $K : k$ is said to be *satisfied* if K and k v-construct the same object. So e.g. [+ **5 7**] : **12** is a (satisfied) match and it can be read as "TIL-construction [+ **5 7**] constructs number 12" since we already know that **12** constructs number 12. Analogously, [+ **5 7**] : x can be read as "TIL-construction

we will on those occasions speak simply of *constructing* instead of *v-constructing*. For a proper specification of TIL-constructions, see Appendix.

[13] For similar conceptions, see also [31, 32].

[14] Even more accurately, they try to explicate Frege's notion of *conceptual content* from *Begriffschrift* [53].

[+ **5 7**] v-constructs some number x", etc. Note that match essentially states a special case of v-congruence between K and k, where k is either Variable or Trivialization. Thus e.g. [+ **5 7**] : **12** is a match, however [+ **5 7**] : [+ **6 6**] is not a match — even though both TIL-construction are congruent, the TIL-construction on the right-hand side of : is neither Variable nor Trivialization. With the concept of match at hand, we can now represent assertions in TIL. For example, if we want to state that some propositional TIL-construction/problem K constructs/resolves to the truth value **true**, we can simply write K : **true**, which can be also read as "the proposition constructed by K is true".[15]

TIL relies on Church-like type theory. All entities including TIL-constructions receive a type and if α and $\beta_1 \dots \beta_m$ are types, then $(\alpha\beta_1 \dots \beta_m)$ is also a type. Specifically, a type of function from the elements of type $\beta_1 \dots \beta_m$ to the elements of type α. We will introduce three base types o, ι, ν for truth values, individuals and natural numbers, respectively. That way we can construct objects (e.g. via Trivializations or Variables) of the following types:[16]

TIL-construction	*type of constructed object*	*object description*
true, **false**	o	truth values
A	o	non-empirical proposition
$\wedge$	(ooo)	conjunction
$\supset$	(ooo)	implication
$+$	$(\nu\nu\nu)$	addition
$=_\alpha$	$(o\alpha\alpha)$	equality
$\exists_\alpha$	$(o(o\alpha))$	existential quantifier
$\forall_\alpha$	$(o(o\alpha))$	universal quantifier
$\imath_\alpha$	$(\alpha(o\alpha))$	definite description

[15] Materna does not employ Tichý's concept of match in his TIL-based theory of problems, nor any deduction system for that matter, however they can be both incorporated without changing any fundamental principles of his theory. We do so to allow for a smoother analysis of our case studies.

[16] We diverge here from the traditional TIL nomenclature where a proposition is considered a function from possible worlds and time moments to truth values, not just something that can be true or false.

Note that we regard here the functions constructed by $=_\alpha$, $\exists_\alpha$, $\forall_\alpha$ and $\imath_\alpha$ as polymorphic functions. Thus, e.g. $=_\nu$ constructs equality function of type $(o\nu\nu)$ between natural numbers, $=_\iota$ constructs equality function of type $(o\iota\iota)$ between individuals, etc. The $\forall_\alpha$ constructs a function that takes class of objects of type α and returns true if that class contains all elements of that type, otherwise it returns false. Analogously for $\exists_\alpha$. The definite description $\imath_\alpha$ constructs a function that takes class of objects of type α and returns the only element of that class if it is a singleton, otherwise it is undefined.

Next, we introduce type $*_1$ for so-called 1st-order TIL-constructions, i.e., TIL-constructions constructing non-constructions. Then we have type $*_2$ for 2nd-order TIL-constructions constructing 1st-order TIL-constructions and so on (this is where the ramified type theory comes into play). More properly defined (see [49] for the original formulation): let B be a collection of base types o, ι, ν.

1. (a) Every member of B is a *type of order* 1 *over* B.
   (b) If $0 < m$ and $\alpha, \beta_1 \dots \beta_m$ are *types of order* 1 *over* B then the collection $(\alpha\beta_1 \dots \beta_m)$ of all m-ary (total and partial) functions from $\beta_1 \dots \beta_m$ into α is also a *type of order* 1 *over* B.
   (c) Nothing is a *type of order* 1 *over* B unless it follows from (a) and (b).
2. (d) Let χ be any *type of order* n *over* B. Every Variable ranging over χ is a *TIL-construction of order* n *over* B. If K is of (i.e., belongs to) type χ then $\mathbf{K}$, $\downarrow K$, $\Downarrow K$ are *TIL-construction of order* n *over* B. Every Variable ranging over χ is a *TIL-construction of order* n *over* B.
   (e) If $0 < m$ and $K, L_1, \dots, L_m$ are *TIL-constructions of order* n then $[K\ L_1 \dots L_m]$ is a *TIL-construction of order* n *over* B.
   (f) If $0 < m$, χ is a *type of order* n *over* B and K as well as the distinct Variables $x_1, \dots, x_m$ are *TIL-constructions of order* n *over* B, then $\lambda_\chi x_1 \dots x_m.K$ is a *TIL-construction of order* n *over* B.
   (g) Nothing is a *TIL-construction of order* n *over* B unless it so follows from (d), (e) and (f).

   Let $*_n$ be the collection of *TIL-constructions of order* n *over* B. The collection of *types of order* $n+1$ *over* B is defined as follows:

3. (h) $*_n$ and every *type of order* n is a *type of order* $n+1$.
   (i) If $0 < m$ and $\alpha, \beta_1, \dots, \beta_m$ are *types of order* $n+1$ *over* B then the collection $(\alpha\beta_1 \dots \beta_m)$ of all m-ary (total and partial) functions from $\beta_1, \dots, \beta$ into α is also a *type of order* $n+1$ *over* B.
   (j) Nothing is a *type of order* $n+1$ *over* B unless it follows from (h) and (i).

For example, **12** is a TIL-construction of order 1, written $*_1$, constructing the number 12, i.e., an object of type ν. Same goes for [+ **5 7**]. But note that **[** + **5 7]** is already a TIL-construction of order 2, written $*_2$, and it constructs other lower-order TIL-construction, in this case the TIL-construction [+ **5 7**] of type $*_1$. TIL-construction ⇓**[** + **5 7]** would be then of order 3 and it would construct 12 again.[17]

We will write K/α to indicate that a TIL-construction K is of type α (see the left column below) and we write $K \rhd \alpha$ to say that a TIL-construction K (v-)constructs object of type α (see the right column below), e.g.

5 / $*_1$	**5** $\rhd \nu$
[+ **5 7**] / $*_1$	[+ **5 7**] $\rhd \nu$
[+ **5 7]** / $*_2$	**[** + **5 7]** $\rhd *_1$
⇓**[** + **5 7]** / $*_3$	⇓**[** + **5 7]** $\rhd \nu$
[$=_\nu$ [× **2 2**] [+ **2 2**]] / $*_1$	[$=_\nu$ [× **2 2**] [+ **2 2**]] $\rhd o$
[¬ [$=_{*_1}$ **[** × **2 2] [** + **2 2]**]] / $*_2$	[¬ [$=_{*_1}$ **[** × **2 2] [** + **2 2]**]] $\rhd o$

Note that / and $\rhd$ are used here as metalanguage symbols.

3.2. TIL Inference Rules and Conceptual Systems

Due to the "stipulation-based" nature of (Materna's) TIL,[18] we will introduce only two inference rules as necessitated by our second case study, namely Implication Introduction rule $\supset$**I** and Implication Elimination rule $\supset$**E** in the traditional ND-style:

$$\frac{\begin{array}{c}(A : \textbf{true})\\ \vdots \\ B : \textbf{true}\end{array}}{[\supset A\ B] : \textbf{true}}\ \supset\textbf{I} \qquad \frac{[\supset A\ B] : \textbf{true} \qquad A : \textbf{true}}{B : \textbf{true}}\ \supset\textbf{E}$$

[17] For more on TIL's ramified type theory, see [49].

[18] Contrary to CTT, Materna's system does not rest on explicit rules, rather on stipulations of kinds of TIL-constructions, which roughly correspond to the specification of well-formed terms (see Section 3.1). The same holds also for TIL as presented in [7] (the lack of rules was later amended to some extent by [5]). This is, however, not the case for all incarnations of TIL, for "TIL with rules", see e.g. [46, 48] (earlier versions of TIL without ramified hierarchy of types) or [39] (with ramified hierarchy).

Finally, we will use Materna's so-called *conceptual systems*,[19] which, roughly put, delimit the "area of operation" of TIL. Conceptual system is essentially a tuple

$$\langle Pr, Type, Var, C, Der\rangle$$

where Pr is finite class of primitive concepts $P_1, \ldots, P_k$, i.e., basic objects of the system, *Type* is an infinite class of types generated over a finite base (e.g. o, ι, ν), *Var* is an infinite set of variables, countably infinite for each type from *Type*, C is the definition of kinds of TIL-constructions (recall Section 3.1) and *Der* is an infinite class of compound concepts derived from Pr and *Var* utilizing C (see also Definition 2.14 in [7]). There are generally no restrictions as to what types, concepts and TIL-constructions we can choose to build our conceptual systems from. The use of conceptual systems is straightforward and will be obvious once we get to a more concrete example below.

Now, we have all the essential tools required for analysing our case studies A and B. But before we start it is important to reiterate that the explication of problems in the framework of TIL was undertaken by Materna. Thus, strictly speaking, we will be comparing Martin-Löf's CTT and Materna's take on TIL, denoted by TIL_M, rather than Tichý's TIL itself, as originally presented in [49]. There are not many crucial differences but it is beneficial to keep them apart.

3.3. TIL_M-Driven Analysis: Case Study A (*Mathematical Problems*)

Analogously to the previous section dedicated to CTT, our goal will be to formalize the problem:

(**P1**) Find natural number x such that $5 + 7 = x$.

and its solution in the framework of TIL_M. To that purpose we utilize a conceptual system "hand-crafted" specifically for our case study such that $Pr = \{\mathbf{0}, \mathbf{succ}, \wedge, =_\nu, \forall_\nu, \imath_{(\nu\nu\nu)}\}$ and *Type* will be generated over the base $\{o, \nu\}$. In other words, the primitive concepts of our toy conceptual system are: (TIL-construction of) the number zero, successor function, conjunction, equality between natural numbers, universal quantifier for natural numbers and definite description for binary functions on natural numbers, i.e., a function of type $((\nu\nu\nu)(o(\nu\nu\nu)))$, respectively (recall Section 3.1).

[19] We rely here on conceptual systems as presented in [7]. For earlier versions, see [24, 25]. Their critique can be found in [38].

In such conceptual system, assuming natural numbers were defined in a standard way via zero and the successor function, we can define new compound concept of addition operation + of type $(\nu\nu\nu)$ in the following way (see e.g. [7, p. 167]):

$$+ =_{\text{def.}} \Big[\imath_{(\nu\nu\nu)}\, \lambda f[\forall_\nu\, \lambda xy\, [\wedge[=_\nu\, [f\ x\ \mathbf{0}]\ x]\ [=_\nu\, [f\ x\ [\mathbf{succ}\ y]]\, [\mathbf{succ}\ [f\ x\ y]]]\Big].$$

Now, we have everything we need for the analysis of the problem **(P1)**. It can be formalized simply as [+ **5 7**], i.e., as a Composition that applies the addition operation (see above) to the arguments 5 and 7. Such Composition constructs (solution as a process) the natural number 12, which constitutes the solution (solution as an object) to the problem **(P1)**.

3.4. TIL_M-Driven Analysis: Case Study B (*Logical Problems*)

Our next goal is to formalize the problem:

(**P2**) Find a proof that $A \supset ((A \supset B) \supset B)$ is a theorem.

and its solution. For simplicity, we skip the specification of the corresponding conceptual system and head straight for the analysis, since the needed inference rules, i.e., $\supset$**I** and $\supset$**E**, have been already introduced.

The theorem itself can be captured in TIL_M in the following manner:

$$[\supset A\, [\supset\ [\supset A\ B]\ B]] : \mathbf{true}.$$

Solution wise we get:

$$\mathbf{c}''.\quad \dfrac{\dfrac{\dfrac{([\supset A\ B] : \mathbf{true}) \qquad (A : \mathbf{true})}{B : \mathbf{true}}\supset\mathbf{E}}{[\supset\ [\supset A\ B]\ B] : \mathbf{true}}\supset\mathbf{I}}{[\supset A\, [\supset\ [\supset A\ B]\ B]] : \mathbf{true}}\supset\mathbf{I}$$

$$\mathbf{d}''.\quad \dfrac{\dfrac{\dfrac{\dfrac{([\supset A\ B] : \mathbf{true}) \qquad (A : \mathbf{true})}{B : \mathbf{true}}\supset\mathbf{E}}{[\supset A\ B] : \mathbf{true}}\supset\mathbf{I} \qquad (A : \mathbf{true})}{B : \mathbf{true}}\supset\mathbf{E}}{[\supset\ [\supset A\ B]\ B] : \mathbf{true}}\supset\mathbf{I}}{[\supset A\, [\supset\ [\supset A\ B]\ B]] : \mathbf{true}}\supset\mathbf{I}$$

At first glimpse, everything might seem fine but the problematic part is that in Materna's framework the solutions to the logical problem [$\supset$

$A\ [\supset\ [\supset\ A\ B]\ B]]$ are not the respective proofs $\mathbf{c}''$ and $\mathbf{d}''$ but the truth value true.

Materna states that "... [problem]... $[\supset\ [\wedge\ A\ B]\ A]$ [has the solution] true..." [25, p. 100]. Of course, from a certain point of view it sounds reasonable. After all, $[\supset\ A\ [\supset\ [\supset\ A\ B]\ B]]$, as well as $[\supset\ [\wedge\ A\ B]\ A]$ in the quotation above, is indeed a TIL-construction of a theorem, so it seems only natural that it would lead into the truth value true. However, it is not immediately obvious in what sense can be a mere truth value considered a solution. Remember that at the beginning of this paper we have said that solution should be something that tells us *how* to solve a problem and it is difficult to see how can a truth value accomplish such a job. In other words, the solution true tells us nothing about how to actually solve the corresponding problem. If the examiner asks us to prove that $[\supset\ A\ [\supset\ [\supset\ A\ B]\ B]]$ is a theorem and we give back the truth value true ("Yes, it is a theorem"), the examiner would probably not consider the problem solved. Thus, we effectively end up with solutions that do not solve anything.[20]

But there is another possibly even more problematic issue that might not be apparent in our analysis since both of the proofs $\mathbf{c}''$ and $\mathbf{d}''$ are essentially the same ($\mathbf{d}''$ normalizes into $\mathbf{c}''$). However, if we consider different problems, and consequently different solutions, the issue will become immediately obvious. Specifically, on Materna's approach all logical theorems share the same solution and thus they are all indistinguishable from each other. To put it differently, there is only one "big solution" true for all logical problems. Such conception of solutions is not very informative, especially when compared directly with CTT.

[20] Later, Materna proposes a more refined account of solutions of logical problems that replaces the initial one, which we have discussed above. However, the same criticism still applies. More specifically, the preliminary solution "true" in case of $[\supset\ [\wedge\ A\ B]\ A]$ is replaced by "The result of applying a function to the function given by the given GNPI" [25, p. 103], where GNPI are so-called *general problems*, i.e., problems that contain λ-bound Variables and construct some function. But if we put it all together, this leads to the same outcome: the fully disclosed form of $[\supset\ [\wedge\ A\ B]\ A]$ in $\mathrm{TIL_M}$ is $[\forall\ \lambda A[\forall\ \lambda B.[\supset\ [\wedge\ A\ B]\ A]]]$ (see e.g. [7, p. 49]) and if we carry this TIL-construction out, we get the value true again. Also note that re-classifying it into so-called *singular problems*, i.e., problems that do not contain λ-bound Variables and are not singular concepts (i.e., Trivializations of non-constructions or Variables), would not help either since its solution would be still true (see [25, p. 103]). The "re-classification" can be achieved by removing the λ-binding and the quantifiers, hence $[\supset\ [\wedge\ A\ B]\ A]$ would become its fully disclosed form.

Let's consider e.g. the theorem $A \supset ((A \supset B) \supset B)$ again. From the CTT perspective, its solution would be:

$$\lambda y.\lambda x.\mathsf{Ap}(x, y)$$

which can be seen as a step by step instruction how to compose the corresponding ND proof (reading backwards): we see two variables x, y, hence two assumptions will be needed, then we see the function Ap, hence one instance of Implication Elimination rule will take place, and then we see two consecutive λ-abstractions, hence two consecutive applications of Implication Introduction rule withdrawing assumptions x and y, respectively. In other words, CTT's solution $\lambda y.\lambda x.\mathsf{Ap}(x, y)$ tells us exactly what it should, i.e., how to solve the problem $A \supset ((A \supset B) \supset B)$, unlike TIL_M's solution true.

Now, let's consider a different theorem $A \supset (B \supset A)$ to demonstrate our second issue with TIL_M, i.e., the conflation of all solutions. In the CTT paradigm its solution would be:

$$\lambda x.\lambda y.x$$

which is clearly a different term from $\lambda y.\lambda x.\mathsf{Ap}(x, y)$. Of course, it should come as no surprise, after all different problems should generally have different solutions. But in TIL_M paradigm the corresponding solution to the above problem would be again the truth value true. Thus, both theorems (and all others for that matter) have the same solution.

3.5. Summary of TIL_M-Driven Analysis

TIL_M analysis of mathematical problems was on par with CTT (at least to the extend of **(P1)**), however, we have encountered two drawbacks in TIL_M analysis of logical problems, as witnessed upon closer examination of the problem **(P2)**. Namely, uninformative solutions (a mere truth value does not provide any guidance solution wise) and the fact that all logical theorems have the same solution (all solutions of logical problem are conflated together and therefore indistinguishable). So TIL_M's logical solutions do not tell us how to solve the corresponding problems and further we cannot distinguish one solution from another. In the remainder of the paper, we will try to remedy these two issues, which are, of course, closely related. It is not difficult to see where the main cause for these troubles lies: TIL_M treats problems as ordinary true/false statements. But, as demonstrated in our case study B, this approach does

not yield good results when applied to logical problems. For inspiration on how to deal with this situation, we will turn to Kolmogorov.

4. TIL$^{+}_{M}$ Modification

4.1. Kolmogorov's Insight: Problems and Propositions

At the beginning of this paper, we have said that Kolmogorov suggested to interpret intuitionistic propositions as problems. That much is true but it is only the first half of the story, so to speak. The other half, which is often omitted, is that Kolmogorov actually didn't want to remove the notion of proposition from the intuitionistic vocabulary completely, he just had different intended meaning for it in mind. What's more, one of his ambitions was to establish some sort of dual-calculus operating simultaneously with both problems and propositions.

To put it shortly, Kolmogorov saw a difference between propositions and problems. In commentary section of *Selected Works of A. N. Kolmogorov* he states:

> Paper No. 19, "On the interpretation of intuitionistic logic" (IIL) [i.e. [15]] was written with the hope that the logic of solutions of problems would later become a regular part of courses on logic. It was intended to construct a unified logical apparatus dealing with objects of two types — propositions and problems. [17, p. 452]

and, if we go even more to the past, in his correspondence with Heyting he elaborates little bit more [16]:

> Each 'proposition' in your framework belongs, in my view, to one of two sorts:
>
> (α) p expresses hope that in prescribed circumstances, a certain experiment will always produce a specified result. (For example, that an attempt to represent an even number n as a sum of two primes will succeed upon exhausting all pairs $(p, q), p < n, q < n$.) Of course, every "experiment" must be realizable by a finite number of deterministic operations.
>
> (β) p expresses intention to find a certain construction.
>
> [...] I prefer to keep the name *proposition* (Aussage) only for propositions of type (α) and to call "propositions" of type β simply *problems* (Aufgaben). Associated to a *proposition* p are the problems $\sim p$ (to derive contradiction from p) and $+p$ (to prove p).

Although these two quotes are quite vague (Kolmogorov never offered fully satisfactory account of his position), we can get from them at least the general idea of the relation between problems and propositions. Specifically, that we can have them both in one system but we need to distinguish the two, keep them apart.

We will accept this Kolmogorov's distinction as a starting point of our upcoming modification of $\mathrm{TIL}_{\mathrm{M}}$, called $\mathrm{TIL}^{+}_{\mathrm{M}}$, and consider a system that deals with both problems, i.e., something that can be solved, and propositions, i.e., something that can be true or false, and views them as entities of different types.[21] This dual system setup allowing the introduction of propositions alongside problems will, however, constitute only the first part of our $\mathrm{TIL}^{+}_{\mathrm{M}}$ modification. The second part will rest upon the idea of viewing proofs as objects of higher-order than the objects they are proving.

4.2. Proofs as Higher-Order Objects

The main idea behind above mentioned change of perspective is quite simple. If we state: $a : A$, with the traditionally intended meaning "a is a proof establishing A", then a is usually understood as a proof-term or generally some sort of a justification to the truth of A. We intend to keep this general interpretation but with a slight twist inspired by TIL.

Recall that in TIL, due to its ramified type hierarchy, we can have 2nd-order TIL-constructions operating on 1st-order TIL-constructions, 3rd-order TIL-constructions operating on 2nd-order TIL-constructions and so on. With this in mind, we propose to understand the statement $a : A$ as claiming that a is some nth-order TIL-construction constructing $(n-1)$th-order TIL-construction A. In other words, we will take justifications to be objects on a different level than the objects they justify. This seems to be in line with some of our basic intuitions about justifying. After all, to justify something we have to provide the corresponding grounds for it. Metaphorically speaking, there has to be something we can put it on, something that will carry its weight.

Of course, in this context talking about *higher*-order TIL-constructions as "beneath positioned" justifications might be slightly confusing.

[21] In a sense, our effort can be considered as continuation of Kolmogorov's original unfulfilled desire of building a dual system. However, we will not try to reconstruct what Kolmogorov had in mind, rather we build upon his ideas. Similar task was recently undertaken also by Melikhov, see [30].

However, we can just as easily think of them as "*deeper*"-order TIL-constructions, i.e., mentally flip the ramified hierarchy upside down and we will get the more familiar picture of ever deeper going justifications. Thus, under this reading, e.g. $x : A$ will not mean that A is an assumption in some ND proof but rather that A is a propositional TIL-construction v-constructed by some other higher-order TIL-construction x. But, of course, it can still be interpreted in that way. This is the basic intuition we utilize in our approach.

It is important to note that the idea of constructions operating on other constructions is nothing new. Recall e.g. probably one of the most famous explanations of implication given by Heyting:

> The *implication* $p \supset q$ can be asserted, if and only if we possess a construction r, which, joined to any construction proving p (supposing that the latter be effected), would automatically effect a construction proving q. In other words, a proof of p, together with r, would form a proof of q [11, p. 98]

Here the construction r definitely seems to be acting as some sort of higher-order construction that takes construction proving p as an input and returns construction proving q as an output.

This, of course, didn't go unnoticed, e.g. van Dalen commented:

> In the case of implication we are faced with a construction which operates on construction [Sic!]... [52, pp. 133–134]

or recently even Howard himself stated in correspondence with Wadler the following [54]:

> I was not familiar with the work of Brouwer or Heyting, let alone Kolmogorov but, from what Kreisel had to say, the idea was clear enough: a construction of $p \supset q$ was to be a construction F which, acting on a construction A of p, gives a construction B of q. So we have constructions acting on constructions, rather like functionals acting on functionals.

This will be the general spirit of our approach as well. The only difference will be that for us constructions (or rather TIL-constructions) will be *always* operating on other constructions, not just in the case of implication.

4.3. Problem Assignment and Solution Constructor

Adopting Kolmogorov's ideas and thus moving away from "propositions-as-problems" conception towards "propositions-and-problems" concep-

tion, we now introduce $\text{TIL}^{+}_{\text{M}}$ modification, which is TIL_{M} extended with additional TIL-construction called Problem assignment and new version of match called solution constructor.

We start with *Problem assignment*. It will be denoted as: $?K$, where K is a TIL-construction. $?K$ can be read as "Find a solution for K", "Prove that K", "Construct K", etc. Typewise, if K is a *construction of order n over B* then $?K$ is a *construction of order* $n+1$ *over B*.

The basic idea is that while the TIL-construction K of some theorem will result in a truth value, i.e., true, TIL-construction $?K$ of a problem will result in a proper, i.e., non-boolean, solution. In other words, if K is a TIL-construction then $?K$ is a TIL-construction v-constructing a TIL-construction that v-constructs the K, e.g. in the scope of a certain ND proof. This, of course, opens up the issue of how to actually establish such proper solution.

To this purpose, we introduce a *solution constructor* that will be exclusively assigned to TIL-constructions of the form $?K$. It will be denoted as: $x :: K$, where x is a higher-order TIL-construction and K is a lower-order TIL-construction v-constructed by x. $x :: K$ can be read as "x is a solution of K", "x solves K", "x is a proof of K" or generally "x is a TIL-construction of K".[22]

The thought process behind the introduction of solution constructor is the following: even though in TIL_{M} all solution-objects to logical problems are the same (i.e., true), the solution-processes still differ (compare $\mathbf{c}''$ and $\mathbf{d}''$). Thus the solution constructor will *not* keep track of what the TIL-constructions are constructing — as did the notion of match[23] — but will rather record what was used to construct these TIL-constructions in the first place, i.e., it will track the overall solution-process itself. To put it differently, we utilize higher-order TIL-constructions to track the "logical moves" done with the lower-order TIL-constructions.

[22] Of course, not everything that counts as a TIL-construction of K can be immediately also considered as a proof of K. In other words, it has to be recognizable as such. For example, while **[** + **5 7]** constructs [+ **5 7**], the former can be hardly thought of as a solution to the latter.

[23] Recall that match $K : k$ tracked a special kind of congruence between K and k. From this perspective, we can think of solution constructor $x :: K$ as tracing another special case of congruence between x and K on the proviso that we double execute (recall Section 3.1) the TIL-construction on the left-hand side of ::. Take e.g. **[** + **5 7]** :: [+ **5 7**], while these two TIL-constructions are not congruent, ⇓**[** + **5 7]** and [+ **5 7**] are because they both construct the number 12.

Therefore, e.g. $x :: K$ can be read as either "TIL-construction K has been constructed by some higher-order TIL-construction x" or, and this is where the important part comes, as "K has been introduced as an assumption" in the scope of some ND proof. And, of course, TIL-constructions Closure and Composition can be used as well in a similar manner to CHdB correspondence, i.e., as tracking Implication Introduction and Implication Elimination rules, respectively.

Let's start with Implication Introduction rule. Assume we have some TIL-construction x of type $*_2$ that v-constructs A of type $*_1$ and we derive from it another TIL-construction y of type $*_2$ that v-constructs B of type $*_1$.[24] Hence, we have effectively established the premise for the Implication Introduction rule. Next, we introduce the higher-order function ii of type $(*_1*_1)$ that will model the behaviour of the required rule. More specifically, it is a higher-order function that takes one 1st-order TIL-construction as an argument (constructing a function that represents the above mentioned deduction of y from x) and returns another 1st-order TIL-construction as the conclusion, otherwise it is undefined. Consequently, $\mathbf{ii}/*_2$. The resulting TIL-construction $[\mathbf{ii}\ \lambda x.y]$ also of type $*_2$ then v-constructs $[\supset\ A\ B]$ of type $*_1$. Note that with some slight variations what we have just described is essentially an instance of Implication Introduction rule with CHdB correspondence attached. In other words, we are emulating the proof-tracking behaviour of CHdB correspondence within the toolset of TIL^+_{M}, i.e., via higher-order TIL-constructions and functions.

So changing the rule $\supset\mathbf{I}$ accordingly and integrating the above mentioned function ii, we get:

$$\frac{\begin{array}{c}(x :: A)\\ \vdots \\ y :: B\end{array}}{[\mathbf{ii}\ \lambda x.y] :: [\supset\ A\ B]}\ \supset\mathbf{I}^+$$

Recall that A and B are TIL-constructions (of type $*_1$) v-constructing non-empirical propositions of type o, i.e., TIL-constructions of truth values. Also note that the inference rule operates on solution constructors, not on matches.[25]

[24] Of course, we do not need to limit ourselves just to 1st- and 2nd-order TIL-constructions, we only do so to keep the presentation simple.

[25] We can observe that the rules $\supset$-Intro and $\supset$-Elim (or rather Π-Intro and Π-Elim) of CTT correspond to the rules $\supset\mathbf{I}^+$ and $\supset\mathbf{E}^+$ of TIL^+_{M}.

Now, to the Implication Elimination rule. Suppose we have some TIL-constructions x and y both of type $*_2$ that v-construct $[\supset A\ B]$ and A both of type $*_1$, i.e., premises of Implication Elimination rule. Next, we introduce a higher-order function that will emulate the behaviour of this particular rule. Specifically, we introduce the function ie of type $(*_1 *_1 *_1)$ that takes two 1st-order TIL-constructions of the form $[\supset A\ B]$ and A as premises and returns another 1st-order TIL-construction of the form B as their conclusion, otherwise it is undefined. It follows that $\mathbf{ie}/*_2$. Then the TIL-construction $[\mathbf{ie}\ x\ y]$ also of type $*_2$ v-constructs B of type $*_1$.

Again modifying the $\supset\mathbf{E}$ rule accordingly, we get:

$$\frac{x :: [\supset A\ B] \qquad y :: A}{[\mathbf{ie}\ x\ y] :: B}\ \supset\mathbf{E}^{+}$$

Note that **ie** appears to be equivalent to the operator Ap from CTT (recall Section 2.2). Strictly speaking, however, this is not the case. First of all, Ap is the operator for general function application and it is only due to the CHdB correspondence (not present in TIL) that we can use it for capturing the Implication Elimination rule as well. From this perspective, Ap rather corresponds to the TIL-construction Composition itself which takes care of function application. But if that is the case, why then not use simply $[x\ y]$ instead of $[\mathbf{ie}\ x\ y]$? The reason for this is the following: $[x\ y]$ is a Composition and as such it requires that its first component, x in this case, v-constructs a function, otherwise it is an improper TIL-construction, i.e., a TIL-construction that v-constructs nothing (see Appendix). Assume that x and y v-construct $[\supset A\ B]$ and A, respectively, as the rule demands. Then we effectively end up with an improper TIL-construction $[[\supset A\ B]\ A]$ because $[\supset A\ B]$ does not v-construct a function but a truth value. Hence, we need to introduce **ie** to circumvent this issue. Secondly, recall that the rule for computing Ap is essentially a β-reduction (the rule Π-**Comp** in Section 2.2). This is not the case for **ie** in TIL, where β-reduction is a rule for equivalent transformation of TIL-constructions (see e.g. [7, 39]) and as such it cannot be used to compute the value of $[\mathbf{ie}\ x\ y]$, i.e., determine what it v-constructs. Thus, roughly put, while Ap in CTT captures the behaviour of Implication Elimination rule (among other things), **ie** captures in $\mathrm{TIL}^{+}_{\mathrm{M}}$ *only* the behaviour Implication Elimination rule.

Now, we possess everything necessary to solve the theorems $A \supset ((A \supset B) \supset B)$ and $A \supset (B \supset A)$ in a satisfactory manner. For

comparison, we first produce their proofs using match (i.e., solving K):

$$\dfrac{\dfrac{\dfrac{([\supset\ A\ B] : \mathsf{true}) \qquad (A : \mathsf{true})}{B : \mathsf{true}}\supset\mathbf{E}}{[\supset\ \ [\supset\ A\ B]\ B] : \mathsf{true}}\supset\mathbf{I}}{[\supset\ A\ [\supset\ \ [\supset\ A\ B]\ B]] : \mathsf{true}}\supset\mathbf{I} \qquad \dfrac{\dfrac{(A : \mathsf{true})}{[\supset\ B\ A] : \mathsf{true}}\supset\mathbf{I}}{[\supset\ A[\supset\ B\ A]] : \mathsf{true}}\supset\mathbf{I}$$

Now, if we use the solution constructor instead of match (i.e., solving $?K$) and the corresponding set of rules, we get:

$$\dfrac{\dfrac{\dfrac{(x :: [\supset\ A\ B]) \qquad (y :: A)}{[\mathbf{ie}\ x\ y] :: B}\supset\mathbf{E}^{+}}{[\mathbf{ii}\ \lambda x.[\mathbf{ie}\ x\ y]] :: [\supset\ \ [\supset\ A\ B]\ B]}\supset\mathbf{I}^{+}}{[\mathbf{ii}\ \lambda y.[\mathbf{ii}\ \lambda x.[\mathbf{ie}\ x\ y]]] :: [\supset\ A\ [\supset\ \ [\supset\ A\ B]\ B]]}\supset\mathbf{I}^{+}$$

$$\dfrac{\dfrac{(x :: A)}{[\mathbf{ii}\ \lambda y.x] :: [\supset\ B\ A]}\supset\mathbf{I}^{+}}{[\mathbf{ii}\ \lambda x.[\mathbf{ii}\ \lambda y.x]] :: [\supset\ A[\supset\ B\ A]]}\supset\mathbf{I}^{+}$$

which is finally the result we were looking for, i.e., different solution-objects ($[\mathbf{ii}\ \lambda y.[\mathbf{ii}\ \lambda x.[\mathbf{ie}\ x\ y]]]$ and $[\mathbf{ii}\ \lambda x.[\mathbf{ii}\ \lambda y.x]]$) for different problems ($?[\supset\ A\ [\supset\ \ [\supset\ A\ B]\ B]]$ and $?[\supset\ A[\supset\ B\ A]]$). Hence, we have successfully unscrambled Materna's conflated solutions true into proper and distinguishable solution-objects.

Note that while $[\supset\ A\ [\supset\ \ [\supset\ A\ B]\ B]]$ results in true, $?[\supset\ A\ [\supset\ [\supset\ A\ B]\ B]]$ results in $[\mathbf{ii}\ \lambda y.[\mathbf{ii}\ \lambda x.[\mathbf{ie}\ x\ y]]]$. So, in a sense, the former outcome can be seen as an answer to the question "Is $A \supset ((A \supset B) \supset B)$ a theorem?", while the latter as an answer to the question "Why is $A \supset ((A \supset B) \supset B)$ a theorem?", i.e., it presents a proof. Or to put it differently, $[\supset\ A\ [\supset\ \ [\supset\ A\ B]\ B]]$ can be seen as a (construction of) proposition and $?[\supset\ A\ [\supset\ \ [\supset\ A\ B]\ B]]$ as a problem. Hence, we have both problems and propositions in our system, as Kolmogorov envisioned it, the latter serving as sort of "derived" versions of the former. Note that while we can get from $[\mathbf{ii}\ \lambda y.[\mathbf{ii}\ \lambda x.[\mathbf{ie}\ x\ y]]]$ to the truth value true, i.e., we can reconstruct the proof and see that it indeed establishes a theorem, the opposite direction is not possible: from true alone we cannot get to $[\mathbf{ii}\ \lambda y.[\mathbf{ii}\ \lambda x.[\mathbf{ie}\ x\ y]]]$. So, in a sense, problems are more fundamental than propositions.

Of course, once equipped with solution constructor, there is no need to stop at $\supset\mathbf{I}^+$ and $\supset\mathbf{E}^+$ and we can easily introduce rules for additional logical connectives. Although our case studies do not require it, we show how we can formulate solution-tracking rules for conjunction in an analogous manner:

$$\frac{x :: A \qquad y :: B}{[\mathbf{ci}\ x\ y] :: [\wedge\ A\ B]}\ \wedge\mathbf{I}^+ \qquad \frac{z :: [\wedge\ A\ B]}{[\mathbf{pr}_l\ z] :: A}\ \wedge\mathbf{E}_l^+ \qquad \frac{z :: [\wedge\ A\ B]}{[\mathbf{pr}_r\ z] :: B}\ \wedge\mathbf{E}_r^+$$

where **ci** constructs the "Conjunction Introduction rule" function in a similar way as did **ii** previously for Implication Introduction rule. The components $\mathbf{pr}_l$ and $\mathbf{pr}_r$ construct the familiar left and right projection functions. Types of all these functions can be easily inferred: **ci** has type $*_2$ and constructs a higher-order function of type $(*_1\ *_1\ *_1)$ and $\mathbf{pr}_l/\mathbf{pr}_r$ are also of type $*_2$ and construct higher-order functions of type $(*_1 *_1)$.

With these rules ready, we can e.g. prove $?[\supset\ [\wedge\ A\ B]\ A]$ as follows:

$$\frac{\dfrac{(z :: [\wedge\ A\ B])}{[\mathbf{pr}_l\ z] :: A}\ \wedge\mathbf{E}_l^+}{[\mathbf{ii}\ \lambda z.[\mathbf{pr}_l\ z]] :: [\supset\ [\wedge\ A\ B]\ A]}\ \supset\mathbf{I}^+$$

But it is important to reiterate that, strictly speaking, CHdB correspondence is not present here (or anywhere else in $\mathrm{TIL}^+_{\mathrm{M}}$), we are just emulating its solution-tracking aspect via TIL's native tools (see the table below).

typed λ-calculus	*natural deduction*	$\mathrm{TIL}^+_{\mathrm{M}}$
term variable	assumption	higher-order Variable
term	proof	higher-order TIL-construction
type	proposition	lower-order TIL-construction
type constructor	connective	function
type inhabitation	provability	constructibility via higher-order TIL-construction

Thus, although it might look similar "on paper", the underlying notions as well as the technical scaffolding are different. Most notably, while in CTT we work with judgements "*term* : *type*" in $\mathrm{TIL}^+_{\mathrm{M}}$ we have "*TIL-construction* :: *TIL-construction*". In other words, $\mathrm{TIL}^+_{\mathrm{M}}$ does not incorporate CHdB correspondence but it can simulate it in order to catch

up with CTT in terms of a finger-grained analysis of logical problems, which was our goal.

That is not to say, however, that $\text{TIL}^{+}_{\text{M}}$ does not have anything interesting to offer on its own. Most importantly, given the ramified hierarchy of types, we can consider not only proofs of TIL-constructions that v-construct propositions (as in $x :: A$, which can be read as "assume A") but also proofs of those proofs (as in $z :: x$, which can be interpreted as "assume that $x :: A$ is an assumption") and so on. Furthermore, we could introduce *two-dimensional* match of the form $x :: A \simeq y :: B$ that would track the congruence between solution constructors — more specifically, between the respective proofs (the first dimension) and the proved objects (the second dimension). Recall e.g. the proofs **d** and **c**, with the two-dimensional match we could capture their relationship as follows:

$$[\mathbf{ii}\ \lambda y.[\mathbf{ii}\ \lambda x.[\mathbf{ie}\ \lambda y.[\mathbf{ie}\ x\ y]\ y]]] :: C\ \simeq\ [\mathbf{ii}\ \lambda y.[\mathbf{ii}\ \lambda x.[\mathbf{ie}\ x\ y]]] :: C$$

where C is $[\supset A\ [\supset\ [\supset A\ B]\ B]]$. This two-dimensional match (if satisfied) would then state that the proofs **d** and **c** are congruent/equivalent. This naturally leads to the topic of equality of proofs in $\text{TIL}^{+}_{\text{M}}$. That is, however, worthy of its own investigation and it is far outside the scope of this paper.

5. Final Remarks

We have surveyed two different conceptions of problems utilizing something that can be generally described as algorithmic or procedural semantics, namely CTT relying on "propositions-as-problems" approach and $\text{TIL}/\text{TIL}_{\text{M}}$ relying on "TIL-constructions-as-problems" approach.

Both approaches dealt with mathematical problems (case study A with the problem **(P1)**) in a satisfactory manner, however, logical problems (case study B with the problem **(P2)**) posed difficulties for TIL_{M}. More precisely, TIL_{M} offered insufficient "boolean" solutions and conflated all solutions together. Further, we tried to show that TIL_{M}-based approach is feasible, although significant modifications (i.e., $\text{TIL}^{+}_{\text{M}}$ inspired by Kolmogorov and CTT) were needed, specifically, the introduction of new TIL-construction called *Problem assignment* and adoption of *solution constructor*, which were used to mimic the CHdB correspondence in order to avoid TIL_{M}'s original shortcomings.

Also note that from the CTT perspective the problems **(P1)** and **(P2)** — although similar in form at first glance — are of a different nature. More specifically, the problem $5 + 7 : \mathbb{N}$ is really a problem of checking whether $5+7$ evaluates to a natural number, namely 12 (i.e., $succ(5+6) : \mathbb{N}$). On the other hand, the problem $A \supset ((A \supset B) \supset B)$ prompts us not for a mere evaluation but for a construction of a certain object, namely, $\lambda y \lambda x.\mathsf{Ap}(x, y)$.[26]

Observe, however, that these are not really two distinct kinds of problems, rather two different stages of solving them, which can be perhaps called *establishing* and *evaluative* stage. Take e.g. the problem $A \supset ((A \supset B) \supset B)$. At the first establishing stage, we present either $\lambda y \lambda x.\mathsf{Ap}(x, y)$ or $\lambda y \lambda x.\mathsf{Ap}(\lambda y.\mathsf{Ap}(x, y), y)$ as a solution. And at the second evaluative stage, we check their correctness, i.e., whether they are in/reducible to canonical form. Analogously in the case of problem **(P1)**: first, we come up with the number 12 as a solution, then we check whether it is a natural number.

In TIL_M, no such two phases can be identified and both **(P1)** and **(P2)** are considered only at the evaluative stage. In other words, TIL_M, in a sense, skips the inventive aspect of problem solving, i.e., the part of "coming up with a solution", and deals only with its correctness checking aspect. This is why TIL_M yielded such similar results with CTT in the case study A in comparison to the case study B: both CTT and TIL_M dealt with the problem **(P1)** only from the evaluative perspective. Given the difficulties TIL_M had encountered with logical problems, we have to conclude that in the overall contest of CTT vs. TIL_M, it was the former that won the first round dedicated to non-empirical problems.[27] However, as we have also shown, these difficulties can be rectified by adopting TIL^+_M. Future work, the second round so to speak, will be focused on empirical problems.

Appendix

TIL-constructions are defined as follows (originally defined by [49], we adopt it here from [7] with slight changes):

[26] I would like to thank Ansten Klev for bringing this point to my attention.

[27] This was to be expected — after all CTT was devised primarily as a framework for constructive mathematics, while TIL_M or TIL in general was designed with a heavy emphasis on natural language analysis.

1. The *Variable* x is a *TIL-construction* that constructs an object O of the respective type dependently on a valuation. We say that it v-constructs O.
2. Where X is any object, $\mathbf{X}$ is the *TIL-construction Trivialization*. It constructs X without any change.
3. The *Composition* $[K\, L_1 \dots L_m]$ is the following *TIL-construction*. If K v-constructs a function f of a type $(\alpha\beta_1 \dots \beta_m)$ and $L_1 \dots L_m$ v-construct objects $b_1, \dots, b_m$ of types $\beta_1, \dots, \beta_m$, respectively, then the *Composition* $[K\, L_1 \dots L_m]$ *v-constructs* the value (an object of type α, if any) of f on the tuple-argument $\langle b_1, \dots, b_m \rangle$. Otherwise, it is a *v-improper TIL-construction*, i.e., *TIL-construction* that does not construct anything.
4. The *Closure* $\lambda x_1 \dots x_m \,.\, K$ is the following *TIL-construction*. Let $x_1, \dots x_m$ be pairwise distinct Variables v-constructing objects of types $\beta_1, \dots, \beta_m$ and K a TIL-construction v-constructing an object of type α. Then $\lambda x_1 \dots x_m \,.\, K$ is the *TIL-construction Closure*. It *v-constructs* the following function f of type $(\alpha\beta_1 \dots \beta_m)$: let $\langle b_1, \dots, b_m \rangle$ be a tuple of objects of type $\beta_1 \dots \beta_m$, respectively, and v' be a valuation that associates x_i with b_i and is identical to v otherwise. Then the value of function f on argument tuple $\langle b_1, \dots, b_m \rangle$ is the object of type α v'-constructed by K. If K is v'-improper, then f is undefined on $\langle b_1, \dots, b_m \rangle$.
5. The *Single execution* $\downarrow K$ is the *TIL-construction* that either v-constructs the object v-constructed by K or, if K v-constructs nothing, is v-improper.
6. The *Double execution* $\Downarrow K$ is the following *TIL-construction*: let K be any object, the *Double execution* $\Downarrow K$ is *v-improper* if K is a non-construction or if K does not v-construct a TIL-construction or if K v-constructs a v-improper TIL-construction. Otherwise, let K v-construct a TIL-construction K' and let K' v-construct and object K'', then $\Downarrow K$ v-constructs K''.
7. Nothing other is a *TIL-construction*, unless it follows from 1–6.

Acknowledgements. An early version of this paper was presented at the "Oberseminar Logik und Sprachtheorie", held at Universität Tübingen (December 2014), and I want to thank the audience for discussion and helpful comments. Also I would like to thank Ansten Klev and Jiří Raclavský for various notes regarding earlier drafts of this paper. And finally, I would like to thank anonymous reviewers for their extensive and most helpful remarks and suggestions.

The work on this paper was supported by the grant of the Czech Science Foundation (GAČR) "Semantic Notions, Paradoxes and Hyperintensional Logic Based on Modern Theory of Types", registration no. GA16-19395S.

References

[1] Church, A., "A set of postulates for the foundation of logic", *Annals of Mathematics* 33, 2 (1932): 346–366. DOI: 10.2307/1968337

[2] Church, A., "A formulation of the simple theory of types", *The Journal of Symbolic Logic* 5, 6 (1940): 56–68. DOI: 10.2307/2266170

[3] Curry, H. B., and R. Feys, *Combinatory Logic*, volume 1 of "Combinatory Logic", North-Holland Publishing Company, 1958.

[4] De Bruijn, N., "Automath, a language for mathematics", Technical report, Department of Mathematics, Eindhoven University of Technology, 1968.

[5] Duží, M., and B. Jespersen, "Procedural isomorphism, analytic information and β-conversion by value", *Logic Journal of IGPL* 21, 2 (2013): 291–308. DOI: 10.1093/jigpal/jzs044

[6] Duží, M., and P. Materna, "Can concepts be defined in terms of sets?", *Logic and Logical Philosophy* 19, 3 (2010): 195–242. DOI: 10.12775/LLP.2010.008

[7] Duží, M., B. Jespersen, and P. Materna, *Procedural Semantics for Hyperintensional Logic: Foundations and Applications of Transparent Intensional Logic*, "Logic, Epistemology, and the Unity of Science", Springer, 2010. DOI: 10.1007/978-90-481-8812-3

[8] Gentzen, G., "Untersuchungen über das logische Schließen. I", *Mathematische Zeitschrift* 39, 1 (1935): 176–210. DOI: 10.1007/BF01201353

[9] Girard, J.-Y., *Proofs and Types*, "Cambridge Tracts in Theoretical Computer Science" 7, Cambridge University Press, 1989.

[10] Granström, J. G., *Treatise on Intuitionistic Type Theory*, "Logic, Epistemology, and the Unity of Science", Springer Netherlands, 2011. DOI: 10.1007/978-94-007-1736-7

[11] Heyting, A., *Intuitionism: An Introduction*, "Studies in Logic and the Foundations of Mathematics", North-Holland Pub. Co., 1956.

[12] Horák, A., *Computer Processing of Czech Syntax and Semantics*, Librix.eu, Brno, Czech Republic, 2008.

[13] Howard, W. A., "The formulae-as-types notion of construction", pages 479–490 in J. R. Hindley and J. P. Seldin (eds.), *To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus, and Formalism*, Academic Press, 1980.

[14] Jespersen, B., and M. Carrara, "A new logic of technical malfunction", *Studia Logica* 101, 3 (2013): 547–581. DOI: 10.1007/s11225-012-9397-8

[15] Kolmogorov, A. N., "Zur Deutung der intuitionistischen Logik", *Mathematische Zeitschrift* 35, 1 (1932): 58–65. English translation in Tikhomirov 1991, pp. 151–158 and Mancosu 1998, pp. 328–334. DOI: 10.1007/BF01186549

[16] Kolmogorov, A. N., "Letters of A. N. Kolmogorov to A. Heyting", *Russian Mathematical Surveys* 43, 6 (1988): 89–93. DOI: 10.1070/RM1988v043n06ABEH001986

[17] Kolmogorov, A. N., "On the interpretation of intuitionistic logic", pages 151–158 in V. M. Tikhomirov (ed.), *Selected Works of A. N. Kolmogorov*, volume 25 of "Mathematics and Its Applications (Soviet Series)", Springer Netherlands, 1991. DOI: 10.1007/978-94-011-3030-1_19

[18] Kovář, V., A. Horák, and M. Jakubíček, "How to analyze natural language with Transparent Intensional Logic?", pages 69–76 in P. Sojka and A. Horák (eds.), *Proceedings of Recent Advances in Slavonic Natural Language Processing*, RASLAN 2010.

[19] Martin-Löf, P., "An intuitionistic theory of types: Predicative part", pages 73–118 in H. E. Rose and J. C. Shepherdson (eds.), *Logic Colloquium '73Proceedings of the Logic Colloquium*, volume 80 of "Studies in Logic and the Foundations of Mathematics", Elsevier, 1975. DOI: 10.1016/S0049-237X(08)71945-1

[20] Martin-Löf, P., "Hauptsatz for the intuitionistic theory of iterated inductive definitions", pages 197–215 in J. E. Fenstad (ed.), *Proceedings of the Second Scandinavian Logic Symposium (Oslo 1970)*, North-Holland, 1971.

[21] Martin-Löf, P., "About models for intuitionistic type theories and the notion of definitional equality", pages 81–109, North-Holland Publishing Company, 1975.

[22] Martin-Löf, P., "Constructive mathematics and computer programming", pages 153–175 in J. L. Cohen and J. Łoś *et al.* (eds.), *Logic, Methodology and Philosophy of Science VI, 1979*, North-Holland, 1982.

[23] Martin-Löf, P., *Intuitionistic Type Theory*, "Studies in Proof Theory", Bibliopolis, 1984.

[24] Materna, P., *Concepts and Objects*, Acta Philosophica Fennica, Helsinki: Philosophical Society of Finland, vol. 63, 1998.

[25] Materna, P., *Conceptual Systems*, Logische Philosophie Logos, 2004.

[26] Materna, P., "Ordinary modalities", *Logique et Analyse* 48, 57–70 (2005): 513–554.

[27] Materna, P., "The notion of problem, intuitionism and partiality", *Logic and Logical Philosophy* 17, 4 (2008): 287–303. DOI: 10.12775/LLP.2008.016

[28] Materna, P., "Mathematical and empirical concepts", pages 209–233 in J. Maclaurin (ed.), *Rationis Defensor*, volume 28 of "Studies in History and Philosophy of Science", Springer Netherlands, 2012. DOI: 10.1007/978-94-007-3983-3_16

[29] Materna, P., "Is Transparent Intensional Logic a non-classical logic?", *Logic and Logical Philosophy* 23, 1 (2014): 47–55. DOI: 10.12775/LLP.2013.032

[30] Melikhov, S. A., "A galois connection between classical and intuitionistic logics. I: Syntax", 2013.

[31] Moschovakis, Y. N., "A logical calculus of meaning and synonymy", *Linguistics and Philosophy* 29, 1 (2006): 27–89. DOI: 10.1007/s10988-005-6920-7

[32] Muskens, R., "Sense and the computation of reference", *Linguistics and Philosophy* 28, 4 (2005): 473–504. DOI: 10.1007/s10988-004-7684-1

[33] Nordström, B., K. Petersson, and J. M. Smith, *Programming in Martin-Löf's Type Theory: An Introduction*, International series of monographs on computer science, Clarendon Press, 1990.

[34] Nordström, B., K. Petersson, and J M. Smith, *Martin-Löf's Type Theory. Handbook of Logic in Computer Science*, Volume 5, "Logic and Algebraic Methods", Oxford University Press, Oxford, 2001.

[35] Pierce, B. C., *Types and Programming Languages*, MIT Press, 2002.

[36] Primiero, G., and B. Jespersen, "Two kinds of procedural semantics for privative modification", *Lecture Notes in Artificial Intelligence* 6284: 251–271, 2010. DOI: 10.1007/978-3-642-14888-0_21

[37] Raclavský, J., "On the interaction of semantics and deduction in Transparent Intensional Logic (Is Tichý's logic a logic?)", *Logic and Logical Philosophy* 23, 1 (2014): 57–68. DOI: 10.12775/LLP.2013.035

[38] Raclavský, J., and P. Kuchyňka, "Conceptual and derivation systems", *Logic and Logical Philosophy* 20, 1–2 (2011): 159–174. DOI: 10.12775/LLP.2011.008

[39] Raclavský, J., P. Kuchyňka, and I. Pezlar, *Transparent Intensional Logic as Characteristica Universalis and Calculus Ratiocinator* (in Czech), Brno: Masaryk University (Munipress), Brno, 2015.

[40] Ranta, A., *Type-Theoretical Grammar*, Indices, Clarendon Press, 1994.

[41] Simmons, H., *Derivation and Computation: Taking the Curry-Howard Correspondence Seriously*, Cambridge Tracts in Theoretical Computer Science, Cambridge University Press, 2000.

[42] Sørensen, M. H., and P. Urzyczyn, *Lectures on the Curry-Howard Isomorphism*, Volume 149 of "Studies in Logic and the Foundations of Mathematic", Elsevier Science Inc., New York, NY, USA, 2006.

[43] Stergios, Ch., and Z. Luo, *Modern Perspectives in Type-Theoretical Semantics*, Springer Publishing Company, Incorporated, 1st edition, 2017. DOI: 10.1007/978-3-319-50422-3

[44] Sundholm, G., "Constructions, proofs and the meaning of logical constants", *Journal of Philosophical Logic* 12, 2 (1983): 151–172. DOI: 10.1007/BF00247187

[45] Thompson, S., *Type Theory and Functional Programming*, International computer science series, Addison-Wesley, 1999.

[46] Tichý, P., "Foundations of partial type theory", *Reports on Mathematical Logic* 14 (1982): 59–72.
[47] Tichý, P., "Constructions", *Philosophy of Science* 53, 4 (1986): 514–534.
[48] Tichý, P., "Indiscernibility of identicals", *Studia Logica* 45, 3 (1986): 251–273. DOI: 10.1007/BF00375897
[49] Tichý, P., *The Foundations of Frege's Logic*, Foundations of Communication, de Gruyter, 1988.
[50] Tichý, P., V. Svoboda, B. Jespersen, and C. Cheyne (eds.), *Pavel Tichý's Collected Papers in Logic and Philosophy*, Filosofia, 2004.
[51] The Univalent Foundations Program, *Homotopy Type Theory: Univalent Foundations of Mathematics*, Institute for Advanced Study, 2013. https://homotopytypetheory.org/book
[52] van Dalen, D., "Interpreting intuitionistic logic", in P. C. Baayen, D. van Dulst, and J. Oosterhoff (eds.), *Proceedings, Bicentennial Congress, Wiskundig Genootschap*, Amsterdam: Mathematisch Centrum, 1979.
[53] van Heijenoort, J., *From Frege to Gödel: A Source Book in Mathematical Logic, 1879–1931*, Source Books in the History of the Sciences, Harvard University Press, 1977.
[54] Wadler, Ph., "Propositions as types", Draft, 29 November 2014. http://homepages.inf.ed.ac.uk/wadler/papers/propositions-as-types/propositions-as-types.pdf
[55] Whitehead, A. N., and B. Russell, *Principia Mathematica*, Cambridge University Press, 1910.
[56] Więckowski, B., "Constructive belief reports", *Synthese* 192, 3 (2015): 603–633. DOI: 10.1007/s11229-014-0540-0

IVO PEZLAR
Department of Philosophy
Masaryk University
Arna Nováka 1
Brno, Czech Republic
pezlar@phil.muni.cz