frontiers | Frontiers in Computer Science

ORIGINAL RESEARCH
published: 12 July 2022
doi: 10.3389/fcomp.2022.792570

Semantics for Combinatory Logic With Intersection Types

Silvia Ghilezan [1,2]* *and Simona Kašterović* [1]

[1] *Faculty of Technical Sciences, University of Novi Sad, Novi Sad, Serbia,* [2] *Mathematical Institute of the Serbian Academy of Sciences and Arts, Belgrade, Serbia*

There is a plethora of semantics of computational models, nevertheless, the semantics of combinatory logic are among the less investigated ones. In this paper, we propose semantics for the computational system of combinatory logic with intersection types. We define extensional applicative structures endowed with special elements corresponding to primitive combinators. We prove two soundness and completeness results. First, the equational theory of untyped combinatory logic is proven to be sound and complete with respect to the proposed semantics. Second, the system of the combinatory logic with intersection types is proven to be sound and complete with respect to the proposed semantics. The usual approach to the semantics for calculi with types that can be found in the literature is based on models for the untyped calculus endowed with a valuation of type variables which enables the interpretation of types to be defined inductively. We propose, however, a different approach. In the semantics we propose, the interpretation of types is represented as a family of subsets that satisfies certain properties, whereas for a given valuation of term variables, the interpretation of terms is defined inductively. Due to the wide applicability of semantics of computational models, the presented approach could be further developed to other computational models and beyond—to current and foreseen application of semantics to large distributed systems and new challenging technologies.

Keywords: computational systems, combinatory logic, equational theory, type theory, intersection types, soundness, completeness, semantics

OPEN ACCESS

Edited by:
Jonni Virtema,
The University of Sheffield,
United Kingdom

Reviewed by:
Marcos Cramer,
Technical University Dresden,
Germany
Harsh Beohar,
The University of Sheffield,
United Kingdom

***Correspondence:**
Silvia Ghilezan
gsilvia@uns.ac.rs

Specialty section:
This article was submitted to Theoretical Computer Science, a section of the journal Frontiers in Computer Science

Received: *10 October 2021*
Accepted: *17 June 2022*
Published: *12 July 2022*

Citation:
Ghilezan S and Kašterović S (2022) Semantics for Combinatory Logic With Intersection Types. Front. Comput. Sci. 4:792570. doi: 10.3389/fcomp.2022.792570

1. INTRODUCTION

In the 1920s two models of computation were invented, *combinatory logic* and *lambda calculus*. The foundations of combinatory logic and lambda calculus was established in Curry (1930) and Church (1936), respectively. They were developed with the aim to describe functions, their behavior and properties. Nowadays they serve as a basis for the design of functional programming languages and programming languages at large. These two models of computation have the same computational power, they can express the same computational concepts; however the syntax of combinatory logic is much simpler. The basic idea of combinatory logic, introduced in Schönfinkel (1924), is to build functions without using variables. In order to control function application, type systems were introduced for both combinatory logic and lambda calculus. A recent comprehensive overview of combinators is given in Wolfram (2021). Apart from usual application in programming languages, typed combinatory logic has found its application in a wide range of different fields, such as machine learning, artificial intelligence, program synthesis. Developments of these fields urge for further investigation and development of the theory of combinatory logic.

In computation and type theory, intersection types characterize exactly all strongly normalizing lambda terms (Coppo and Dezani-Ciancaglini, 1978; Sallé, 1978; Pottinger, 1980) providing, on the computational side, a significant extension of simple types, which do not type even some normal forms, such as $\lambda x.xx$. On the semantical side, intersection types became a powerful tool suitable for analysing lambda models (Barendregt et al., 1983, 2013). Intersection types for different systems have been studied (e.g., van Bakel et al., 2018; de'Liguoro and Treglia, 2019). Combinatory logic was equipped with intersection types in Dezani-Ciancaglini and Hindley (1992) with the same computational advantage of characterizing exactly all strongly normalizing combinatory terms, i.e., computations that always terminate.

The main contributions and results of this paper:

- We propose a novel semantics for combinatory logic with intersection types. We define extensional applicative structures endowed with special elements corresponding to primitive combinators S, K and I.
- We prove that the equational theory of untyped combinatory logic is sound and complete with respect to the proposed semantics.
- We then prove that combinatory logic with intersection types is sound and complete with respect to the proposed semantics.

1.1. Background and Motivation

There have been several approaches to semantics of lambda calculus with intersection types (Barendregt et al., 1983; Dezani-Ciancaglini and Margaria, 1984; Barbanera et al., 1995; Ong and Tsukada, 2012; de Carvalho, 2018) nevertheless none of them addresses combinatory logic. In Barendregt et al. (1983), the authors have introduced the filter models for lambda calculus and proved the soundness and completeness of the type assignment system with respect to the proposed semantics. Further, in Dezani-Ciancaglini and Margaria (1984) a modification of intersection type discipline has been considered, and it has been proved that this type system is sound and complete with respect to the *F*-semantics. The type assignment system with intersection and union types has been introduced in Barbanera et al. (1995), where authors proposed three semantics for the introduced system and proved soundness and completeness results. In de Carvalho (2018), the author introduced a non-uniform semantics of lambda calculus which allows to measure execution time and presented the intersection type system induced by these semantics, which can be seen as a reformulation of the system of Coppo et al. (1980). In Ong and Tsukada (2012), the two-level game semantics is used to model intersection types.

On the other hand, semantics for combinatory logic with intersection types has not been investigated as much as the semantics for lambda calculus with intersection types. An overview of models of combinatory logic is given in Bimbó (2012), however most of these models are models of untyped combinatory logic. Particularly, one class of models for combinatory logic with intersection types is presented based on theories of types which are in $1-1$ correspondence with filters. Nevertheless, to the best of our knowledge filter models for combinatory logic with intersection types has not been published anywhere. The lack of study of models of combinatory logic has been our motivation for the present research.

$$M = M \quad (id) \qquad \mathsf{S}MNL = (ML)(NL) \quad (S) \qquad \mathsf{K}MN = M \quad (K)$$

$$\mathsf{I}M = M \quad (I) \qquad \frac{M = N}{N = M}\ (\text{sym}) \qquad \frac{M = N \quad N = L}{M = L}\ (\text{trans})$$

$$\frac{M = N}{MP = NP}\ (\text{app-l}) \qquad \frac{M = N}{PM = PN}\ (\text{app-r})$$

FIGURE 1 | Equational theory $\mathcal{EQ}$.

1.2. Organization of the Paper

In Section 2, we briefly review basic notions of the untyped combinatory logic and intersection type assignment system for combinatory logic. We then define the semantics in Section 3. Section 4 presents the results of the paper: in Section 4.1, we prove the soundness and completeness of the equational theory of the untyped combinatory logic with respect to the proposed semantics; in Section 4.2, we prove the soundness and completeness of the combinatory logic with intersection types with respect to the proposed semantics. Section 5 concludes and contains the discussion of the related work.

2. COMBINATORY LOGIC WITH INTERSECTION TYPES

We shortly review some basic notions of the untyped combinatory logic *CL* (Barendregt, 1985; Hindley and Seldin, 1986; Bimbó, 2012), and intersection types for combinatory logic (Dezani-Ciancaglini and Hindley, 1992).

2.1. Combinatory Logic

The language of combinatory logic is build up from a countable set of term variables V and a set of term constants using application, the only (binary) operation. Terms of combinatory logic, called *CL*-terms, are generated by the following grammar:

$$M, N ::= x \mid \mathsf{S} \mid \mathsf{K} \mid \mathsf{I} \mid MN$$

where x is a term variable and $\mathsf{S}, \mathsf{K}, \mathsf{I}$ are term constants, or constants for short. The set of all *CL*-terms is denoted by *CL* and is ranged over by capital letters $M, N, \ldots, M_1, \ldots$. The set of variables that occur in a term M is denoted by $FV(M)$, and the result of substitution of the term N for variable x in the term M is denoted by $M\{N/x\}$. The use of parentheses is minimized by the convention that application associates to the left: $M_1 M_2 \ldots M_n$ means $[\ldots(M_1 M_2)\ldots M_n]$.

The main objects of study in combinatory logic are the relations between terms. We will consider the equivalence relation generated by the equational theory $\mathcal{EQ}$ given in **Figure 1**.

If $M = N$ can be derived from the set of axioms and rules of **Figure 1** we say that terms M and N are equal, and we write

$M = N$. The equational theory obtained by adding the rule

$$\frac{Mx = Nx \qquad x \notin FV(M) \cup FV(N)}{M = N} \text{ (ext)}$$

to the equational theory $\mathcal{EQ}$ will be denoted by $\mathcal{EQ}^{\eta}$, and it represents the extensional equational theory. We write $M =_{\eta} N$, whenever $M = N$ is provable in $\mathcal{EQ}^{\eta}$. We use the notion of equivalence classes: $[M] = \{N \mid M = N \text{ is provable in } \mathcal{EQ}^{\eta}\}$.

2.2. Intersection Types

Intersection types for combinatory logic were introduced in Dezani-Ciancaglini and Hindley (1992). The set of intersection types is generated by the following grammar:

$$\sigma, \tau ::= a \mid \omega \mid \sigma \to \tau \mid \sigma \cap \tau$$

where a belongs to a countable set of type variables V_{Type} and ω is a type constant. There are two type forming operations: the *arrow*, $\to$, which generates the simple types and the *intersection*, $\cap$, which provides the extension of the simple types. The set of all intersection types is denoted by Types and is ranged over by $\sigma, \tau, \ldots, \sigma_1, \ldots$. We can omit parentheses in arrow types by removing outer parentheses and using the convention that, unless otherwise bracketed, nested arrows associate to the right: $\sigma_1 \to \ldots \to \sigma_{n-1} \to \sigma_n$ means $(\sigma_1 \to \ldots \to (\sigma_{n-1} \to \sigma_n))$. Intersection types are associative: $(\sigma \cap \tau) \cap \rho$ is the same as $\sigma \cap (\tau \cap \rho)$, and an intersection type takes precedence over an arrow type: $\sigma \cap \tau \to \rho$ means $(\sigma \cap \tau) \to \rho$.

Definition 2.1 (Dezani-Ciancaglini and Hindley, 1992; Barendregt et al., 2013).

(i) *A statement* is of the form $M : \sigma$, where $M \in CL$ and $\sigma \in$ Types. The term M is the *subject* and the type σ is the *predicate* of the statement.
(ii) A *declaration* is a statement with a term variable as subject, i.e., $x : \sigma$.
(iii) A *basis* (*context*) is a set of declarations with distinct term variables as subjects.
(iv) Let $\Gamma = \{x_1 : \sigma_1, \ldots, x_n : \sigma_n\}$ be a basis. The set $dom(\Gamma) = \{x_1, \ldots, x_n\}$ is the *domain* of the basis Γ.
(v) Let $\Gamma = \{x_1 : \sigma_1, \ldots, x_n : \sigma_n\}$ be a basis. Then $|\Gamma| = \{\sigma_1, \ldots, \sigma_n\}$.

The subtyping (pre-order) relation on the set of intersection types is defined in the following way.

Definition 2.2 (Dezani-Ciancaglini and Hindley, 1992). The relation $\leq$ is the smallest binary relation satisfying:

(1) $\sigma \leq \sigma$;
(2) $\sigma \leq \sigma \cap \sigma$;
(3) $\sigma \leq \omega$;
(4) $\sigma \cap \tau \leq \sigma, \sigma \cap \tau \leq \tau$;
(5) $\omega \leq \omega \to \omega$;
(6) $(\sigma \to \rho) \cap (\sigma \to \tau) \leq \sigma \to (\rho \cap \tau)$;
(7) if $\sigma \leq \tau, \tau \leq \rho$ then $\sigma \leq \rho$;
(8) if $\sigma \leq \sigma_1, \tau \leq \tau_1$, then $\sigma \cap \sigma_1 \leq \tau \cap \tau_1$;
(9) if $\sigma \leq \sigma_1, \tau \leq \tau_1$, then $\sigma_1 \to \tau \leq \sigma \to \tau_1$,

The induced equivalence relation $\sim$ is defined by

$$\sigma \sim \tau \quad \text{if and only if } \sigma \leq \tau \text{ and } \tau \leq \sigma.$$

The type-assignment system for combinatory logic with intersection types, denoted by $CL_{\cap}$, is presented in **Figure 2**. If the typing judgment $\Gamma \vdash M : \sigma$ can be derived by the rules in **Figure 2**, then it means that the statement $M : \sigma$ is derivable from the basis Γ and that the term M is typable in the given basis Γ with type σ.

The rules (axiom S), (axiom K), and (axiom I) type the combinators S, K, and I, respectively. The rule (axiom ω) enables every combinatory term M to be typable. The rules ($\to$ elim), ($\cap$ elim-l), and ($\cap$ elim-r) are the usual elimination rules for $\to$ and $\cap$, respectively. The rule ($\cap$ intro) is the introduction rule for $\cap$. The rule (sub-type) is the subsumption rule induced by the subtyping relation on types and the rule (eq) is the rule which enables type preservation under equality.

Remark 2.3. The type-assignment system TA_{CLB_1} of Dezani-Ciancaglini and Hindley (1992) is obtained from $CL_{\cap}$ by omitting the rule (eq). Type-assignment statements in TA_{CLB_1} are preserved by the equality generated by the equational theory $\mathcal{EQ}$ (Dezani-Ciancaglini and Hindley, 1992, Theorem 3.12). In order to obtain a type-assignment system in which types are preserved by the equality generated by the extensional equational theory $\mathcal{EQ}^{\eta}$, the rule (eq) has to be added.

Lemma 2.4 (Weakening lemma). *If* $\Gamma \vdash M : \sigma$ *and* $\Gamma \subseteq \Gamma'$ *then* $\Gamma' \vdash M : \sigma$.

Proof: By induction on the length of derivation of $\Gamma \vdash M : \sigma$. □

3. SEMANTICS FOR $CL_{\cap}$

In this section, we introduce a semantics for combinatory logic with intersection types. We define a model for $CL_{\cap}$ as an applicative structure equipped with an environment.

Definition 3.1. *An applicative structure for* $CL_{\cap}$ is a tuple

$$\mathcal{M} = \langle D, \{A^{\sigma}\}, App \rangle$$

which consists of:

(i) a non-empty set D, called *domain*,
(ii) a family $\{A^{\sigma}\}$ of sets indexed by types σ, such that for every $\sigma, \tau \in$ Types,
  - $A^{\sigma} \subseteq D$,
  - $A^{\omega} = D$,
  - $A^{\sigma \cap \tau} = A^{\sigma} \cap A^{\tau}$,
  - if $\sigma \leq \tau$, then $A^{\sigma} \subseteq A^{\tau}$.
(iii) an application function $App : D \times D \to D$, such that for every $\sigma, \tau \in$ Types, $App \restriction (A^{\sigma \to \tau} \times A^{\sigma}) : A^{\sigma \to \tau} \times A^{\sigma} \to A^{\tau}$.

Definition 3.2. An applicative structure

$$\mathcal{M} = \langle D, \{A^{\sigma}\}, App \rangle$$

$$\frac{x:\sigma\in\Gamma}{\Gamma\vdash x:\sigma}\ (\text{axiom}\in) \qquad \frac{}{\Gamma\vdash \mathsf{S}:(\sigma\to(\rho\to\tau))\to(\sigma\to\rho)\to(\sigma\to\tau)}\ (\text{axiom } S)$$

$$\frac{}{\Gamma\vdash \mathsf{K}:\sigma\to(\tau\to\sigma)}\ (\text{axiom } K) \qquad \frac{}{\Gamma\vdash \mathsf{I}:\sigma\to\sigma}\ (\text{axiom } I)$$

$$\frac{}{\Gamma\vdash M:\omega}\ (\text{axiom }\omega) \qquad \frac{\Gamma\vdash M:\sigma\to\tau \quad \Gamma\vdash N:\sigma}{\Gamma\vdash MN:\tau}\ (\to\text{ elim})$$

$$\frac{\Gamma\vdash M:\sigma\cap\tau}{\Gamma\vdash M:\sigma}\ (\cap\text{ elim-l}) \qquad \frac{\Gamma\vdash M:\sigma\cap\tau}{\Gamma\vdash M:\tau}\ (\cap\text{ elim-r})$$

$$\frac{\Gamma\vdash M:\sigma \quad \Gamma\vdash M:\tau}{\Gamma\vdash M:\sigma\cap\tau}\ (\cap\text{ intro}) \qquad \frac{\Gamma\vdash M:\sigma \quad \sigma\leq\tau}{\Gamma\vdash M:\tau}\ (\text{sub-type})$$

$$\frac{\Gamma\vdash M:\sigma \quad M=_\eta N}{\Gamma\vdash N:\sigma}\ (\text{eq})$$

FIGURE 2 | Type-assignment system $CL_\cap$.

has combinators if there exist elements $\mathbf{s},\mathbf{k},\mathbf{i}$ of the domain D such that

- $\mathbf{s}\in A^{(\sigma\to(\rho\to\tau))\to((\sigma\to\rho)\to(\sigma\to\tau))}$ for every $\sigma,\rho,\tau\in$ Types,
- $\mathbf{k}\in A^{\sigma\to(\tau\to\sigma)}$ for every $\sigma,\tau\in$ Types,
- $\mathbf{i}\in A^{\sigma\to\sigma}$ for every $\sigma\in$ Types,
- for every $a,b,c\in D$ the following equations hold:

$$App(App(App(\mathbf{s},a),b),c)=App(App(a,c),App(b,c)) \quad (1)$$

$$App(App(\mathbf{k},a),b)=a \quad (2)$$

$$App(\mathbf{i},a)=a \quad (3)$$

Definition 3.3. An applicative structure

$$\mathcal{M}=\langle D,\{A^\sigma\},App\rangle$$

is extensional if for all $f,g\in D$ it holds that

$$\text{if } (\forall a\in D)(App(f,a)=App(g,a)),\ \text{then } f=g.$$

Definition 3.4. *An environment* ρ for an applicative structure $\mathcal{M}$ is a mapping from a set of term variables to the domain, $\rho:V\to D$.

Although the proposed semantics is not Kripke-style semantics, its definition is inspired by the Kripke-style semantics introduced in Mitchell and Moggi (1991) and Kašterović and Ghilezan (2020), where a model is also defined as an applicative structure provided with an environment. The important difference between Definition 3.4 and the definition of an environment in Kašterović and Ghilezan (2020) is the fact that the environment in Kašterović and Ghilezan (2020) was a partial mapping, whereas herein it is a total mapping. The motivation for defining environments as total mappings will be explained in Remark 4.5.

Let ρ be an environment for $\mathcal{M}$ and $a\in D$. By $\rho(x:=a)$ we denote the environment for $\mathcal{M}$ such that for all $y\in V$

$$\rho(x:=a)(y)=\begin{cases}a, & y=x\\ \rho(y), & y\neq x\end{cases}$$

Definition 3.5. A model for $CL_\cap$ is a pair $\mathcal{M}_\rho=\langle\mathcal{M},\rho\rangle$, where $\mathcal{M}=\langle D,\{A^\sigma\},App\rangle$ is an extensional applicative structure for $CL_\cap$ with combinators, and ρ is an environment for $\mathcal{M}$.

Definition 3.6. Let $\mathcal{M}_\rho$ be a model for $CL_\cap$. We define the meaning of a term M in the environment ρ, denoted by $[\![M]\!]_\rho$, inductively as follows:

1. $[\![x]\!]_\rho=\rho(x)$,
2. $[\![\mathsf{S}]\!]_\rho=\mathbf{s}$,
3. $[\![\mathsf{K}]\!]_\rho=\mathbf{k}$,
4. $[\![\mathsf{I}]\!]_\rho=\mathbf{i}$,
5. $[\![MN]\!]_\rho=App([\![M]\!]_\rho,[\![N]\!]_\rho)$.

Note that the meaning of a term does not depend on the meaning of the variables that do not occur in the term. Since an environment ρ is a total mapping and $\rho(x)$ is defined for every variable x, the meaning of every term M, denoted by $[\![M]\!]_\rho$, is also well-defined.

Remark 3.7. An important difference between the applicative structure we define and the one introduced in Mitchell and Moggi (1991) is that we have a domain in the applicative structure. In Mitchell and Moggi (1991) the authors define the interpretation of the judgment $\Gamma\vdash M:\sigma$, they interpret the term with its type. In turn, we wanted to be able to define the interpretation of a term independent of its type. For this reason we have introduced a domain as part of the applicative structure.

Lemma 3.8. *Let $\mathcal{M}$ be an extensional applicative structure for $CL_\cap$ with combinators, ρ_1 and ρ_2 environments for $\mathcal{M}$ and M a CL-term. If $\rho_1(x)=\rho_2(x)$ holds for every variable $x\in FV(M)$, then $[\![M]\!]_{\rho_1}=[\![M]\!]_{\rho_2}$.*

Proof: We prove the statement by induction on the structure of the term M.

- Let M be a variable x. If $\rho_1(x) = \rho_2(x)$, then $[\![x]\!]_{\rho_1} = \rho_1(x) = \rho_2(x) = [\![x]\!]_{\rho_2}$.
- If M is a term constant, then its interpretation does not depend on the environment, but only on the applicative structure. For example, let M be a term constant S, then $[\![\mathsf{S}]\!]_{\rho_1} = \mathbf{s} = [\![\mathsf{S}]\!]_{\rho_2}$.
- If M is an application NL, then by the induction hypothesis the statement holds for terms N and L, $[\![N]\!]_{\rho_1} = [\![N]\!]_{\rho_2}$ and $[\![L]\!]_{\rho_1} = [\![L]\!]_{\rho_2}$. Now, by Definition 3.6 we have

$$\begin{aligned}[\![M]\!]_{\rho_1} &= [\![NL]\!]_{\rho_1} = App([\![N]\!]_{\rho_1}, [\![L]\!]_{\rho_1}) = App([\![N]\!]_{\rho_2}, [\![L]\!]_{\rho_2}) \\ &= [\![NL]\!]_{\rho_2} = [\![M]\!]_{\rho_2}.\end{aligned}$$

□

Lemma 3.9 (Substitution lemma). *Let M, N be CL-terms and ρ an environment for an applicative structure $\mathcal{M}$. Then,*

$$[\![M\{N/x\}]\!]_\rho = [\![M]\!]_{\rho(x:=[\![N]\!]_\rho)}.$$

The satisfiability of a statement $M:\sigma$ in a model and the notion of semantical consequence are defined as follows.

Definition 3.10 (Kašterović and Ghilezan, 2020).

1. A statement $M:\sigma$ is satisfied in a model M_ρ, denoted by $\mathcal{M}_\rho \models M:\sigma$, if and only if $[\![M]\!]_\rho \in A^\sigma$. In this case we say $\mathcal{M}_\rho$ is a model of the statement $M:\sigma$. If a model $\mathcal{M}_\rho$ does not satisfy the statement $M:\sigma$, we write $\mathcal{M}_\rho \not\models M:\sigma$.
2. A model $\mathcal{M}_\rho$ is a model of a basis Γ, denoted by $\mathcal{M}_\rho \models \Gamma$, if and only if every declaration $x:\sigma$ from the basis Γ is satisfied in the model $\mathcal{M}_\rho$, namely $\mathcal{M}_\rho \models x:\sigma$.
3. A statement $M:\sigma$ is a semantical consequence of a basis Γ, denoted by $\Gamma \models M:\sigma$, if and only if every model of the basis Γ is also a model of the statement $M:\sigma$.

4. SOUNDNESS AND COMPLETENESS RESULTS

In this section we present the main results of the paper. We prove in Section 4.1 that the equational theory of the untyped combinatory logic $\mathcal{EQ}^\eta$ is sound and complete with respect to the proposed semantics. In Section 4.2, we prove that the type-assignment system $CL_\cap$ is sound and complete with respect to the proposed semantics.

4.1. Soundness and Completeness of Equational Theory $\mathcal{EQ}^\eta$

In the previous section we have introduced models for $CL_\cap$ such that the interpretation of every term is defined in a model. A fundamental question to ask is: if two terms are proven to be equal in the equational theory are their interpretations equal in a model?—this is referred to as **the soundness** of the equational theory with respect to the model. The other direction: if interpretations of two terms are equal in every model are the terms proven to be equal in the equational theory?—is referred to as **the completeness**.

In the sequel, we prove that the equational theory of untyped combinatory logic $\mathcal{EQ}^\eta$ is sound and complete with respect to the proposed semantics.

Theorem 4.1 (Soundness of $\mathcal{EQ}^\eta$). *If $M =_\eta N$, then $[\![M]\!]_\rho = [\![N]\!]_\rho$ for every model $\mathcal{M}_\rho$.*

Proof: The proof is by induction on the length of the proof of $M = N$ in $\mathcal{EQ}^\eta$.

If $M = N$ is an instance of axiom (*id*), then terms M and N represent the same term L and it holds that $[\![M]\!]_\rho = [\![L]\!]_\rho = [\![N]\!]_\rho$ for every model $\mathcal{M}_\rho$.
If $M = N$ is obtained by axiom (*S*), then there exist terms P, Q and R such that terms M and N are terms $\mathsf{S}PQR$ and $(PR)(QR)$, respectively. Then, from Definitions 3.2 and 3.6 we obtain

$$\begin{aligned}[\![\mathsf{S}PQR]\!]_\rho &= App(App(App([\![\mathsf{S}]\!]_\rho, [\![P]\!]_\rho), [\![Q]\!]_\rho), [\![R]\!]_\rho) \\ &= App(App(App(\mathbf{s}, [\![P]\!]_\rho), [\![Q]\!]_\rho), [\![R]\!]_\rho) \\ &= App(App([\![P]\!]_\rho, [\![R]\!]_\rho), App([\![Q]\!]_\rho, [\![R]\!]_\rho)) \\ &= [\![(PR)(QR)]\!]_\rho.\end{aligned}$$

If $M = N$ is obtained by axiom (*K*), then there exist terms P and Q, such that terms M and N are terms $\mathsf{K}PQ$ and P, respectively. Then, from Definitions 3.2 and 3.6 we obtain

$$\begin{aligned}[\![\mathsf{K}PQ]\!]_\rho &= App(App([\![\mathsf{K}]\!]_\rho, [\![P]\!]_\rho), [\![Q]\!]_\rho) \\ &= App(App(\mathbf{k}, [\![P]\!]_\rho), [\![Q]\!]_\rho) \\ &= [\![P]\!]_\rho.\end{aligned}$$

If $M = N$ falls under axiom (*I*), then terms M and N are of the form $\mathsf{I}P$ and P, respectively, for some term P. In a similar way as in the previous two cases, we obtain

$$[\![\mathsf{I}P]\!]_\rho = App([\![\mathsf{I}]\!]_\rho, [\![P]\!]_\rho) = App(\mathbf{i}, [\![P]\!]_\rho) = [\![P]\!]_\rho.$$

If $M = N$ is obtained from $N = M$ by rule (sym), then by induction hypothesis $[\![N]\!]_\rho = [\![M]\!]_\rho$ and the statement holds.

If $M = N$ is obtained from $M = L$ and $L = N$ by rule (trans), then $[\![M]\!]_\rho = [\![L]\!]_\rho$ and $[\![L]\!]_\rho = [\![N]\!]_\rho$ hold by induction hypothesis. Thus, $[\![M]\!]_\rho = [\![N]\!]_\rho$ also holds.

Next, let us suppose that $M = N$ is obtained from $L = Q$ by rule (app-l), then terms M and N are of the form LP and QP, respectively, for some term P. By induction hypothesis $[\![L]\!]_\rho = [\![Q]\!]_\rho$. From the latter we obtain

$$[\![LP]\!]_\rho = App([\![L]\!]_\rho, [\![P]\!]_\rho) = App([\![Q]\!]_\rho, [\![P]\!]_\rho) = [\![QP]\!]_\rho.$$

Similarly to the previous case, if $M = N$ is obtained by rule (app-r), then terms M and N are of the form PL and PQ, respectively, for some terms P, L, Q such that $L = Q$ is provable in $\mathcal{EQ}^\eta$. By induction hypothesis we obtain $[\![L]\!]_\rho = [\![Q]\!]_\rho$, and thus

$$[\![PL]\!]_\rho = App([\![P]\!]_\rho, [\![L]\!]_\rho) = App([\![P]\!]_\rho, [\![Q]\!]_\rho) = [\![PQ]\!]_\rho.$$

Finally, we consider the case where $M = N$ is obtained by applying rule (ext) to $Mx = Nx$, with x being a variable

which appears neither in M nor in N. Let $\mathcal{M}_\rho$ be a model for $CL_\cap$. In order to prove that $[\![M]\!]_\rho = [\![N]\!]_\rho$ holds, it is enough to prove that for every $d \in D$ it holds that $App([\![M]\!]_\rho, d) = App([\![N]\!]_\rho, d)$, because of the extensionality of the applicative structure $\mathcal{M}$. Let $d \in D$. Since by induction hypothesis $[\![Mx]\!]_\rho = [\![Nx]\!]_\rho$ holds in any model, it also holds in model $\mathcal{M}_{\rho(x:=d)}$, that is $App([\![M]\!]_{\rho(x:=d)}, [\![x]\!]_{\rho(x:=d)}) = App([\![N]\!]_{\rho(x:=d)}, [\![x]\!]_{\rho(x:=d)})$. From the assumption that x does not appear in M and Lemma 3.8, we obtain $[\![M]\!]_\rho = [\![M]\!]_{\rho(x:=d)}$. Similarly, $[\![N]\!]_\rho = [\![N]\!]_{\rho(x:=d)}$. Now, we have

$$\begin{aligned} App([\![M]\!]_\rho, d) &= App([\![M]\!]_{\rho(x:=d)}, d) \\ &= App([\![M]\!]_{\rho(x:=d)}, [\![x]\!]_{\rho(x:=d)}) \\ &= [\![Mx]\!]_{\rho(x:=d)} \\ &= [\![Nx]\!]_{\rho(x:=d)} \\ &= App([\![N]\!]_{\rho(x:=d)}, [\![x]\!]_{\rho(x:=d)}) \\ &= App([\![N]\!]_\rho, d). \end{aligned}$$

Thus, $[\![M]\!]_\rho = [\![N]\!]_\rho$ due to the extensionality of the applicative structure.

This concludes the proof. □

Recall that by $[M]$ we denote the equivalence class of term M with respect to the equivalence relation generated by the equational theory $\mathcal{EQ}^\eta$, $[M] = \{N \mid M = N \text{ is provable in } \mathcal{EQ}^\eta\}$.

In order to prove the completeness, we first prove the following property.

Lemma 4.2. *Let $\mathcal{M} = \langle D, \{A^\sigma\}, App\rangle$ be an extensional applicative structure with combinators, such that $D = \{[M] \mid M \in CL\}$, $App([M],[N]) = [MN]$, $\boldsymbol{s} = [\mathsf{S}]$, $\boldsymbol{k} = [\mathsf{K}]$ and $\boldsymbol{i} = [\mathsf{I}]$. If an environment $\rho^\star$ is defined by $\rho^\star(x) = [x]$, then $[\![M]\!]_{\rho^\star} = [M]$.*

Proof: The proof is by induction on the structure of M.

- If the term M is a variable x, then by Definition 3.6 we have $[\![x]\!]_{\rho^\star} = \rho^\star(x) = [x]$.
- If the term M is a term constant, for example S, then by Definition 3.6 we have $[\![\mathsf{S}]\!]_{\rho^\star} = \mathbf{s} = [\mathsf{S}]$. Similarly, for term constants K and I.
- If the term M is an application NL, then by the induction hypothesis the statement holds for terms N and L, $[\![N]\!]_{\rho^\star} = [N]$ and $[\![L]\!]_{\rho^\star} = [L]$. Further, by Definition 3.6 we obtain $[\![NL]\!]_{\rho^\star} = App([\![N]\!]_{\rho^\star}, [\![L]\!]_{\rho^\star}) = App([N],[L]) = [NL]$.

□

Theorem 4.3 (Completeness of $\mathcal{EQ}^\eta$). *If $[\![M]\!]_\rho = [\![N]\!]_\rho$ holds in every model $\mathcal{M}_\rho$, then $M =_\eta N$.*

Proof: Suppose that $[\![M]\!]_\rho = [\![N]\!]_\rho$ holds for any model $\mathcal{M}_\rho$. We define a model $\mathcal{M}'_{\rho^\star} = \langle \mathcal{M}_0, \rho^\star\rangle$ where $\mathcal{M}_0$ is an applicative structure $\langle D, \{A^\sigma\}, App\rangle$ with the following components:

- the domain D is defined by $D = \{[M] \mid M \in CL\}$;
- for every $\sigma \in \mathsf{Types}$, $A^\sigma = \{[M] \mid M \in CL \text{ and } \vdash M : \sigma\}$;
- App is a function defined by $App([M],[N]) = [MN]$.

The environment $\rho^\star$ is defined by $\rho^\star(x) = [x]$.

First we prove that the structure $\mathcal{M}_0$ is an applicative structure, meaning that it satisfies conditions of Definition 3.1. The set D is a non-empty set. For every $\sigma \in \mathsf{Types}$, the set $A^\sigma = \{[M] \mid M \in CL \text{ and } \vdash M : \sigma\}$ is a subset of the set $D = \{[M] \mid M \in CL\}$. Since for every term M it holds that $\vdash M : \omega$, we have that $A^\omega = \{[M] \mid M \in CL \text{ and } \vdash M : \omega\} = \{[M] \mid M \in CL\} = D$. Next, we look at the sets A^σ and A^τ. By the definition of the applicative structure $\mathcal{M}_0$ we have $A^\sigma = \{[M] \mid M \in CL \text{ and } \vdash M : \sigma\}$ and $A^\tau = \{[M] \mid M \in CL \text{ and } \vdash M : \tau\}$. If $[M] \in A^\sigma \cap A^\tau$, then $\vdash M : \sigma$ and $\vdash M : \tau$. By rule ($\cap$ intro) of **Figure 2** we obtain $\vdash M : \sigma \cap \tau$, which is equivalent to $[M] \in A^{\sigma\cap\tau}$, and we conclude $A^\sigma \cap A^\tau \subseteq A^{\sigma\cap\tau}$. Similarly, we prove that $A^{\sigma\cap\tau} \subseteq A^\sigma \cap A^\tau$ and we conclude $A^{\sigma\cap\tau} = A^\sigma \cap A^\tau$. Let $\sigma, \tau \in \mathsf{Types}$ such that $\sigma \leq \tau$. If $[M] \in A^\sigma$, then $\vdash M : \sigma$ holds, and since $\sigma \leq \tau$ we obtain $\vdash M : \tau$ by rule (sub-type). It follows that $[M] \in A^\tau$ and we can conclude that $\sigma \leq \tau$ implies $A^\sigma \subseteq A^\tau$. The function App defined by $App([M],[N]) = [MN]$ maps $D \times D$ to D. If $[M] \in A^{\sigma\to\tau}$ and $[N] \in A^\sigma$, then by the definition of the applicative structure $\mathcal{M}_0$ we have $\vdash M : \sigma \to \tau$ and $\vdash N : \sigma$. By rule ($\to$ elim) of **Figure 2** we obtain $\vdash MN : \tau$, thus we conclude $[MN] \in A^\tau$ and $App \restriction (A^{\sigma\to\tau} \times A^\sigma) : A^{\sigma\to\tau} \times A^\sigma \to A^\tau$.

Next, we prove that the applicative structure $\mathcal{M}_0$ has combinators. We consider the equivalence classes $[\mathsf{S}], [\mathsf{K}], [\mathsf{I}] \in D$ and denote them by $\mathbf{s}, \mathbf{k}, \mathbf{i}$, respectively. Since $\vdash \mathsf{S} : (\sigma \to (\tau \to \rho)) \to ((\sigma \to \tau) \to (\sigma \to \rho))$ holds by (axiom S) of **Figure 2**, we have that $\mathbf{s} = [\mathsf{S}] \in A^{(\sigma\to(\tau\to\rho))\to((\sigma\to\tau)\to(\sigma\to\rho))}$ for any $\sigma, \tau, \rho \in \mathsf{Types}$. Similarly, we obtain $\mathbf{k} = [\mathsf{K}] \in A^{\sigma\to(\tau\to\sigma)}$ and $\mathbf{i} = [\mathsf{I}] \in A^{\sigma\to\sigma}$. We also have to prove that the elements $\mathbf{s}, \mathbf{k}$, and $\mathbf{i}$ satisfy the Equations (1)–(3). By the rules of the equational theory $\mathcal{EQ}^\eta$ we obtain $[\mathsf{S}MNL] = [(ML)(NL)]$, $[\mathsf{K}MN] = [M]$ and $[\mathsf{I}M] = [M]$ and it follows that:

$$\begin{aligned} App(App(App([\mathsf{S}],[M]),[N]),[L]) &= [\mathsf{S}MNL] = [(ML)(NL)] \\ &= App([ML],[NL]) \\ &= App(App([M],[L]), App([N],[L])) \end{aligned}$$

$$App(App([\mathsf{K}],[M]),[N]) = [\mathsf{K}MN] = [M]$$

$$App([\mathsf{I}],[M]) = [\mathsf{I}M] = [M]$$

The extensionality of the applicative structure follows from the extensionality of combinatory logic. Let $[M],[N] \in D$ such that for every $[L] \in D$, $App([M],[L]) = App([N],[L])$, i.e., $[ML] = [NL]$. Then, for a variable x, that does not appear in terms M and N, it holds that $[Mx] = [Nx]$, that is $Mx = Nx$ is provable in $\mathcal{EQ}^\eta$. By rule (ext) we obtain $M = N$ is provable in $\mathcal{EQ}^\eta$, so $[M] = [N]$.

We have proved that $\mathcal{M}'_{\rho^\star}$ is a model for $CL_\cap$. If $[\![M]\!]_\rho = [\![N]\!]_\rho$ in any model, then it also holds in the model $\mathcal{M}'_{\rho^\star}$. Hence, we have that $[\![M]\!]_{\rho^\star} = [\![N]\!]_{\rho^\star}$. From Lemma 4.2 we get $[\![M]\!]_{\rho^\star} = [M]$ and $[\![N]\!]_{\rho^\star} = [N]$. Now we can conclude that $[M] = [N]$ holds, that is $M =_\eta N$. □

Theorems 4.1 and 4.3 prove that the equational theory of untyped combinatory logic is both sound and complete with respect to the proposed semantics, hence the proposed semantics is a semantics of the untyped combinatory logic.

4.2. Soundness and Completeness of Type-Assignment System $CL_\cap$

In the sequel we consider another fundamental question: if $M:\sigma$ is (syntactically) derivable in the type system $CL_\cap$, $\vdash M:\sigma$, is it valid in all models, $\models M:\sigma$?—this is referred to as **the soundness** of the type-assignment system with respect to the model. More general, if $\Gamma \vdash M:\sigma$ is (syntactically) derivable in the type system $CL_\cap$, is it semantically derivable, $\Gamma \models M:\sigma$? The other direction: if $M:\sigma$ is valid in all models of $CL_\cap$, is it derivable in $CL_\cap$?—is referred to as **the completeness**. More general, does semantical derivability $\Gamma \models M:\sigma$ imply syntactical derivability $\Gamma \vdash M:\sigma$?

We prove herein the type-assignment system $CL_\cap$ to be sound and complete with respect to the proposed semantics.

Theorem 4.4 (Soundness of $CL_\cap$). *If* $\Gamma \vdash M:\sigma$, *then* $\Gamma \models M:\sigma$.

Proof: We prove the statement by induction on the derivation of $\Gamma \vdash M:\sigma$, considering the last rule used.

- Let $\dfrac{x:\sigma \in \Gamma}{\Gamma \vdash x:\sigma}$ be the last rule used. Assume that $\mathcal{M}_\rho$ is a model, such that $\mathcal{M}_\rho \models \Gamma$. By Definition 3.10, the latter implies that for every $y:\tau \in \Gamma$, $\mathcal{M}_\rho \models y:\tau$. From the premise we have $x:\sigma \in \Gamma$, thus $\mathcal{M}_\rho \models x:\sigma$.
- If $\Gamma \vdash M:\sigma$ falls under (axiom S), then we have $\Gamma \vdash \mathsf{S}:(\sigma_1 \to (\rho_1 \to \tau_1)) \to ((\sigma_1 \to \rho_1) \to (\sigma_1 \to \tau_1))$ for some $\sigma_1,\tau_1,\rho_1 \in \mathsf{Types}$. Let $\mathcal{M}_\rho$ be an arbitrary model for $CL_\cap$. By Definition 3.6 we have that $[\![\mathsf{S}]\!]_\rho = \mathbf{s} \in A^{(\sigma_1\to(\rho_1\to\tau_1))\to((\sigma_1\to\rho_1)\to(\sigma_1\to\tau_1))}$. Thus, $\mathcal{M}_\rho \models \mathsf{S}:(\sigma_1 \to (\rho_1 \to \tau_1)) \to ((\sigma_1 \to \rho_1) \to (\sigma_1 \to \tau_1))$ and we can conclude $\Gamma \models \mathsf{S}:(\sigma_1 \to (\rho_1 \to \tau_1)) \to ((\sigma_1 \to \rho_1) \to (\sigma_1 \to \tau_1))$.
- If $\Gamma \vdash M:\sigma$ falls under (axiom K), then we have $\Gamma \vdash \mathsf{K}:\sigma_1 \to (\tau_1 \to \sigma_1)$ for some $\sigma_1,\tau_1 \in \mathsf{Types}$. Let $\mathcal{M}_\rho$ be an arbitrary model for $CL_\cap$. We have that $[\![\mathsf{K}]\!]_\rho = \mathbf{k} \in A^{\sigma_1\to(\tau_1\to\sigma_1)}$. Thus, $\mathcal{M}_\rho \models \mathsf{K}:\sigma_1 \to (\tau_1 \to \sigma_1)$ holds and we can conclude $\Gamma \models \mathsf{K}:\sigma_1 \to (\tau_1 \to \sigma_1)$.
- If $\Gamma \vdash M:\sigma$ falls under (axiom I), then we have $\Gamma \vdash \mathsf{I}:\sigma_1 \to \sigma_1$ for some $\sigma_1 \in \mathsf{Types}$. For an arbitrary model $\mathcal{M}_\rho$ it holds that $[\![\mathsf{I}]\!]_\rho = \mathbf{i} \in A^{\sigma_1\to\sigma_1}$. Hence, $\mathcal{M}_\rho \models \mathsf{I}:\sigma_1 \to \sigma_1$ and $\Gamma \models \mathsf{I}:\sigma_1 \to \sigma_1$.
- If $\Gamma \vdash M:\sigma$ falls under (axiom ω), then we have $\Gamma \vdash M:\omega$. Let $\mathcal{M}_\rho$ be an arbitrary model for $CL_\cap$. We have that $[\![M]\!]_\rho \in D$ and $A^\omega = D$. From this we obtain $[\![M]\!]_\rho \in A^\omega$, that is $\mathcal{M}_\rho \models M:\omega$.
- Let $\dfrac{\Gamma \vdash M:\sigma \to \tau \qquad \Gamma \vdash N:\sigma}{\Gamma \vdash MN:\tau}$ be the last rule used and $\mathcal{M}_\rho$ a model which satisfies Γ. By the induction hypothesis, we have that the statement holds for premises $\Gamma \vdash M:\sigma \to \tau$ and $\Gamma \vdash N:\sigma$, i.e., $\Gamma \models M:\sigma \to \tau$ and $\Gamma \models N:\sigma$. From the induction hypothesis and the assumption $\mathcal{M}_\rho \models \Gamma$, we obtain $\mathcal{M}_\rho \models M:\sigma \to \tau$ and $\mathcal{M}_\rho \models N:\sigma$. The former implies $[\![M]\!]_\rho \in A^{\sigma\to\tau}$ and the latter implies $[\![N]\!]_\rho \in A^\sigma$. Thus, it holds that $[\![MN]\!]_\rho = App([\![M]\!]_\rho,[\![N]\!]_\rho) \in A^\tau$, and we obtain $\mathcal{M}_\rho \models MN:\tau$.
- Let $\dfrac{\Gamma \vdash M:\sigma \cap \tau}{\Gamma \vdash M:\sigma}$ be the last rule used and $\mathcal{M}_\rho$ a model of Γ. By the induction hypothesis we have $\mathcal{M}_\rho \models M:\sigma \cap \tau$, which implies that $[\![M]\!]_\rho \in A^{\sigma\cap\tau}$ holds. Since $A^{\sigma\cap\tau} = A^\sigma \cap A^\tau$ by Definition 3.1, we have $[\![M]\!]_\rho \in A^\sigma \cap A^\tau \subseteq A^\sigma$. Thus, we conclude $\mathcal{M}_\rho \models M:\sigma$.
- If the last rule used is $\dfrac{\Gamma \vdash M:\sigma \cap \tau}{\Gamma \vdash M:\tau}$ the proof proceeds similarly as in the previous case.
- Let $\dfrac{\Gamma \vdash M:\sigma \qquad \Gamma \vdash M:\tau}{\Gamma \vdash M:\sigma \cap \tau}$ be the last rule used and $\mathcal{M}_\rho$ a model of Γ. By the induction hypothesis we obtain $\mathcal{M}_\rho \models M:\sigma$ and $\mathcal{M}_\rho \models M:\tau$. The former implies that $[\![M]\!]_\rho \in A^\sigma$ holds, and the latter implies that $[\![M]\!]_\rho \in A^\tau$ holds. Since $A^\sigma \cap A^\tau = A^{\sigma\cap\tau}$ by Definition 3.1, we have $[\![M]\!]_\rho \in A^\sigma \cap A^\tau = A^{\sigma\cap\tau}$. Finally, we conclude $\mathcal{M}_\rho \models M:\sigma \cap \tau$.
- Let $\dfrac{\Gamma \vdash M:\sigma \qquad \sigma \le \tau}{\Gamma \vdash M:\tau}$ be the last rule used and $\mathcal{M}_\rho$ a model which satisfies the basis Γ. By the induction hypothesis we obtain that the statement holds for the premise, i.e., $\Gamma \models M:\sigma$ and since $\mathcal{M}_\rho \models \Gamma$ we have that $\mathcal{M}_\rho \models M:\sigma$. The latter implies $[\![M]\!]_\rho \in A^\sigma$. By Definition 3.1 we know that $\sigma \le \tau$ implies $A^\sigma \subseteq A^\tau$. Hence, it holds that $[\![M]\!]_\rho \in A^\tau$, and we can conclude $\mathcal{M}_\rho \models M:\tau$.
- Let $\dfrac{\Gamma \vdash M:\sigma \qquad M =_\eta N}{\Gamma \vdash N:\sigma}$ be the last rule used and $\mathcal{M}_\rho$ a model of Γ. By the induction hypothesis we have $\mathcal{M}_\rho \models M:\sigma$, which implies that $[\![M]\!]_\rho \in A^\sigma$ holds. From $M =_\eta N$, that is $M = N$ is provable in $\mathcal{EQ}^\eta$, and Theorem 4.1 we obtain $[\![M]\!]_\rho = [\![N]\!]_\rho$. Thus, we can conclude $[\![N]\!]_\rho = [\![M]\!]_\rho \in A^\sigma$, that is $\mathcal{M}_\rho \models N:\sigma$.

  This concludes the proof.

□

Remark 4.5. As pointed out in Section 3, we have defined an environment as a total mapping, whereas in Kašterović and Ghilezan (2020) an environment was defined as a partial mapping. Every combinatory term is typable with the type ω, i.e., $\Gamma \vdash M:\omega$ for any basis Γ, and in order to obtain the soundness of the type-assignment system we need to prove that $M:\omega$ holds in every model. Thus, we had to define environments so that the meaning of every term M is defined in all models. This is specific to intersection type systems and is due to the rule (axiom ω) which ensures typability of all terms.

We define a class of canonical models, following the approach used in Kašterović and Ghilezan (2020), in order to prove the completeness of the type-assignment system $CL_\cap$ with respect to the models presented in Section 3. Each canonical model is defined with respect to some basis, so that it satisfies only statements that can be derived from that basis. More precisely, we will define a model $\mathcal{M}^\Gamma_{\rho^\star}$ such that

$$\mathcal{M}^\Gamma_{\rho^\star} \models M:\sigma \text{ if and only if } \Gamma \vdash M:\sigma.$$

Definition 4.6. Let Γ be a basis. A canonical model is a pair $\mathcal{M}^{\Gamma}_{\rho^\star} = \langle \mathcal{M}^{\Gamma}, \rho^\star \rangle$, where

$$\mathcal{M}^{\Gamma} = \langle D, \{A^\sigma\}, App \rangle$$

such that

- $D = \{[M] \mid M \in CL\}$,
- $A^\sigma = \{[M] \mid M \in CL \text{ and } \Gamma \vdash M : \sigma\}$,
- $App([M],[N]) = [MN]$,

and $\rho^\star(x) = [x]$.

Lemma 4.7. *The canonical model $\mathcal{M}^{\Gamma}_{\rho^\star}$ is a model for $CL_\cap$.*

Proof: We need to prove that the tuple $\mathcal{M}^{\Gamma}_{\rho^\star}$ introduced in Definition 4.6 satisfies the conditions of Definition 3.5. First, we prove that $\mathcal{M}^{\Gamma} = \langle D, \{A^\sigma\}, App \rangle$ is an applicative structure. The set D is a non-empty set. The set $A^\sigma = \{[M] \mid M \in CL \text{ and } \Gamma \vdash M : \sigma\}$ is a subset of $D = \{[M] \mid M \in CL\}$. For every CL-term M we have $\Gamma \vdash M : \omega$, thus $A^\omega = \{[M] \mid M \in CL \text{ and } \Gamma \vdash M : \omega\} = \{[M] \mid M \in CL\} = D$. The proofs that the family $\{A^\sigma\}$ and the application function App satisfy the conditions of Definition 3.1 are obtained similarly as in the proof of Theorem 4.3.

We have proved that the tuple $\mathcal{M}^{\Gamma} = \langle D, \{A^\sigma\}, App \rangle$ is an applicative structure. Next, we prove that it has combinators and that it is extensional. Similarly as in the proof of Theorem 4.3 we denote equivalence classes $[\mathsf{S}], [\mathsf{K}], [\mathsf{I}]$ by $\mathbf{s}, \mathbf{k}, \mathbf{i}$, respectively. From the type-assignment system (**Figure 2**) we know that for the basis Γ, $\Gamma \vdash \mathsf{S} : (\sigma \to (\tau \to \rho)) \to ((\sigma \to \tau) \to (\sigma \to \rho))$. From the latter it follows that $\mathbf{s} = [\mathsf{S}] \in A^{(\sigma \to (\tau \to \rho)) \to ((\sigma \to \tau) \to (\sigma \to \rho))}$. Similarly, we obtain $\mathbf{k} = [\mathsf{K}] \in A^{\sigma \to (\tau \to \sigma)}$ and $\mathbf{i} = [\mathsf{I}] \in A^{\sigma \to \sigma}$. The proof that elements $[\mathsf{S}], [\mathsf{K}], [\mathsf{I}]$ satisfy Equations (1)–(3) is given in the proof of Theorem 4.3.

We have proved that the applicative structure introduced in Definition 4.6 has combinators. The proof that the applicative structure is extensional is the same as in the proof of Theorem 4.3.

The mapping $\rho^\star$ defined by $\rho^\star(x) = [x]$ is an environment for the applicative structure $\mathcal{M}^{\Gamma}$.

This completes the proof that $\mathcal{M}^{\Gamma}_{\rho^\star}$ is a model for $CL_\cap$. □

Lemma 4.8. *Let $\mathcal{M}^{\Gamma}_{\rho^\star}$ be the canonical model. For every CL-term M, it holds $[\![M]\!]_{\rho^\star} = [M]$.*

Proof: This is a straightforward consequence of Lemma 4.2. □

Theorem 4.9. *Let $\mathcal{M}^{\Gamma}_{\rho^\star}$ be a canonical model. It holds that*

$$\mathcal{M}^{\Gamma}_{\rho^\star} \models M : \sigma \text{ if and only if } \Gamma \vdash M : \sigma.$$

Proof: By Definition 4.6 and Lemma 4.8 we obtain

$$\begin{aligned} \mathcal{M}^{\Gamma}_{\rho^\star} \models M : \sigma &\text{ if and only if } [\![M]\!]_{\rho^\star} \in A^\sigma \\ &\text{if and only if } [M] \in \{[N] \mid N \in CL \text{ and } \Gamma \vdash N : \sigma\} \\ &\text{if and only if } \Gamma \vdash M : \sigma. \end{aligned}$$

□

Theorem 4.10 (Completeness of $CL_\cap$). *If $\Gamma \models M : \sigma$, then $\Gamma \vdash M : \sigma$.*

Proof: Suppose that $\Gamma \models M : \sigma$. Let $\mathcal{M}^{\Gamma}_{\rho^\star}$ be a canonical model (Definition 4.6). First, we need to prove that $\mathcal{M}^{\Gamma}_{\rho^\star} \models \Gamma$. Let $x : \sigma$ be a declaration from the basis Γ. By Lemma 4.8 and Definition 4.6 we have

$$[\![x]\!]_{\rho^\star} = [x] \in \{[N] \mid N \in CL \text{ and } \Gamma \vdash N : \sigma\} = A^\sigma$$

It follows that $\mathcal{M}^{\Gamma}_{\rho^\star} \models \Gamma$. From the latter and the assumption $\Gamma \models M : \sigma$ we obtain $\mathcal{M}^{\Gamma}_{\rho^\star} \models M : \sigma$. By Theorem 4.9 we obtain $\Gamma \vdash M : \sigma$. □

Theorems 4.4 and 4.10 prove that the type-assignment system $CL_\cap$ is both sound and complete with respect to the proposed semantics. Hence the proposed semantics is a semantics of the combinatory logic with intersection types and of the untyped combinatory logic, as proven in Section 4.1.

5. DISCUSSION AND CONCLUSION

Combinatory logic, both typed and untyped, found its application in different scientific fields of computer science, e.g., machine learning (Liang et al., 2010) and artificial intelligence (e.g., Garrette et al., 2015), cognitive representation (e.g., Pierre Desclés, 2004), program synthesis (e.g., Düdder et al., 2012). The development of these fields urge for the further development of combinatory logic, typed, and untyped. Combinatory logic with intersection types has been the object of several studies (Dezani-Ciancaglini and Hindley, 1992; Bunder, 2002; Rehof and Urzyczyn, 2011; Bimbó, 2012). In Dezani-Ciancaglini and Hindley (1992), two different formulations of intersection types for combinatory logic are proposed and we consider in the paper one of these formulations. The authors proved that type-assignment statements are preserved by combinatory β-equality. A logic of intersection types is considered in Venneri (1994), by studying a logical characterization for the intersection-type discipline. The author presents a new formulation of the intersection type inference, equivalent to the original one, and then define a Hilbert-style axiomatization such that a formula is provable in the logical system if and only if it is an inhabited intersection type. Bunder (2002) also studies a logic of intersection types for lambda-terms and combinators by introducing a new system for intersection types, weaker than the one of Venneri (1994), and with the advantage that the logic of types can be obtained directly from the rules of the type-assignment system. None of these papers consider semantics for combinatory logic with intersection types. Models for combinatory logic with intersection types are presented in Bimbó (2012) along with the proof that the type-assignment system is sound and complete with respect to the presented models. Kripke-style semantics for lambda calculus and combinatory logic with types are introduced in Mitchell and Moggi (1991) and Kašterović and Ghilezan

(2020). In Mitchell and Moggi (1991), calculi with simple types are considered, whereas in Kašterović and Ghilezan (2020) the authors considered full simply typed calculi. However, to the best of our knowledge the semantics we propose is novel.

We have introduced a semantics for combinatory logic with intersection types. As the main results of the paper we have proved that both the extensional equational theory of the untyped *CL* and the intersection type-assignment system $CL_{\cap}$ are sound and complete with respect to the proposed semantics.

The usual approach to the semantics for calculi with types that can be found in the literature, e.g., filter models for lambda calculus with intersection types (Barendregt et al., 1983), is based on a model for the untyped calculus endowed with a valuation of type variables which enables the interpretation of types to be defined inductively. We propose a different approach. In the semantics we propose, the interpretation of types is represented as a family of subsets that satisfies certain properties. Moreover, for a given valuation of term variables, the interpretation of terms is defined inductively, whereas in the previously mentioned semantics the interpretation of terms is defined as a function that satisfies specified conditions.

We plan to further develop this approach to lambda calculus, to different frameworks in logic and computation, e.g., polymorphic types.

DATA AVAILABILITY STATEMENT

The original contributions presented in the study are included in the article/supplementary materials, further inquiries can be directed to the corresponding author/s.

AUTHOR CONTRIBUTIONS

Both authors listed have made a substantial, direct, and intellectual contribution to the work and approved it for publication.

FUNDING

This research reported in the paper is partly supported by the Science Fund Republic of Serbia #6526707 AI4TrustBC.

ACKNOWLEDGMENTS

The authors would like to thank the referees whose suggestions lead to major improvements.

REFERENCES

Barbanera, F., Dezani-Ciancaglini, M., and de'Liguoro, U. (1995). Intersection and union types: syntax and semantics. *Inf. Comput*. 119, 202–230. doi: 10.1006/inco.1995.1086

Barendregt, H. P. (1985). *The Lambda Calculus - Its Syntax and Semantics, Volume 103 of Studies in Logic and the Foundations of Mathematics*. Elsevier: North-Holland.

Barendregt, H. P., Coppo, M., and Dezani-Ciancaglini, M. (1983). A filter lambda model and the completeness of type assignment. *J. Symb. Logic*. 48, 931–940. doi: 10.2307/2273659

Barendregt, H. P., Dekkers, W., and Statman, R. (2013). *Lambda Calculus With Types. Perspectives in Logic*. Cambridge: University Press. doi: 10.1017/CBO9781139032636

Bimbó, K. (2012). *Combinatory Logic: Pure, Applied, and Typed*. Boca Raton, FL: CRC Press; Taylor & Francis Group. doi: 10.1201/b11046

Bunder, M. W. (2002). Intersection types for lambda-terms and combinators and their logics. *Log. J. IGPL* 10, 357–378. doi: 10.1093/jigpal/10.4.357

Church, A. (1936). An unsolvable problem of elementary number theory. *Am. J. Math*. 58, 345–363. doi: 10.2307/2371045

Coppo, M., and Dezani-Ciancaglini, M. (1978). A new type assignment for λ-terms. *Arch. Math. Log*. 19, 139–156. doi: 10.1007/BF02011875

Coppo, M., Dezani-Ciancaglini, M., and Venneri, B. (1980). "Principal type schemes and lambda-calculus semantics," in *H.B.Curry: Essays on Combinatory Logic*, Lambda-Calculus and Formalism (San Diego, CA: Academic Press), 535–560.

Curry, H. B. (1930). Grundlagen der kombinatorischen logik. *Am. J. Math*. 52, 509–536. doi: 10.2307/2370619

de Carvalho, D. (2018). Execution time of λ-terms *via* denotational semantics and intersection types. *Math. Struct. Comput. Sci*. 28, 1169–1203. doi: 10.1017/S0960129516000396

de'Liguoro, U., and Treglia, R. (2019). "Intersection types for the computational lambda-calculus," in *Proceedings of the 20th Italian Conference on Theoretical Computer Science, ICTCS 2019*, eds A. Cherubini, N. Sabadini, and S. Tini (Como), 184–189.

Dezani-Ciancaglini, M., and Hindley, J. R. (1992). Intersection types for combinatory logic. *Theor. Comput. Sci*. 100, 303–324. doi: 10.1016/0304-3975(92)90306-Z

Dezani-Ciancaglini, M., and Margaria, I. (1984). "F-semantics for intersection type discipline," in *Semantics of Data Types, International Symposium Sophia-Antipolis, Vol. 173 of Lecture Notes in Computer Science*, eds G. Kahn, D. B. MacQueen, and G. D. Plotkin (Berlin; Heidelberg: Springer), 279–300. doi: 10.1007/3-540-13346-1_14

Düdder, B., Martens, M., Rehof, J., and Urzyczyn, P. (2012). "Bounded combinatory logic," in *Computer Science Logic (CSL'12) - 26th International Workshop/21st Annual Conference of the EACSL, CSL 2012, Vol. 16 of LIPIcs*, eds P. Cégielski and A. Durand (Fontainebleau: Schloss Dagstuhl - Leibniz-Zentrum für Informatik), 243–258.

Garrette, D., Dyer, C., Baldridge, J., and Smith, N. A. (2015). "Weakly-supervised grammar-informed Bayesian CCG parser learning," in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, eds B. Bonet and S. Koenig (Austin, TX: AAAI Press), 2246–2252.

Hindley, J. R., and Seldin, J. P. (1986). *Introduction to Combinators and Lambda-Calculus*. Cambridge: Cambridge University Press.

Kašterović, S., and Ghilezan, S. (2020). Kripke-style semantics and completeness for full simply typed lambda calculus. *J. Log. Comput*. 30, 1567–1608. doi: 10.1093/logcom/exaa055

Liang, P., Jordan, M. I., and Klein, D. (2010). "Learning programs: a hierarchical Bayesian approach," in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, eds J. Fürnkranz and T. Joachims (Haifa: Omnipress), 639–646.

Mitchell, J. C., and Moggi, E. (1991). Kripke-style models for typed lambda calculus. *Ann Pure Appl. Logic* 51, 99–124. doi: 10.1016/0168-0072(91)90067-V

Ong, C. L., and Tsukada, T. (2012). "Two-level game semantics, intersection types, and recursion schemes," in *Automata, Languages, and Programming - 39th International Colloquium, ICALP 2012, Vol. 7392 of Lecture Notes in Computer Science*, eds A. Czumaj, K. Mehlhorn, A. M. Pitts, and R. Wattenhofer (Warwick: Springer), 325–336. doi: 10.1007/978-3-642-31585-5_31

Pierre Desclés, J. (2004). "Combinatory logic, language, and cognitive representations," in *Alternative Logics. Do Sciences Need Them?*, ed P. Weingartner (Berlin: Springer Verlag), 115–148. doi: 10.1007/978-3-662-05679-0_9

Pottinger, G. (1980). "A type assignment for the strongly normalizable λ-terms," in *To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus and Formalism*, eds J. P. Seldin and J. R. Hindley (London: Academic Press), 561–577.

Rehof, J., and Urzyczyn, P. (2011). "Finite combinatory logic with intersection types," in *Typed Lambda Calculi and Applications - 10th International Conference, TLCA 2011, Volume 6690 of Lecture Notes in Computer Science*, ed C. L. Ong (Novi Sad: Springer), 169–183. doi: 10.1007/978-3-642-21691-6_15

Sallé, P. (1978). "Une extension de la théorie des types en lambda-calcul," in *5th International Conference on Automata, Languages and Programming, ICALP'78, Vol. 62 of Lecture Notes in Computer Science*, eds G. Ausiello and C. Böhm (Udine: Springer), 398–410. doi: 10.1007/3-540-08860-1_30

Schönfinkel, M. (1924). Über die bausteine der mathematischen logik. *Math. Ann.* 92, 305–316. doi: 10.1007/BF01448013

van Bakel, S., Barbanera, F., and de'Liguoro, U. (2018). Intersection types for the lambda-mu calculus. *arXiv:1704.00272*. doi: 10.48550/arXiv.1704.00272

Venneri, B. (1994). Intersection types as logical formulae. *J. Log. Comput.* 4, 109–124. doi: 10.1093/logcom/4.2.109

Wolfram, S. (2021). Combinators: a centennial view. *CoRR, abs/2103.12811*.

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.