자연 연역 체계의 전산화에 관한 연구

이상길*

요 약

우리는 엄밀하면서도 읽고 쓰기 쉬운 증명을 작성할 수 있도록 자연 연역 체계를 전산화하였으며, 이를 위해 단순 타입 람다 계산법에 기반하여 형식 언어를 구축하였다. 우리의 자연 연역 체계는 증명 체계의 동작 방식을 묘사할 수 있는 능력을 가지며, 이는 사용자가 형식 언어 및 공리, 추론 규칙을 자유롭게 정할 수 있도록 한다. 우리는 증명 체계를 만들고, 만들어진 증명 체계에서 컴퓨터에 의해 검증되는 증명을 할 수 있도록 하는 웹 기반 컴퓨터 프로그램 `math-o-matic`을 만들었으며, Morse–Kelly 집합론을 기반으로 하여 이항관계 및 함수, 자연수 집합, 정수 집합에 관한 개념을 정의하고 관련 정리를 증명할 수 있었다.

주제어: 수학기초론, 형식주의, 람다 계산법, 자연 연역 체계

1 서론

수학적 형식주의의 관점에서 모든 수학적 정의 및 정리는 어떤 증명 체계(proof system) 상에서 표현할 수 있으며, 곧 증명 체계는 수학 전체를 표현할 수 있는 능력을 갖는다. 이때 증명 체계 상에서 이루어지는 증명은 그 형식적 특성에 의하여 컴퓨터에 의해 기계적으로 검증될 수 있으며, 이는 증명 과정에서 발생할 수 있는 실수를 배제한다. 그러므로 증명 체계를 전산화하는 것이 필요하며, 우리는 엄밀하면서도 읽고 쓰기 쉬운 증명을 작성할 수 있도록 가언적 추론(假言的推論, hypothetical derivation)을 지원하는 자연 연역 체계를 전산화하기로 하였다.

증명 체계를 만들기 위해서는 먼저 증명 체계가 사용할 형식 언어를 만들어야 한다. 이때 잘못된 문장을 구성하는 것을 방지하기 위하여 언어 상의 모든 용어는 타입을 갖도록 할 것인데, 예를 들어 "$1+1=2$"라는 명제는 명제 타입 Prop을 가질 것이고, 연언(連言)을 뜻하는 이항연산자 $\wedge$는 두 개의 명제를 받아 새로운 명제를 만드는 함수이므로 $\mathsf{Prop}\times\mathsf{Prop}\to\mathsf{Prop}$ 타입을 가질 것이다. 이때 p 및 q가 명제인 경우 $p\wedge q$는 이항연산자 $\wedge$에 인자 (p,q)를 주어 호출한 것이며 $p\wedge q$의 타입은 $\wedge$의 반환 타입인 Prop이 된다. 예시로부터 알 수 있듯 우리의 언어는 함수의 생성 및 호출에 관한 구문을 지원하여야 하며, 곧 특정 조건을 만족하는 람다 항의 집합이 되어야 한다. 우리는 단순 타입 람다 계산법(simply typed lambda calculus)에 기반하여 언어를 구축할 것이다.

또 우리의 체계는 형식 언어 및 공리, 추론 규칙을 미리 정해두지 않고 우리의 체계를 활용하려는 자가 임의로 정할 수 있도록 하려고 하며, 이를 위하여 우리의 언어는 추론 규칙에 대한 표현력을 가진다. 예를 들어 Hilbert 체계의 대표적 추론 규칙인 함의 소거(implication elimination) $p, p\to q\vdash q$는 $\vdash$를 포함하는 메타문장이므로 대상언어에 속하지 않으나, 우리의 체계는 이를 언어에 포함시킨다. 그러므로 우리의 언어를 구성하기 위해서는 단순 타입 람다 계산법에 $\vdash$ 구문을 추가할 필요가 있다. 이때 우리의 증명 체계는 $\vdash$ 구문이 추론 규칙처럼 작동할 수 있도록 하는 추론 규칙들을 가진다.

문제는 $\vdash$ 구문을 추가하여도 함의 소거가 메타변항(metavariable) p 및 q를 포함하는 스키마(schema)이므로 대상언어에 속할 수 없다는 것이다. 이를 해결하기 위하여 우리는 스키마를 메타변항들을 매개변항으로 하는 함수라고 생각하여 메타변항들을 종속시킬 것이다. 예를 들어 함의 소거는 명제 p 및 q를 받아 $p, p\to q\vdash q$를 반환하는 함수가 된다. 이때 우리의 증명 체계는 함수로 표현한 스키마들이 스키마처럼 작동할 수 있도록 하는 추론 규칙들을 가질 것이다.

즉 우리의 체계는 증명 체계의 동작 방식을 묘사하는 증명 체계이다. 또 우리의 체계는 람다 계산법을 활용한 자연 연역 체계임에도 불구하고 람다 계산법의 타입 체계와 자연 연역 체계 간의 Curry–Howard 대응[1]에 의존하지 않는다. 이는 수식을 타입이 아니라 람다 항이라 생각하기 때문이며, 이로 인해 어떤 정리의 증명 가능성과 type inhabitation 문제가 대응된다는 해석을 받아들이지 않아도 된다는 점에서 더 직관적이다.

*ossia@korea.ac.kr

2 형식 언어의 정의

$\boldsymbol{\lambda}^{\mathrm{Ch}}_{\to}$가 [2]의 정의 1.1.30과 같이 정의된다고 하자. $\boldsymbol{\lambda}^{\mathrm{Ch}}_{\to}$에서는 모든 변항(變項)에 고유한 타입이 지정되어 있어서 타이핑 문맥(typing context)이 필요하지 않다. 이제 $\boldsymbol{\lambda}^{\mathrm{Ch}}_{\to}$에 $\vdash$ 구문을 추가하여 새로운 체계 $\boldsymbol{\lambda}^{\mathrm{Ch}}_{\to,\vdash}$를 만들어 보자. 먼저 타입 생성자 $\vdash$를 추가하기 위하여 [2]의 정의 1.1.11의 BNF를 다음과 같이 수정하자.

$$\mathbb{T} ::= \mathbb{A} \mid \mathbb{T} \to \mathbb{T} \mid \mathbb{T} \vdash \mathbb{T}.$$

또 $M \vdash N$ 형태의 람다 항이 생성될 수 있도록 [2]의 정의 1.1.30(ii)에 다음과 같은 규칙을 추가하자.

$$M \in \Lambda^{\mathrm{Ch}}_{\to,\vdash}(A), N \in \Lambda^{\mathrm{Ch}}_{\to,\vdash}(B) \Rightarrow (M \vdash N) \in \Lambda^{\mathrm{Ch}}_{\to,\vdash}(A \vdash B).$$

$\vdash$는 오른쪽 결합성을 갖는다고 하자. 즉 $p \vdash q \vdash r$는 $p \vdash (q \vdash r)$를 뜻한다. $\vdash$ 구문이 전건을 하나만 가질 수 있도록 하는 것은 추상화 구문에서 매개변항을 하나만 지정할 수 있는 것과 같은 이유로, $\vdash$ 구문에 커링(currying)을 적용하여

$$p_1, p_2, \ldots, p_n \vdash q$$

를

$$p_1 \vdash p_2 \vdash \cdots \vdash p_n \vdash q$$

로 표현할 수 있기 때문이다.

또 우리 체계는 함수의 외연성(extensionality)을 받아들여 $\boldsymbol{\lambda\beta\eta}$를 equational theory로 사용할 것인데, $\vdash$에 대한 합동을 고려하기 위하여 규칙

$$\frac{M = M' \quad N = N'}{(M \vdash N) = (M' \vdash N')}$$

을 추가하여야 한다. 이때 Church식 의미론에 의하여 $A \neq B$일 때 $\lambda x^A.x^A \neq \lambda x^B.x^B$이다. 이제 $\beta\eta$ 동치인 두 람다 항은 서로 같다고 볼 것이다.

또 $\wedge$나 $\forall$ 등의 무정의용어를 도입하기 위해서는 정항(正項)을 추가할 수 있어야 하는데, 정항의 집합 $\mathcal{C}$는 [2]의 정의 1.1.30(i)에서 정의된 변항의 집합 $\mathsf{V}^{\mathbb{T}}$의 부분집합으로 정의하자. 이때 우리의 형식 언어 $\mathcal{L}$은 다음과 같이 정의된다.

정의 1 (형식 언어)**.** $\mathbb{A}$가 원시 타입의 집합이고 $\mathbb{T} = \mathbb{T}^{\mathbb{A}}$일 때, 정항의 집합 $\mathcal{C} \subseteq \mathsf{V}^{\mathbb{T}}$에 대하여 형식 언어 $\mathcal{L}$이 다음과 같이 정의된다.

$$\mathcal{L} = \{M \in \Lambda^{\mathrm{Ch}}_{\to,\vdash} : \mathrm{FV}(M) \subseteq \mathcal{C}\}.$$

즉 $\mathcal{L}$은 $\Lambda^{\mathrm{Ch}}_{\to,\vdash}$에 있는 람다 항 중에서 자유 변항이 전부 정항인 것들의 집합이다.

3 공리 및 추론 규칙의 정의

가정의 집합 $\Gamma \subseteq \Lambda^{\mathrm{Ch}}_{\to,\vdash}$ 및 람다 항 $\varphi \in \Lambda^{\mathrm{Ch}}_{\to,\vdash}$에 대해 시퀀트(sequent)가 $\Gamma \Rightarrow \varphi$ 형태의 식을 뜻한다고 하자. $\Gamma \Rightarrow \varphi$는 "가정 Γ로부터 φ를 증명할 수 있다"는 뜻이라 생각할 수 있다. 우리의 증명 체계는 증명 체계의 동작을 모사하는 5개의 추론 규칙을 가진다.

정의 2 (공리 및 추론 규칙)**.** 우리 증명 체계의 공리 및 추론 규칙은 다음과 같다.

1. (R 규칙) 임의의 $\varphi \in \Gamma$에 대하여 $\Gamma \Rightarrow \varphi$.
2. ($\vdash$E 규칙) $\dfrac{\Gamma \Rightarrow \varphi \quad \Gamma \Rightarrow (\varphi \vdash \psi)}{\Gamma \Rightarrow \psi}$.
3. ($\vdash$I 규칙) $\dfrac{\Gamma \cup \{\varphi\} \Rightarrow \psi}{\Gamma \Rightarrow (\varphi \vdash \psi)}$.
4. ($\mapsto$E 규칙) $\varphi\psi \in \Lambda^{\mathrm{Ch}}_{\to,\vdash}$일 때
$$\frac{\Gamma \Rightarrow \varphi}{\Gamma \Rightarrow \varphi\psi}.$$
5. ($\mapsto$I 규칙) $x^A \notin \mathcal{C}$일 때
$$\frac{\Gamma \Rightarrow \varphi}{\Gamma \Rightarrow \lambda x^A.\varphi}.$$

공리 및 추론 규칙이 모두 시퀀트에 관한 것인 이유는 우리 체계가 가언적 추론을 지원하는 자연 연역 체계이기 때문이다. 이때 $\vdash$E 및 $\vdash$I 규칙에 의하여 $\vdash$ 구문이 추론 규칙처럼 작동하며 $\mapsto$E 및 $\mapsto$I 규칙에 의하여 함수로 표현한 스키마들이 스키마처럼 작동한다. 특히 $\vdash$I 규칙은 유도 가능한 규칙(derivable rule)을 유도할 수 있도록 하며, 이로 인하여 우리 체계가 가언적 추론을 지원하게 된다.

예시 3.

$$\Gamma = \left\{ \begin{array}{l} \lambda p^{\mathsf{Prop}} \lambda q^{\mathsf{Prop}}.(p \vdash (q \vdash \wedge pq)), \\ \lambda p^{\mathsf{Prop}} \lambda q^{\mathsf{Prop}}.(\wedge pq \vdash p), \\ \lambda p^{\mathsf{Prop}} \lambda q^{\mathsf{Prop}}.(\wedge pq \vdash q) \end{array} \right\}$$

일 때

$$\Gamma \Rightarrow \lambda p^{\mathsf{Prop}} \lambda q^{\mathsf{Prop}}.(\wedge pq \vdash \wedge qp)$$

는 그림 1과 같이 증명할 수 있으며, 이는 Fitch 다이어그램으로 그림 2와 같이 더 간단히 표현할 수 있다. 단 그림 2에서 연속된 $\mapsto$E 또는 $\vdash$E, $\mapsto$I는 하나로 표현되었다.

1	$\Gamma \cup \{\wedge p'q'\} \Rightarrow \wedge p'q'$	R
2	$\Gamma \cup \{\wedge p'q'\} \Rightarrow \lambda p \lambda q.(\wedge pq \vdash p)$	R
3	$\Gamma \cup \{\wedge p'q'\} \Rightarrow \lambda q.(\wedge p'q \vdash p')$	$\mapsto$E (2)
4	$\Gamma \cup \{\wedge p'q'\} \Rightarrow (\wedge p'q' \vdash p')$	$\mapsto$E (3)
5	$\Gamma \cup \{\wedge p'q'\} \Rightarrow p'$	$\vdash$E (1, 4)
6	$\Gamma \cup \{\wedge p'q'\} \Rightarrow \lambda p \lambda q.(\wedge pq \vdash q)$	R
7	$\Gamma \cup \{\wedge p'q'\} \Rightarrow \lambda q.(\wedge p'q \vdash q)$	$\mapsto$E (6)
8	$\Gamma \cup \{\wedge p'q'\} \Rightarrow (\wedge p'q' \vdash q')$	$\mapsto$E (7)
9	$\Gamma \cup \{\wedge p'q'\} \Rightarrow q'$	$\vdash$E (1, 8)
10	$\Gamma \cup \{\wedge p'q'\} \Rightarrow \lambda p \lambda q.(p \vdash (q \vdash \wedge pq))$	R
11	$\Gamma \cup \{\wedge p'q'\} \Rightarrow \lambda q.(q' \vdash (q \vdash \wedge q'q))$	$\mapsto$E (10)
12	$\Gamma \cup \{\wedge p'q'\} \Rightarrow (q' \vdash (p' \vdash \wedge q'p'))$	$\mapsto$E (11)
13	$\Gamma \cup \{\wedge p'q'\} \Rightarrow (p' \vdash \wedge q'p')$	$\vdash$E (9, 12)
14	$\Gamma \cup \{\wedge p'q'\} \Rightarrow \wedge q'p'$	$\vdash$E (5, 13)
15	$\Gamma \Rightarrow (\wedge p'q' \vdash \wedge q'p')$	$\vdash$I (14)
16	$\Gamma \Rightarrow \lambda q'.(\wedge p'q' \vdash \wedge q'p')$	$\mapsto$I (15)
17	$\Gamma \Rightarrow \lambda p' \lambda q'.(\wedge p'q' \vdash \wedge q'p')$	$\mapsto$I (16)
	$= \lambda p \lambda q.(\wedge pq \vdash \wedge qp)$	α 동치

그림 1: $\Gamma \Rightarrow \lambda p^{\mathsf{Prop}} \lambda q^{\mathsf{Prop}}.(\wedge pq \vdash \wedge qp)$의 증명

4 증명 체계의 정의

증명 체계는 원시 타입의 집합 $\mathbb{A}$, 정항의 집합 $\mathcal{C} \subseteq \mathsf{V}^{\mathbb{T}}$, 공리의 집합 $\Gamma_0 \subseteq \mathcal{L}$에 의해 결정된다. 이때 어떤 명제 $\varphi \in \mathcal{L}$이 체계 $\langle \mathbb{A}, \mathcal{C}, \Gamma_0 \rangle$ 상에서 증명 가능하다는 것은 다음과 같은 뜻이다.

정의 4 (증명 가능성)**.** $\langle \mathbb{A}, \mathcal{C}, \Gamma_0 \rangle$ 상에서 $\varphi \in \mathcal{L}$이 증명 가능하다는 것은 $\Gamma_0 \Rightarrow \varphi$가 유도 가능하다는 뜻이다.

이제부터 함수 타입 및 $\vdash$ 타입을 언커링(uncurrying) 하여 표시하자. $\lambda x^A \ldots \lambda z^C.M$은 $(x^A, \ldots, z^C) \mapsto M$이라 쓰자. 또 fxy 대신 $f(x, y)$라 쓰고 $p \vdash q \vdash r$ 대신 $p, q \vdash r$라 쓰자. $\wedge$ 등의 이항연산자의 경우 $\wedge(p, q)$ 대신 $p \wedge q$라고도 쓸 수 있다.

예시 5. 명제논리를 위한 증명 체계는 다음과 같이 정의해 볼 수 있다.

$$\mathbb{A} = \{\mathsf{Prop}\},$$
$$\mathcal{C} = \{\to, \neg, \wedge, \top\},$$
$$\Gamma_0 = \left\{ \begin{array}{l} (p^{\mathsf{Prop}}, q^{\mathsf{Prop}}) \mapsto ((p \vdash q) \vdash p \to q)\,, \\ (p^{\mathsf{Prop}}, q^{\mathsf{Prop}}) \mapsto (p, p \to q \vdash q)\,, \\ (p^{\mathsf{Prop}}, q^{\mathsf{Prop}}) \mapsto (p, q \vdash p \wedge q)\,, \\ (p^{\mathsf{Prop}}, q^{\mathsf{Prop}}) \mapsto (p \wedge q \vdash p)\,, \\ (p^{\mathsf{Prop}}, q^{\mathsf{Prop}}) \mapsto (p \wedge q \vdash q)\,, \\ \top, \\ (p^{\mathsf{Prop}}, q^{\mathsf{Prop}}) \mapsto (\neg p \to \neg q \vdash q \to p) \end{array} \right\}.$$

1	p'	q'	$\wedge p'q'$	가정
2			$\lambda p \lambda q.(\wedge pq \vdash p)$	R
3			$\wedge p'q' \vdash p'$	$\mapsto$E (2)
4			p'	$\vdash$E (1, 3)
5			$\lambda p \lambda q.(\wedge pq \vdash q)$	R
6			$\wedge p'q' \vdash q'$	$\mapsto$E (5)
7			q'	$\vdash$E (1, 6)
8			$\lambda p \lambda q.(p \vdash (q \vdash \wedge pq))$	R
9			$q' \vdash (p' \vdash \wedge q'p')$	$\mapsto$E (8)
10			$\wedge q'p'$	$\vdash$E (7, 4, 9)
11			$\wedge p'q' \vdash \wedge q'p'$	$\vdash$I (1–10)
12	$\lambda p \lambda q.(\wedge pq \vdash \wedge qp)$			$\mapsto$I (1–11)

그림 2: Fitch 다이어그램으로 표현한 증명

4.1 매크로의 정의

예시 5의 체계에는 $\vee$, $\leftrightarrow$ 연산자 및 $\bot$도 정의되어 있는데 이들은 무정의용어로서가 아닌 다른 개념에 의존하는 매크로로서 정의되어 있으므로 무정의용어의 집합인 $\mathcal{C}$에 나타나지 않는다. 이들은 다음과 같이 정의되어 있다.

$$\vee \coloneqq (p^{\mathsf{Prop}}, q^{\mathsf{Prop}}) \mapsto (\neg p \to q)\,,$$
$$\leftrightarrow \coloneqq (p^{\mathsf{Prop}}, q^{\mathsf{Prop}}) \mapsto ((p \to q) \wedge (q \to p))\,,$$
$$\bot \coloneqq \neg \top.$$

또 Morse–Kelley 집합론에서 단항 술어는 클래스 하나를 받아 명제를 출력하는 함수이므로 $\mathsf{Cls} \to \mathsf{Prop}$ 타입일 것이며, 이 역시 매크로로서

$$\mathsf{Predicate} \coloneqq \mathsf{Cls} \to \mathsf{Prop}$$

이라 정의할 수 있다.

5 `math-o-matic`: 전산화된 증명 체계

우리는 증명 체계 $\langle \mathbb{A}, \mathcal{C}, \Gamma_0 \rangle$을 만들고, 만들어진 증명 체계에서 컴퓨터에 의해 검증되는 증명을 할 수 있도록 하는 웹 기반 컴퓨터 프로그램을 만들었으며 이름을 `math-o-matic`이라 하였다. `math-o-matic`의 소스 코드 및 `math-o-matic`을 실행할 수 있는 웹 페이지는

`https://github.com/logico-philosophical/math-o-matic`

에서 찾을 수 있다.

math-o-matic은 증명 체계의 정의 및 증명 작성을 위한 기술 언어를 제공한다. 예를 들어 타입 Prop 및 무정의용어 ∧를 정의하고, 예시 3의 Γ를 공리계에 포함시키는 코드는 다음과 같이 작성할 수 있다.

```
type Prop;

Prop A(Prop p, Prop q);

axiom Ai(Prop p, Prop q) {
    p, q |- A(p, q)
}

axiom Ae1(Prop p, Prop q) {
    A(p, q) |- p
}

axiom Ae2(Prop p, Prop q) {
    A(p, q) |- q
}
```

단 A가 ∧를 뜻한다. 이때 그림 2의 증명은 다음과 같이 작성할 수 있다.

```
theorem A_flip(Prop p, Prop q) {
    A(p, q) |- {
        [
            @h1 > Ae2(p, q);
            @h1 > Ae1(p, q)
        ] > Ai(q, p)
    }
}
```

둘째 줄이 가정 A(p, q)를 도입하며 @h1이 이를 가리킨다. Ae2(p, q)는 Ae2를 인자 (p, q)로 호출하므로 넷째 줄은

```
A(p, q) > (A(p, q) |- q)
```

가 되며, > 연산자는 좌변 A(p, q)와 우변의 전건 A(p, q)를 비교하여 이들이 $\beta\eta$ 동치이면 우변의 후건 q를 유도한다. 그러므로 넷째 줄이 q가 되고 같은 방식으로 다섯째 줄이 p가 되므로, 3–6번째 줄은

```
[q; p] > (q, p |- A(q, p))
```

가 되며 이것이 A(q, p)를 유도한다. 그러므로 2–7번째 줄이

```
A(p, q) |- A(q, p)
```

가 되어 $(p^{\mathsf{Prop}}, q^{\mathsf{Prop}}) \mapsto (p \wedge q \vdash q \wedge p)$가 증명된다.

math-o-matic 프로그램은 위의 증명을 검증하고 그림 3과 같이 Fitch 다이어그램으로 표시할 수 있다. 이때 표현의 간결함을 위하여 R, ⊢I, ↦E, ↦I 규칙의 적용은 생략되었으며, 추가적인 코드를 작성하여 A(p, q)가 $p \wedge q$로 표시되도록 하였다.

theorem A_flip(Prop p, Prop q)

A_flip(p, q) :
$p \wedge q \vdash q \wedge p$

proof explorer

#	expression	rule	
1	$p \wedge q$	assumption	
2	q	⊢E	Ae2(p, q) [1]
3	p	⊢E	Ae1(p, q) [1]
4	$q \wedge p$	⊢E	Ai(q, p) [2, 3]

그림 3: A_flip의 증명

6 수학의 전산화

우리는 math-o-matic 체계 상에서 Morse–Kelley 집합론을 기반으로 하여 이항관계 및 함수, 자연수 집합, 정수 집합에 관한 개념을 정의하고 관련 정리들을 증명할 수 있었으며, 이들은 math-o-matic 메인 페이지에서 확인할 수 있다. 현재 $\langle \mathbb{A}, \mathcal{C}, \Gamma_0 \rangle$은 다음과 같이 정의되어 있다.

원시 타입. Prop 및 Cls 두 개가 있다.

무정의용어. 8개의 무정의용어가 있으며, 명제논리를 위한 무정의용어는 예시 5에서와 같이 →, ¬, ∧, ⊤이다. 술어논리를 위한 무정의용어는 ∀ 하나이며, 타입 Predicate → Prop을 갖는다. 이는 $(\forall x)(f(x))$를 $\forall(x \mapsto f(x))$라고 생각한 것이다. 집합론을 위한 무정의용어는 =, ∈ 및 조건제시법(set-builder notation)이 있다.

공리계. 17개의 공리가 정의되어 있으며, 명제논리를 위한 공리계는 예시 5의 Γ_0와 같다. 또 술어논리를 위한 공리는 다음의 두 개 뿐이다.

$$(f^{\mathsf{Predicate}}) \mapsto (f \vdash \forall f),$$
$$(f^{\mathsf{Predicate}}, x^{\mathsf{Cls}}) \mapsto (\forall f \vdash f(x)).$$

이외의 공리는 모두 Morse–Kelley 집합론의 공리를 가져온 것이다.

우리는 위 체계에서 다음을 포함한 500개 가량의 정리를 증명할 수 있었다.

- 수학적 귀납법.
- 재귀 정리(recursion theorem).
- $1 + 1 = 2$.
- Cantor의 정리.
- Schröder–Bernstein 정리.

7 결론 및 향후 연구

우리는 자연 연역 체계의 가언적 추론을 활용하여 Schröder–Bernstein 정리의 증명과 같은 복잡한 증명을 자연스러운 방식으로 전개할 수 있었다. 그러나 우리는 형식적 증명이 훨씬 더 간단히 읽히고 쓰일 수 있을 것이라 예상한다. 현재의 체계에는 다음과 같은 불편함이 있다.

- 그림 3에서처럼 Fitch 다이어그램으로 나타낸 Schröder–Bernstein 정리의 증명은 300줄이 넘음에도 불구하고, 증명의 각 단계를 구분 없이 일렬로 나열하여 증명의 골자를 파악하기 쉽지 않다.
- 현재 증명의 자동화에 대한 지원이 없어 증명의 모든 세부 단계를 일일이 기술하여야 한다는 점이 번거롭다.

또 기능적 측면에서 다음과 같은 문제가 있다.

- 증명 체계 $\langle \mathbb{A}, \mathcal{C}, \Gamma_0 \rangle$을 명시적으로 선언하는 구문이 없어 체계를 하나만 만들 수 있다. 여러 개의 체계를 만들 수 있다면 두 증명 체계 $\mathcal{P}_1 = \langle \mathbb{A}_1, \mathcal{C}_1, \Gamma_1 \rangle$ 및 $\mathcal{P}_2 = \langle \mathbb{A}_2, \mathcal{C}_2, \Gamma_2 \rangle$의 형식 언어 $\mathcal{L}_1$ 및 $\mathcal{L}_2$ 간의 준동형 사상 $f : \mathcal{L}_1 \to \mathcal{L}_2$에 관해 생각해 볼 수 있을 것이다. $\mathcal{T}_i$가 증명 체계 $\mathcal{P}_i$에서 증명할 수 있는 모든 식의 집합을 뜻한다고 하자. 이때 $f[\Gamma_1] \subseteq \mathcal{T}_2$이면 $f[\mathcal{T}_1] \subseteq \mathcal{T}_2$라는 성질을 사용할 수 있을 것이다. 예를 들어 집합론 상에서의 어떤 대상이 Peano 공리계의 공리들을 모두 만족할 때, 그 대상에 Peano 공리계의 임의의 정리를 적용할 수 있도록 하는 기능을 지원할 수 있을 것이다.
- 형식 언어가 단순 타입 람다 계산법에 기반하고 있으므로 가변 길이 시퀀스에 타입을 지정하지 못하며, 이로 인하여 시퀀트 계산법(sequent calculus) 체계를 기술하지 못한다.

이러한 문제들을 해결하는 것이 향후 연구 주제가 될 것이다.

참고문헌

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

[2] H. Barendregt, W. Dekkers, and R. Statman, *Lambda Calculus with Types.* Perspectives in Logic, Cambridge University Press, 2013.