DOR: 20.1001.1.2538161.1402.8.4.6.4
DOI : 10.22034/csj.2024.192416
زمستان ۱۴۰۲، دوره ۸، شماره ۴، پیاپی ۳۱

دریافت مقاله: ۱۴۰۲/۰۹/۰۷
پذیرش مقاله: ۱۴۰۲/۱۱/۱۱

نوع مقاله: ترویجی

تناظر کوری–هاوارد چیست؟*

مجید علی‌زاده**
دانشیار دانشکدگان علوم دانشکده ریاضی، آمار و علوم کامپیوتر- گروه علوم کامپیوتر- دانشگاه تهران- تهران- ایران
پست الکترونیکی: majidalizadeh@ut.ac.ir

چکیده

تناظر کوری- هاوارد نتیجه‌ایست در حوزه منطق و علوم کامپیوتر که ارتباط عمیقی میان برهان‌ها و برنامه‌های کامپیوتری برقرار می‌کند. در این نوشتارِ ترویجی، ایده‌های کلیدی پشت این تناظر، پیامدهای آن برای هر دو حوزهٔ منطق و علوم کامپیوتر، و تأثیر آن بر توسعه زبان‌های برنامه‌نویسی(تابعی) و اثبات‌یارها بررسی می‌شود. تناظر کوری- هاوارد، در ساده‌ترین شکلش، یک تناظر میان نوع‌ها و ترم‌ها از حساب لامبدای نوع‌دار (ساده‌ترین زبان برنامه‌نویسی تابعی) و فرمول‌ها و برهان‌ها در دستگاه استنتاج طبیعی منطق شهودی گزاره‌ای برقرار می‌کند. این تناظر نه تنها یک نتیجه مهم در نظریه برهان است؛ بلکه زیربنایی برای گسترش و توسعهٔ دستگاه‌های نوع‌دار برای زبان‌های برنامه‌نویسی تابعی است. این نوشتار این تناظر را به گونه‌ای معرفی می‌کند که درک آن برای افراد غیرمتخصص (با دانش پایه در مبانی منطق) ممکن باشد.

کلیدواژه‌ها: تناظر کوری- هاروارد، منطق، زبان‌های برنامه‌نویسی تابعی

مقدمه

درحوزه منطق، فرگه[1]را به خاطر نوشتن کتاب «مفهوم نگاشت»[2]در سال ۱۸۷۹ م. خالق منطق صوری جدید می‌نامند. اما نگاه واقع‌بینانه‌تر به تاریخ پیدایش منطق جدید، افراد تاثیرگذار دیگری را نیز در این حوزه معرفتی نشان می‌دهد. خیلی خلاصه این‌که، در اواخر قرن نوزدهم میلادی تقریباً سه مکتب زیر به پیشبرد منطق مدرن کمک کردند. (برای مطالعه بیشتر به [۲] مراجعه کنید.)

۱. سنت جبری با کارهای بول، پیرس، جوونز، شرودر و ون که کوشیدند رابطهٔ میان استدلال و اعمال جبری مانند جمع و ضرب را مشخص کنند. می‌توان گفت کارهای این مکتب انتزاعی کردن مصادیق استدلال با تبدیلشان به اصول منطقی بوده است.

۲. مکتب دوم معروف به منطق‌گرایی است که به نوعی در جهت عکس مکتب نخست بود. پیروان این مکتب اصول منطقی را زیربنای همه تفکرات، از جمله تفکر علمی، می‌دانستند. افرادی چون فرگه، راسل و ویتکنشتاین (اوّل) مروج این مکتب بودند.

۳. در نهایت، ریاضیدانانی بودند که به اصل‌بندی کردن نظام‌های گوناگون ریاضی مانند حساب، هندسه،

1- Ferege
2- Begriffsschrift

* Curry-Howard correspondence

** نویسندهٔ مسئول

نظریـۀ مجموعه‌ها و آنالیز علاقه‌منـد بودند. پژوهش‌های ریاضیدانانی چون پیانـو[۳]، ددکیند،[۴] هیلبـرت،[۵] زرملو[۶] و هیتینگ[۷] در این راستا بوده است.

هـر چند فرگه بـرای ارایۀ یـک مبانی برای حسـاب تلاش‌های خوبی کرد؛ امّا از پاسـخ دادن به اشکال راسل درماند. پارادکس راسل آغازگر مجموعه بحث‌ها و مطالعات طولانی به هدف یافتن راهی برای گذر از این تنازع شد. در همین دوره راسل با همکاری وایتهد نظریه جدیدی را به نام «نظریۀ انضمامی انواع»[۸] در کتاب اصول ریاضیات خویش معرفی و بررسـی کرد. این نظریه با تمام پیچیدگی‌هایش می‌خواست از ایدۀ اولیۀ «استوار بودن ریاضیات یا حداقل حسـاب، به اصول منطقی» دفاع کند. هرچند این نظریه در آن زمـان به دلیل پیچیده بودن و سـاختگی بودن اصول منطقی‌اش چندان مورد اسـتقبال عموم (ریاضیدانان) قرار نگرفت؛ اما بایـد گفت که امروزه نظریۀ نـوع، در طراحی زبان‌های برنامه‌نویسی (تابعی) نقش انکارناپذیری دارد. با تخصیص نـوع به ترم‌های برنامه بهتر می‌توانیم خطاهای برنامـه را بیابیم. تنها کاری که باید انجام دهیم (حتی پیش از اجرای برنامه)، این است که در هر مرحله نوع را بررسی کنیم. بـه عبارتی، می‌توان گفت تناظر کـوری- هاوارد از نظریه انواع راسل و وایتهد نشأت گرفت.

هم زمان با کارهای فرگه، راسـل و وایتهد در پروژه‌ی منطق‌گرایی، افراد دیگری مشـغول پژوهش روی موضوع طبیعـت ریاضیات بودنـد. بـرای نمونه براوئـر[۹] مکتب شـهودگرایی را بـه عنوان یـک مکتب فلسـفی جدید در ریاضیات معرفـی کرد که نه تنها تأثیر بسـزایی در نگاه ریاضیدانان به طبیعت ریاضیات داشت، بلکه در منطق نیز جایگاه ویژه‌ای یافت.

براوئر صدق یک گزارۀ ریاضی را وابسـته به ساختار ذهنی متناظر با اثبات آن گزاره می‌دانسـت. از نگاه براوئر صدق گزاره معادل با اثبات‌پذیری آن گزاره اسـت. با این تفسـیر از مفهوم صـدق، برخی اصول معتبـر در منطق کلاسـیک نامعتبر شدند. بسـیاری از ریاضیدانان از نگاه براوئر به مفهوم صدق به مثابه اثبات‌پذیری، استقبال کردند و ایـن امر به تعبیر منطقی این دیدگاه به نام تعبیر BHK[۱۰] منجر شـد. دسـتگاه منطقی‌ای که مطابق این تعبیر شکل گرفت به منطق شهودی معروف است. منطق شهودی برای منظور ما در این نوشتار بسیار مهم است؛ زیرا قرار است تناظری میان این منطق و حسـاب لامبدای نوع‌دار معرفی کنیم. در واقع ارتباط عمیقی میان این دو مقوله وجود دارد؛ بـرای نمونه تعبیر BHK می‌گوید که یک برهان برای گزاره شـرطی $A \to B$ دقیقاً یک روش برای تبدیل یک برهانِ A به یک برهانِ B اسـت و این دقیقاً همان کاریست که ما در حسـاب لامبدای نوع‌دار، برای خلق یک نوعِ تابعی[۱۱] انجام می‌دهیم.

با وجود این تحولات، همچنان به یک پرسـش پاسـخ دقیقی داده نشد: «برهان چیست؟». هیلبرت نخستین کسی بود که پیشـنهاد داد مفهـوم برهان را، همچـون اعداد و اشکال هندسـی، به عنوان یک شیء ریاضی مطالعه کنیم. به این ترتیب پژوهش‌ها بر روی برهان صوری آغاز شـد و گسترش یافت، تا جایی‌که شاخۀ مهمی از منطق ریاضی با عنوان «نظریه برهان» به مثابه قلب منطق یا منطقِ منطق پدید آمد. هیلبرت می‌خواسـت همچون حساب و هندسه، براهیـن سـازگاری فراریاضیاتی‌ای را بـرای ریاضیات نامتناهی به دست دهد. حاصل تلاش وی به همراه برنیز[۱۲] معرفی حساب اپسیلن[۱۳] به روش اصل‌بندی بود. یک عیب یا محدودیت روش اصل‌بندی برای معرفی دستگاه منطقی این است که نشـان‌دهنده روش استدلال کردن یک ریاضیدان نیست.

لوکاسـویچ[۱۴] پیشـنهاد داد منطقدانـان در پی معرفی روشی باشند که اسـتدلال ریاضی واقعی را نشان دهد و

3- Peano
4-Dedekind
5- Hilbert
6- Zermelo
7- Heyting
8- The ramified theory of types
9- Brouwer

10- Brouwer-Heyting-Kolmogorov
11- Function type
12- Bernays
13- Epsilon calculus
14- Łukasiewicz

همزمان همان قضایایی را تولید کند که روش اصل‌بندی تولید می‌کند. برای رسیدن به این هدف، جاسکوسکی[۱۵] دو دستگاه استنتاج طبیعی معرفی کرد. همزمان گنتزن[۱۶] دستگاه‌های استنتاج طبیعی‌ایی را معرفی کرد که در آنها مفروضات نقش مرکزی داشتند. با معرفی این دستگاه‌ها به یکباره پیشرفت شگرفی در نظریه برهان پدید آمد.

شوون فینکل[۱۷]، یک دهه پیش از معرفی و توسعه دستگاه استنتاج طبیعی، نخستین کسی بود که منطق ترکیبی[۱۸] را معرفی کرد: دستگاه صوری‌ای که هدفش توصیف سه مفهوم، انتزاع تابع[۱۹]، کاربرد[۲۰] و جانشینی[۲۱] بود. در واقع در منطق ترکیبی، انتزاع تابع به کمک ترکیب‌گرهای پایه (به عنوان عملگرهای ابتدایی) تعریف می‌شود. چند سال پس از کار شوون فینکل، کوری، به طور مستقل، منطق ترکیبی خودش و تقریباً همزمان، چرچ[۲۲] حساب لامبدای خودش را معرفی کرد. چرچ حدس زد که خانواده همه توابع محاسبه‌پذیر کارا همان خانواده همه توابع لامبدا-تعریف‌پذیر هستند. به عبارت دیگر، تز چرچ می‌گوید که هر تابع محاسبه‌پذیر را می‌توان با ترم‌هایی در حساب لامبدا نمایش داد. پس از معرفی ماشین‌های تورینگ این تز به تز چرچ- تورینگ معروف شد. نسخۀ اولیه حساب لامبدای چرچ و منطق کوری ناسازگار بودند. برای رفع این مشکل این دستگاه‌ها به نوع مجهز شدند. در میان همه این تحولات هیجان‌انگیز، پدیده‌ای محوری در این پژوهش‌ها نمود یافت: کوری متوجه شد که بین منطق ترکیبی و منطق گزاره‌ای شهودی تناظری وجود دارد. هر چند این کشف کوری تنها بخشی از تناظر اصلی را نشان می‌داد؛ با این حال بلافاصله موجب برانگیختن علایق تحقیقاتی در راستای کشف ارتباط عمیق‌تر میان این دو مقوله شد. این نکته مهم است که مشاهده اولیه کوری تمام آن چیزی نیست که امروزه با نام «تناظر کوری- هاوارد » می‌شناسیم. در واقع تناظر کامل را سه دهه بعد، هاوارد با ترکیب مشاهدات کوری و مشاهدات تیت[۲۳]، که می‌گوید تناظری میان حذف برش و تبدیل در حساب لامبدا وجود دارد، به‌دست آورد.

کار هاوارد موجب تکامل بعدی این تناظر شد. به ویژه، مارتین لاف[۲۴] که این تناظر را مستقیماً از هاوارد یاد گرفته بود، برای معرفی نظریه خویش عمیقاً از آن تأثیر گرفت. مارتین لاف نظریه نوع شهودی را معرفی کرد که در آن تناظری بین گزاره‌ها و انواع وجود داشت. نظریه انواع مارتین لاف را می‌توان به عنوان صوری‌سازی ریاضیات شهودی یا ریاضیات ساختی به‌شمار آورد. این نظریه مبنایی برای زبان برنامه‌نویسی تابعی Nuprl شد که کانستبل[۲۵] و همکارانش به عنوان یک اثبات‌یار معرفی کردند. کارهای هاوارد، مارتین لاف و ژیرارد[۲۶] پایه‌ای برای به وجود آمدن اثبات‌یارهای گوناگون، از جمله اثبات‌یار Coq، شد. این اثبات‌یار در کنار اثبات‌یار آتومت[۲۷] پیشرفت‌های چشمگیری در حوزه‌هایی چون منطق، ریاضیات و علوم کامپیوتر رقم زد. رایانه‌ها برای نخستین بار می‌توانستند برهان‌های صوری را تأیید کنند یا به جستجوی اثبات کمک کنند. هنوز هم اثبات‌یارها از موضوعات بسیار داغ پژوهشی هستند. در حال حاضر یک باور نانوشته وجود دارد که اثبات‌یارها ممکن است آینده ریاضیات را رقم بزنند.

این بخش را با اشاره‌ای به اهمیت تناظر کوری- هاوارد به پایان می‌بریم. در حالت کلی، یک نتیجه ریاضی ممکن است به دلایل گوناگونی مهم باشد؛ مثلاً ممکن است ارتباط یک مفهوم در یک حوزه را به مفاهیمی در حوزه‌های دیگر مشخص کند. قضیه تمامیت گودل از همین رو مهم است که میان تعریف معنایی رابطۀ استنتاج و تعریف نظریه برهانی آن ارتباط برقرار می‌سازد و در واقع نشان می‌دهد این دو مفهوم هم‌ارزند. نیز ممکن است یک نتیجه ریاضی به این سبب که موجب اثبات دیگر نتایج می‌شود اهمیت پیدا کند.

15- Jaskowski
16- Gentzen
17- Schönfinkel
18- Combinatory logic
19- Function abstraction
20- Application
21- Substitution
22- Church
23- Tait
24- Martin Lof
25- Constable
26- Girard
27- Automath

قضیه فشردگی در منطق این‌گونه است. این قضیه می‌گوید که برای نشـــان دادن ســـازگاری یک مجموعه از گزاره‌ها کافیســـت ســـازگاری هر بخش متناهی از آن را ثابت کنیم. یک معیار دیگر برای برآورد اهمیت یک نتیجهٔ علمی گستره حوزه‌هایی است که آن نتیجه می‌تواند با توضیح مفاهیم یا با ارایهٔ روش‌های جدید اثبات برای حل مسائل آن حوزه، بر آنها تأثیرگذار باشد. با این توصیفات، ارزیابی اهمیت تناظر کوری- هاوارد چندان آسان نیست. باید بگوییم کوری تنها با یک کنجکاوی توانست در سال ۱۹۳٤ م. شباهتی احتمالاً تصادفـــی میان منطق شـــهودی و منطق ترکیبی مشـــاهده کند. پس از آن، با کارهای هاوارد و رینولد[۲۸] روشـــن شـــد کـــه این تناظر در موارد گوناگون در حوزهٔ برنامه‌نویســـی کاربـــرد دارد و در پـــی آن، از دیدگاه نظری نیز اهمیت پیدا کرد. این اهمیت از آنجاســـت که می‌توان این تناظر را برای اثبات نرمال‌ســـازی قوی دســـتگاه اســـتنتاج طبیعی به کار بســـت؛ یعنی نه تنها برهان نرمال وجود دارد، بلکه می‌توان ثابت کردکه این‌گونه برهان‌ها منحصربه فرد هستند. اهمیت دیگر آن این است که ترم‌های نسبت داده شده به یک برهان دارای محتوای محاسباتی‌اند. برای درک بهتر این‌که منظور از محتوای محاســـباتی چیست، زمان بیشتری صرف شد، اما سرانجام اهمیت علمی و کاربردی این تناظر روشن شد. هم‌اکنون این تناظر پایهٔ طراحی سیستم‌های نوع در زبان‌های برنامه‌نویســـی است. از سوی دیگر، این تناظر، پایه‌ای برای طراحی وارسی‌کنندهٔ نوع خودکار و دستگاه‌های استنتاجی نوع برای زبان‌های برنامه‌نویسی واقعیست.

ســـاختار این نوشـــتار به شرح زیر اســـت: در بخش تاریخچـــه (بخش جاری) پیدایش این تناظر و اهمیت آن به صورت تاریخی بیان شـــد. در بخش دوم منطق شـــهودی و اســـتنتاج طبیعی از یک ســـو و حســـاب لامبدای نوع‌دار (به عنوان ســـاده‌ترین زبان برنامه‌نویسی) از سوی دیگر معرفی می‌شـــوند. در بخش ســـوم تناظر کوری- هاوارد صورت‌بنـــدی و اثبات می‌شـــود و در بخش پایانی درباره اهمیت این تناظر بحث خواهیم کرد.

28- Raynold

بخش دوم: استنتاج طبیعی[۲۹]، منطق شهودی[۳۰] و حساب لامبدای نوع‌دار[۳۱]

یکریختی کـــوری- هاوارد یک تناظر میـــان برهان در دستگاه‌های استنتاج طبیعی و ترم‌ها در حساب لامبداست. برای معرفی این تناظر باید دستگاه استنتاج طبیعی گنتزن را برای منطق شهودی معرفی کنیم. برای بیان قواعد دستگاه استنتاج طبیعی، رشته‌هایی به فرم $\Gamma \vdash A$ را به‌کار می‌بریم کـــه در آن Γ یک مجموعـــه از فرمول‌ها و A یک فرمول در زبان گزاره‌ای شامل رابطهای منطقی عطف و شرط است. این دســـتگاه برای هر رابط منطقی دو دسته قاعده تعریف می‌کنـــد: قاعده‌های معرفی و قاعده‌هـــای حذف. قاعده‌های معرفی نقش تعریف رابطهای منطقی را دارند و قاعده‌های حذف آن تعریف‌ها را توجیه می‌کنند. همان‌طور که پیشتر گفته شد، زبان گزاره‌ای تنها شامل دو رابط منطقی شرطی $\rightarrow$ و عطف $\wedge$ است. قواعد حذف و معرفی این دو رابط در جدول زیر آمده است.

قواعد معرفی	قواعد حذف
$\dfrac{A.\Gamma \vdash B}{\Gamma \vdash A \rightarrow B}(\rightarrow I)$	$\dfrac{\Gamma \vdash A \rightarrow B \quad \Gamma \vdash A}{\Gamma \vdash B}(\rightarrow E)$
$\dfrac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \wedge B}(\wedge I)$	$\dfrac{\Gamma \vdash A \wedge B}{\Gamma \vdash A}(\wedge E)$
	$\dfrac{\Gamma \vdash A \wedge B}{\Gamma \vdash B}(\wedge E)$

یک استنتاج در این دستگاه یک درخت است که برگ‌های آن مفروضات‌انـــد و گره‌های میانی با اســـتفاده از قواعد جـــدول فوق از گره‌های قبلی به دســـت می‌آیند. اگر Γ یک مجموعه از گزاره‌ها (مفروضات) و A یک گزاره باشـــد که با این مفروضات در این دستگاه استنتاج شود (یک درخت برهان داشته باشد)، آنگاه می‌نویسیم $\Gamma \vdash A$. اگر مجموعه

29- Natural deduction
30- Intuitionistic Logic
31- Typed Lambda Calculus

مفروضات Γ تهی باشد، می‌نویسیم $\vdash A$.

یکی از توانایی‌های دســـتگاه استنتاج طبیعی این است که هر برهان یا استنتاج در این دستگاه نرمال می‌شود؛ به این معنی که همواره یـــک دنباله از مراحل کاهش یا تبدیل وجـــود دارد که یک برهان را به یـــک برهان به فرم نرمال تبدیل کند. یک برهان به فرم نرمال است، هرگاه هیچ قاعدهٔ معرفیِ یک رابط بلافاصله پس از قاعدهٔ حذف آن رابط در درخت برهان ظاهر نشود. نرمال‌سازی برهان همان نقش را در دستگاه استنتاج طبیعی دارد که قاعدهٔ حذف- برش[32] در دســـتگاه حساب رشـــته‌ای گنتزن دارد. به طور خاص، یک نتیجه مستقیم نرمال‌سازی، اثبات ویژگی زیرفرمولی[33] دســـتگاه اســـتنتاج طبیعی اســـت؛ یعنی مقدمات یک قاعده (مفروضـــات)، زیرفرمول نتیجهٔ آن قاعده اســـت. ویژگی زیرفرمولی برای پیاده‌سازی عملی الگوریتم‌های جستجوی برهان ضروریست؛ زیرا فضای جستجو را محدود می‌کند و در نتیجه یکی از عامل‌های ناممکن شـــدن یک جستجوی حقیقی را برای پیدا کردن برهان حذف می‌کند. اکنون ببینیم فرآیند نرمال‌ســـازی یک برهان چگونـــه صورت می‌گیرد. نرمال‌ســـازی با حذف گام به گام مســـیرهای انحرافی در درخت برهان صورت می‌گیرد (اســـتفاده از قاعدهٔ معرفی بلافاصلـــه پس از یک قاعدهٔ حذف را یک مســـیر انحرافی، یـــا به اختصار یک انحراف در درخت برهان، می‌نامیم)؛ به عنوان مثال:

$$\dfrac{\dfrac{\overset{\pi}{A.\Gamma \vdash B}}{\Gamma \vdash A \to B}\to I \qquad \overset{\pi'}{\Gamma \vdash A}}{\Gamma \vdash B}\to E \quad \mapsto \quad \begin{matrix}\pi[\pi'/A] \\ \Gamma \vdash B\end{matrix}$$

کـــه در آن $\pi\ [\pi'/A]$ نتیجه جایگزینـــی در زیردرخت برهـــان π اســـت که به جای هر رشـــته بـــه فرم $A \vdash A$، زیردرخت π' که به $\Gamma \vdash A$ ختم می‌شود، جایگزین می‌شود. توجه شود که قواعد دســـتگاه هیچ تغییری در سمت چپ رشـــته ایجاد نمی‌کنند. بنابراین، هر وقوع A در سمت چپ یک رشته با Γ جایگزین می‌شود و به این ترتیب A ازسمت چپ رشـــته‌های موجود در درخت π حذف می‌شـــود. لذا اگر درخت π دارای رشـــته انتهایی $A, \Gamma \vdash B$ باشد، آنگاه درخت $\pi\ [\pi'/A]$ دارای رشته انتهایی $\Gamma \vdash B$ است. برای قواعد عطف نرمال‌سازی راحت‌تر است اگر $\wedge E$ بلافاصله پس از $\wedge I$ به‌کار رود. در این صورت نیازی نیست کاری بر روی درخت برهان انجام شود. مثال زیر را ببینید.

$$\dfrac{\dfrac{\overset{\pi}{\Gamma \vdash A} \qquad \overset{\pi'}{\Gamma \vdash B}}{\Gamma \vdash A \wedge B}\wedge I}{\Gamma \vdash A}\wedge E \quad \mapsto \quad \overset{\pi}{\Gamma \vdash A}$$

حساب لامبدای نوع‌دار

یادآوری می‌کنیم که یکریختی کوری- هاوارد تناظری میان برهان در دستگاه استنتاج طبیعی و ترم‌ها در حساب لامبدای نوع‌دار برقرار می‌ســـازد. در بخش پیش دستگاه اســـتنتاج طبیعـــی را معرفی کردیم. در این بخش حســـاب لامبدای نوع‌دار را معرفی می‌کنیم. با معرفی حساب لامبدای بدون نوع آغاز می‌کنیم. چرچ نخست همین حساب را برای دادن پاسخ منفی به مسئلهٔ تصمیم هیلبرت معرفی کرد، ولی به دلیل ناسازگاری بعدها این حساب را به حساب لامبدای نوع‌دار گسترش داد.

حســـاب لامبدا یک حساب روی ترم‌هاســـت. ترم‌ها از روی متغیرها ساخته می‌شوند و اگر t یک ترم باشد، آنگاه $\lambda x.t$ نیز یک ترم اســـت که آن را انتـــزاع لامبدا می‌نامیم. انتزاع، به طور شهودی، نمایش‌دهنده یک تابع (برنامه) است کـــه x را به عنوان یک ورودی می‌گیرد و مقدار مشـــخص شـــده توسط ترم t را برمی‌گرداند (توجه شود که x دارای وقوع آزاد اســـت). یادآور می‌شویم متغیر x در ترم $\lambda x.t$ مقیّد است. با این حســـاب می‌توانیم از ترم‌ها برای نمایش توابع استفاده کنیم. بنابراین می‌توان رفتار یا عمل یک تابع را روی یک ورودی (ترم) به سادگی با یک ترم به فرم (ts) نشـــان داد که در آن ترم s ورودی تابعیســـت که با ترم t نشان داده می‌شود. اضافه می‌کنیم که اگر t و s ترم باشند، آنگاه $< t, s >$ نیز یک ترم است. پس اگر t ترمی باشد که یک زوج را نشان دهد، آنگاه $\pi_1 t$ نشان‌دهندهٔ مولفه نخست

32- Cut elimination
33- Subformula property

t و $\pi_2 t$ نشـــان‌دهنده مولفهٔ دوم آن است. با این تعریف‌ها می‌توان به حساب لامبدا به عنوان یک زبان برنامه‌نویسی بســـیار ســـاده نگریســـت که در آن ترم‌ها برنامه هستند. همچنین اجرای یک برنامه برای یک ورودی چیزی نیست جز تبدیل ترم‌هایی که بیش از آن نتوان آنها را ســـاده کرد (مقداردهـــی کرد). به چنین ترم‌هایی ترم‌های به فرم نرمال گفته می‌شود که در واقع نتیجه محاسبه را نشان می‌دهند. تبدیل ترم‌ها از قوانین زیر تبعیت می‌کنند.

$(\lambda x.t)s \mapsto t[s/x]$

$\pi_1\langle t,s\rangle \mapsto t$

$\pi_1\langle t,s\rangle \mapsto t$

کـــه در آن $[{}^{s}/_{x}]$ یعنی هر وقـــوع آزاد متغیر x در ترم t با ترم s جایگزین می‌شـــود. برای نشان دادن این‌که این قواعد چگونه کار می‌کنند یک برنامه ساده‌ای را، که ترتیب مولفه‌هـــا را در یک زوج جابه‌جا می‌کنـــد، به عنوان نمونه پیاده‌سازی می‌کنیم.

$(\lambda x,\langle\pi_2 x,\pi_1 x\rangle)\langle u,v\rangle \mapsto <\pi_2\langle u,v\rangle,\pi_1\langle u,v\rangle$

$\mapsto <v,\pi_1 <u_1 v \gg$

$\mapsto \langle v,u\rangle$

در ایـــن مثال انتظار بر این اســـت کـــه ورودی برنامهٔ (ترم) $(\lambda x.\langle\pi_2 x,\pi_1 x\rangle)$ یک زوج باشد؛ ولی قواعد نحوی معرفی شـــده چنین انتظاری را برآورده نمی‌سازند؛ یعنی در حســـاب لامبدای بدون نوع مجازیم ورودی یک ترم را آزادانـــه انتخاب کنیم. برای جلوگیـــری از این آزادی عمل، حســـاب لامبدا را به نوع مجهز می‌کنیم و نحو آن را چنان معرفـــی می‌کنیم که محدودیت‌هایی برای ورودی برنامه‌ها (ترم‌ها) ایجاد کند. در این حســـاب جدیـــد، که معروف به حساب لامبدای نوع‌دار است، هر متغیر با یک نوع مشخص می شود. معمولاً نوع‌ها را با حروف بزرگ نشان می‌دهیم. پس وقتی می نویسیم x:A یعنی متغیر x فقط مقادیر نوع A را می‌گیرد. لازم به ذکر اســـت که انواع را به عنوان مفاهیم اولیهٔ نحـــوی در نظر می‌گیریم؛ مثلاً مقادیر ارزش یا اعداد طبیعی نوع هستند. ممکن اســـت نوع‌های تابعی هم داشته باشیم؛ برای نمونه توابعی از اعداد طبیعی به مقادیر ارزش. نـــوع $A \to B$ توابعی با ورودی از نوع A و خروجی از نوع B هســـتند. همچنین اگر A و B دو نوع باشند، آنگاه $A \wedge B$ یک نوع ضربی است شامل یک زوج با مولفهٔ اول از نوع A و مولفهٔ دوم از نوع B. با این توضیح برنامه مثال قبل تبدیل به فرم $\lambda x^{A\wedge B}.\langle\pi_2 x,\pi_1 x\rangle$ می‌شـــود که در آن متغیر x به اشـــیائی از نوع $A \wedge B$ محدود می‌شـــود (نوع ضرب را به صورت A×B نیز نمایش می‌دهند). لذا به راحتی می‌توان دیـــد که این ترم تابعـــی از یک زوج به یک زوج اســـت که ورودی‌اش از نـــوع $A \wedge B$ و خروجـــی‌اش از نوع $B \wedge A$ اســـت و در نتیجه ترم مذکور از نوع $A \wedge B \longrightarrow B \wedge A$ است که معمولاً آن را به صورت زیر نمایش می‌دهیم.

$$\lambda x^{A\wedge B}.\langle\pi_2 x,\pi_1 x\rangle : A \wedge B \longrightarrow B \wedge A$$

به ایـــن ترتیب، حســـاب لامبدای نوع‌دار بـــه هدفی که می‌خواســـت می‌رسد؛ یعنی ترم‌هایی را که در آنها انواع با هم مطابقت نداشـــته باشند، مجاز نمی‌شمارد. برای نمونه، ترم زیر مجاز نیست.

$$(\lambda x^{A\wedge B}.\langle\pi_2 x,\pi_1 x\rangle)(\lambda y^{A}.y)$$

زیرا ترم ســـمت چپ از نوع $A \wedge B \longrightarrow B \wedge A$ است؛ در حالی‌که $\lambda y^{A}.y$ از نوع $A \longrightarrow A$ است که به وضوح یک زوج نیســـت. برای بیان نظامند این محدودیت‌ها در حساب لامبدای نوع‌دار، باید قواعد ســـاختن (شکل دهی) ترم‌ها را ارائه دهیم.

پیش از معرفی قواعد، معرفی یک نماد ضروری است. فرض کنید t یک ترم، A یک نوع و Γ یک مجموعه از احکام نوع[۳۴] برای متغیرهای آزاد ظاهر شـــده در ترم t باشد. در این صورت می‌نویسیم: $\Gamma \vdash t:A$؛ برای مثال حکم:

$$x:A, y:B \vdash <x,y>: A \wedge B$$

به این معنی است که اگر x از نوع A و y از نوع B باشد، آنگاه ترم $\langle x,y\rangle$ از نوع $A \wedge B$ اســـت. با این مقدمه قواعد شکل‌دهی ترم‌ها به صورت زیر است.

34- Typed judgment

قواعد حذف	قواعد معرفی
$\dfrac{\Gamma \vdash t{:}A \to B \quad \Gamma' \vdash s{:}A}{\Gamma, \Gamma' \vdash (ts){:}B}(\to E)$	$\dfrac{x{:}A, \Gamma \vdash t{:}B}{\Gamma \vdash \lambda x^{A}.t{:}A \to B}(\to I)$
$\dfrac{\Gamma \vdash t{:}A \wedge B}{\Gamma \vdash \pi_1 t{:}A}(\wedge E)$	$\dfrac{\Gamma \vdash t{:}A \quad \Gamma \vdash s{:}B}{\Gamma \vdash < t,s >{:}A \wedge B}(\wedge I)$
$\dfrac{\Gamma \vdash t{:}A \wedge B}{\Gamma \vdash \pi_2 t{:}B}(\wedge E)$	

البته کنار این قواعد حکم $x{:}A \vdash x{:}A$ همواره درست است. به این گونه احکام اصل می‌گوییم. توجه شود که قواعد را تنها برای $\to$ و $\wedge$ بیان کردیم. بدیهی است اگر زبان شامل عملگرهای دیگر باشد، باید قواعد متناظر آنها نیز بیان شود. هم اکنون مفاهیم مورد نیاز برای بیان تناظر کوری- هاوارد فراهم شده است. در ادامه این تناظر با جزئیات بیشتری می‌آید.

بخش سوم: تناظر کوری- هاوارد

در بخش‌های قبل پیش‌نیازها را فراهم کردیم تا اکنون بتوانیم تناظری میان برهان‌ها و برنامه‌ها برقرار سازیم. تناظر مورد نظر دو بخش دارد. بخش نخست تناظر میان احکام و قواعد نوع در حساب لامبدای نوع‌دار از یک سو و قواعد استنتاج منطق شهودی از سوی دیگر را بیان می کند. در بخش دوم تناظر میان فرایند کاهش در حساب لامبدا و نرمال‌سازی برهان در منطق نشان داده می‌شود. بنابراین، تناظر کوری- هاوارد شامل مشاهدهٔ این مطلب است که سمت راست احکام نوع در قواعد تشکیل ترم در حساب نوع‌دار لامبدا دقیقاً همان قواعد معرفی و حذف در دستگاه استنتاج طبیعی است. برای مثال قاعده‌ی $(\longrightarrow E)$ به عنوان قاعدهٔساخت ترم و استنتاج نوع می‌گوید که اگر s یک ترم از نوع $A \to B$ و t یک ترم از نوع A باشد، آنگاه (st) یک ترم از نوع B است. به علاوه، این قواعد این امکان را فراهم می‌سازند که بتوانیم به طور نظام‌مندی ترم‌هایی را به یک استنتاج طبیعی اختصاص دهیم؛ تنها با این شرط که متغیرها به مفروضات اختصاص داده شوند.

حال اگر به جای توجه به ترم‌ها در یک حکم نوع، بر خود نوع در حکم تمرکز کنیم، آنگاه می‌توانیم قواعد استنتاج را به شیوهٔ دیگری تفسیر کنیم. به عنوان مثال قاعدهٔ $(\longrightarrow E)$ می‌گوید که اگر s به نوع $A \to B$ و t به نوع A اختصاص داده شوند، آنگاه ترم (st) باید به نتیجهٔ B اختصاص داده شود. به این ترتیب می‌توان به هر برهان در دستگاه استنتاج طبیعی یک ترم به رشته‌ها در حساب لامبدا نسبت داد؛ برای نمونه درخت برهان $A \wedge B \longrightarrow B \wedge A$ را با ترم‌های متناظر هر فرمول بازنویسی می‌کنیم. اما این کار را زمانی می‌توان انجام داد که از پیش به مفروضات، متغیرهایی اختصاص داده شده باشد. در این مثال متغیر x به $A \wedge B$ اختصاص داده شده است.

$$\dfrac{\dfrac{\dfrac{x{:}A \wedge B \vdash x{:}A \wedge B}{x{:}A \wedge B \vdash \pi_2 x{:}B}\wedge E \quad \dfrac{x{:}A \wedge B \vdash x{:}A \wedge B}{x{:}A \wedge B \vdash \pi_1 x{:}A}\wedge E}{x{:}A \wedge B \vdash < \pi_2 x, \pi_1 x >{:}A \wedge B}\wedge I}{\vdash \lambda x^{A \wedge B}. < \pi_2 x, \pi_1 x >{:}(A \wedge B) \to (B \wedge A)}\to I$$

بخش دوم تناظر کوری- هاوارد می‌گوید که روند نرمال‌سازی یک برهان در دستگاه استنتاج طبیعی متناظر با روند کاهش (ارزیابی-محاسبه) یک ترم در حساب لامبداست. هنگامی که یک مسیر انحرافی مانند $\to I/\to E$ را از درخت برهان حذف می‌کنیم، آنگاه ترمی که به نتیجهٔ برهان پیش از حذف اختصاص می‌دهیم به فرم $(\lambda x.t)s$ و پس از حذف ترم اختصاص داده شده به نتیجه، به فرم $t[{}^{s}/{}_{x}]$ است؛ یعنی:

$$\dfrac{\dfrac{\overset{\pi}{x{:}A, \Gamma \vdash t{:}B}}{\Gamma \vdash \lambda x^{A}.t{:}A \to B}\to I \quad \overset{\pi'}{\Gamma \vdash s{:}A}}{\Gamma \vdash (\lambda x^{A}.t)s{:}B}\to E \quad \mapsto \quad \begin{array}{c}\pi[\pi'/A]\\ \Gamma \vdash t[s/x]{:}B\end{array}$$

مشابهاً برای حذف انحراف ${}^{\wedge I}/{}_{\wedge E}$ داریم:

$$\dfrac{\dfrac{\overset{\pi}{\Gamma \vdash t{:}A} \quad \overset{\pi'}{\Gamma \vdash s{:}B}}{\Gamma \vdash < t,s >{:}A \wedge B}\wedge I}{\Gamma \vdash \pi_1 < t,s >{:}A}\wedge E \quad \mapsto \quad \begin{array}{c}\pi\\ \Gamma \vdash t{:}A\end{array}$$

توجه شود که یک برهان در دستگاه استنتاج طبیعی نرمال است (شامل هیچ انحرافی نیست) اگر و تنها اگر ترم اختصاص داده شده به آن نرمال باشد. هر انحراف $\to I/\to E$ متناظر با یک زیرترم به فرم $(\lambda x^{A}.t)s$ و هر

انحراف $\wedge I / \wedge E$ متناظر با یـک زیرترم به فرم. $\pi_i(s,t)$ است. ترمی را که کاهش‌پذیر به ترم دیگر نباشد ترم نرمال می‌گوییم. (برای مطالعهٔ بیشتر به [۳] مراجعه کنید.)

بخش چهارم: اهمیت تناظر کوری- هاوارد

تناظر کوری- هاوارد در شکل منطقی خود بیان می‌کند که:

۱. می‌تـوان به برهان‌های دســتگاه اســتنتاج طبیعی ترم‌های برهان اختصاص داد.

۲. نرمال‌ســازی برهان‌هــا بــا کاهــش در ترم‌هــای متناظرشان معادل است.

تناظــر کوری- هاوارد یک نتیجهٔ نظری مهم به‌شــمار می‌آید؛ زیرا این تناظر امکان اثبات نرمال‌سازی قوی را در دستگاه استنتاج طبیعی فراهم می‌سازد. هرچند نرمال‌سازی برهان فرآیندی نسبتاً ساده است؛ ولی نرمال‌سازی قوی، که می‌گوید هر فرآیند کاهش حتماً پایان می‌یابد و مهمتر از آن همهٔ آنها به یک فرم نرمال واحد ختم می‌شــوند، چندان ساده نیســت. با تناظر کوری- هاوارد این فرآیند، متناظر اســت با فرآیند مشــابه در حســاب لامبدا که می‌گوید هر دنباله از محاســبات (کاهش) به یک ترم نرمال (یک مقدار) منتهی می‌شود و همه مسیرهای گوناگون محاسبه به یک نتیجه ختم می‌شــوند. در این نوشــتار برای سادگی، این تناظر تنها برای بخشی از زبان منطق شهودی آمد. به طور مشــابه می‌توان و با تلاش بیشــتر این کار را برای طیف گســتردەتری از دســتگاه‌ها و منطق‌ها در حساب استنتاج طبیعی ارائه داد.

از ســوی دیگر، نســخه محاســباتی این تناظر مبنای بسیاری از کارهای مهم دهه گذشته در حوزهٔ اثبات خودکار و نظریهٔ زبان‌های برنامه‌نویسی بوده است. لامبداترم‌ها به اســتنتاج های نوع و مقداردهی یا ارزیابی ترم‌های شامل نرمال‌سازی استنتاج‌های نوع مرتبط می‌شوند. اهمیت این تناظر برای نظریهٔ زبان‌های برنامه‌نویسی در این امر نهفته اســت که مدلی برای وارسی نوع در بسیاری از زبان‌های برنامه‌نویســی فراهم می‌کند. لازم به ذکر است که حساب لامبدای نوع‌دار حکم زبان برنامه‌نویســی را دارد. یک ترم بــه فرم $\lambda x^A.t$ در واقــع یک برنامه بــا ورودی از نوع A و خروجی از نوع B اســت؛ در صورتی کــه ترم t از نوع $A \longrightarrow B$ بوده باشــد. لذا منظور از وارسی نوع، حصول اطمینان از این اســت که ترم از نوع مورد نظر باشــد. این وارســی دقیقاً متناظر با این پرسش نظریهٔ برهانیست که: آیا اســتنتاجی برای $\vdash \lambda x^A.t{:}A \to B$ وجود دارد؟ یعنی آیا اســتنتاجی برای $A \longrightarrow B$ وجــود دارد که نوع $\lambda x^A.t$، بــه عنوان یک تــرم اثبات، به آن اختصاص داده شــود؟ تناظر کوری- هاوارد می‌گوید که همواره می‌توان به کمک یک اســتنتاج به فرم نرمال، نوع ترم‌ها را وارســی کرد و لذا از آنجا که فرم نرمال دارای ویژگی زیرفرمولی اســت، جستجوی استنتاج برای وارسی نوع در یک برنامه همواره کارآمد است. یک زبان برنامه‌نویسی «ایمن از نوع»[۳۵] نامیده می‌شــود؛ هرگاه از برنامه‌هایی که منجر به خطاهای نوع شوند جلوگیری کند. یک خطای نوع هنگامی رخ می‌دهد که مثلاً یک برنامه از نوع $A \longrightarrow B$ روی ورودی‌ای که از نوع A نباشــد پیاده‌سازی شــود یا ممکن است ورودی از نوع A باشــد؛ ولی نتیجه (خروجی) چیزی غیر از نوع B باشد. تناظر کوری- هاوارد و ویژگی‌های مشــابه، امنیت نوع را برای زبان‌های برنامه‌نویسی تضمین می‌کنند. این موضوع اهمیت نظری و کاربــردی فوق‌العاده‌ای دارد. برنامه‌ها در زبان‌های ایمن از نوع هرگز ناخواســته متوقف نمی‌شوند؛ زیــرا همواره راهی برای اجرای برنامه تا رســیدن به یک مقدار وجود دارد. اینها همگی از نتایج تناظر کوری- هاوارد است. این تناظر دو نتیجه سودمند دارد: یک این‌که اگر یک ترم به یک مقدار منتهی نشود (یعنی به فرم نرمال در نیامده باشــد)، آنگاه مقداردهی می‌تواند ادامه یابــد؛ زیرا در این صورت هنوز زیرترمی وجود دارد که قابل جانشینی باشد. این ویژگی به پیشرفت[۳۶] معروف است. مهمتر این که، تناظر کوری- هاوارد نقشی اساسی در بیان این حقیقت دارد که

35- Type safe
36- Progress

در تبدیل یا کاهش یک ترم به ترم دیگر نوع ثابت می‌ماند. همچنین استنتاج طبیعی متناظری وجود دارد که این مطلب را تأیید می‌کند. ویژگی عدم تغییر نوع در فرایند کاهش به «حفظ نوع»[۳۷] معروف است. همراهی دو ویژگی ایمنی نوع و حفظ نوع این موضوع را تأیید می‌کند که اگر t یک ترم (برنامه) از نوع $B \longrightarrow A$ و s یک ترم (ورودی) از نوع A باشد، آنگاه ترم (ts) همواره یک ترم نرمال (مقدار) از نوع B را ارزیابی می‌کند. اکنون ویژگی نرمال‌سازی قوی برای حساب لامبدای نوع‌دار، تأیید می‌کند که این ارزیابی پایان می‌یابد و به علاوه، هر ترتیبی از مراحل ارزیابی به یک مقدار منجر می‌شود. سخن آخر این‌که توانایی وارسی نوع برنامه‌ها و ویژگی ایمنی نوع، گونه‌ای از قضیهٔ درستی و تمامیت برای برنامه‌هاست. همانند سازگاری در مبانی ریاضیات، که دست‌کم انتظار ما از یک نظریه ریاضی است، ایمنیِ نوع تضمین حداقل ایمنی برای برنامه‌ها در زبان‌های ایمن از نوع است. (برای اطلاعات بیشتر دربارهٔ اهمیت تناظر کوری- هاوارد به مراجع [۳] و [٤] مراجعه کنید.)

مراجع

1.Hindley, J. Roger. Lambda-Calculus and Combinators: An Introduction. In collab. With Jonathan P. Seldin. Cambridge: Cambridge University Press, 2008.
2. Stewart Shapiro and Peter King. "The History of Logic". In: The Oxford Companion to Philosophy. Ed. by Ted Honderich. Oxford University Press, 1995.
3. Sørensen, Morten Heine and Pawel Urzyczyn. Lectures on the Curry-Howard Isomorphism. 1st ed. Amsterdam: Elsevier, 2006.
4. Philip Wadler. "Propositions as Types". In: Communications of the ACM 58.12 (2015), pp. 75–84. DOI: 10.1145/2699407, p. 79.

37- Type preservation