Chapter 1

Classical Logic

Unlike constructive logic which is usually explained operationally by associating each proposition A with a proof proving or refuting its truth, classical logic is a logic that is usually explained *denotationally* by associating each proposition A with a truth value, either true T or false F. As there are only two truth values in classical logic, it is both sound and complete to check the truth value of a proposition A with a truth table in which all possible combinations of truth values of atomic propositions in A are considered in turn. For example, the connectives in classical propositional logic can be explained as follows:

A	B	$A \wedge B$	$A \vee B$	$A \supset B$
T	T	T	T	T
T	F	F	T	F
F	T	F	T	T
F	F	F	F	T

In this chapter, we investigate proof-theoretic formulations of classical logic based on inference rules. We also investigate its operational interpretation, or its computational contents, to obtain useful constructs for programming languages.

1.1 A judgmental formulation of classical logic

We have already seen in Chapter **??** that the addition of the following rule to constructive logic yields classical logic:

$$\frac{}{A \vee \neg A \ \mathit{true}}\ \mathsf{EM}$$

The rule EM, called *the law of excluded middle*, asserts that for any proposition A, either $A\ \mathit{true}$ or $\neg A\ \mathit{true}$ must hold regardless of the existence of an actual proof. Another way to obtain classical logic is to add one of the following rules:

$$\frac{}{\neg\neg A \supset A\ \mathit{true}}\ \mathsf{DNE} \qquad \frac{}{((A \supset B) \supset A) \supset A\ \mathit{true}}\ \mathsf{Pierce}$$

The rule DNE, called *the law of double-negation elimination*, asserts that if A cannot be false, it must be true. The rule Pierce, called *Pierce's law*, says that a proof of $A\ \mathit{true}$ may freely assume $A \supset B\ \mathit{true}$ for an arbitrary proposition B. The three rules above are all equivalent to each other in that the addition of any of these rules renders the other two rules derivable.

Note that these rules destroy the orthogonality of the system. For example, in the presence of the rule EM which uses two connectives $\supset$ and $\neg$ in the conclusion, the meaning of $\supset$ depends on the meaning of $\neg$ (or vice versa). The rule Pierce is also bad because it tries to explain the meaning of $\supset$ presupposing the notion of $\supset$. (As a rule of thumb, an inference rule using multiple connectives, whether same or different,

is always bad.) Instead of using these rules, therefore, we develop a system of classical logic based purely on judgmental notions.

We recall that at the heart of classical logic lies the principle of *proof by contradiction* (which is commonly employed in mathematical proofs). Using a truth judgment $A\ \mathit{true}$ and a falsehood judgment $A\ \mathit{false}$ (which denotes "*A is false*"), we can explain the idea as follows: in order to prove $A\ \mathit{true}$, we first assume $A\ \mathit{false}$ and then show a contraction, which is established by a proof of $B\ \mathit{true}$ when an assumption $B\ \mathit{false}$ is available for some proposition B. Observe that we may have to make an assumption of a falsehood judgment in the course of a proof, but we never directly conclude a falsehood judgment. Thus we use a new form of hypothetical judgment $\Gamma; \Delta \vdash_{\mathsf{K}} C\ \mathit{true}$ (where the subscript K stands for *K*lassical), in which antecedents may include falsehood judgments, but the succedent is always a truth judgment:

$$\Gamma; \Delta \vdash_{\mathsf{K}} C\ \mathit{true} \qquad \text{where} \qquad \begin{array}{rcl} \Gamma & ::= & \cdot \mid \Gamma, A\ \mathit{true} \\ \Delta & ::= & \cdot \mid \Delta, A\ \mathit{false} \end{array}$$

All previous rules from constructive propositional logic (which do not need falsehood judgments) are turned into corresponding rules for classical propositional logic by rewriting $\Gamma \vdash C\ \mathit{true}$ as $\Gamma; \Delta \vdash_{\mathsf{K}} C\ \mathit{true}$. The principle of proof by contradiction is implemented by the following two rules:

$$\frac{\Gamma; \Delta, A\ \mathit{false} \vdash_{\mathsf{K}} A\ \mathit{true}}{\Gamma; \Delta \vdash_{\mathsf{K}} A\ \mathit{true}}\ \mathsf{Contra}\uparrow \qquad \frac{\Gamma; \Delta, A\ \mathit{false} \vdash_{\mathsf{K}} A\ \mathit{true}}{\Gamma; \Delta, A\ \mathit{false} \vdash_{\mathsf{K}} C\ \mathit{true}}\ \mathsf{Contra}\downarrow$$

The rule Contra $\uparrow$ states that in order to prove $A\ \mathit{true}$, we may assume $A\ \mathit{false}$ and then show a contradiction by proving $A\ \mathit{true}$; hence it is best read in a bottom-up way (as indicated by the upward arrow in Contra $\uparrow$). The rule Contra $\downarrow$ states that if antecedents Γ and $\Delta, A\ \mathit{false}$ are inconsistent (because $A\ \mathit{true}$ is provable from Γ and $\Delta, A\ \mathit{false}$), we may conclude $C\ \mathit{true}$ for any proposition C as long as the same antecedents Γ and $\Delta, A\ \mathit{false}$ are available; hence it is best read in a top-down way (as indicated by the downward arrow in Contra $\downarrow$). Note that the premise is the same in both rules.

Here are a couple of examples showing that the rules EM and DNE are now derivable.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{A\ \mathit{true}; A \vee \neg A\ \mathit{false} \vdash_{\mathsf{K}} A\ \mathit{true}}\ \mathsf{Hyp}}{A\ \mathit{true}; A \vee \neg A\ \mathit{false} \vdash_{\mathsf{K}} A \vee \neg A\ \mathit{true}}\ \vee\mathsf{I}_{\mathsf{L}}}{A\ \mathit{true}; A \vee \neg A\ \mathit{false} \vdash_{\mathsf{K}} \bot\ \mathit{true}}\ \mathsf{Contra}\downarrow}{\cdot; A \vee \neg A\ \mathit{false} \vdash_{\mathsf{K}} \neg A\ \mathit{true}}\ \neg\mathsf{I}}{\cdot; A \vee \neg A\ \mathit{false} \vdash_{\mathsf{K}} A \vee \neg A\ \mathit{true}}\ \vee\mathsf{I}_{\mathsf{R}}}{\cdot; \cdot \vdash_{\mathsf{K}} A \vee \neg A\ \mathit{true}}\ \mathsf{Contra}\uparrow$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\neg\neg A\ \mathit{true}; A\ \mathit{false} \vdash_{\mathsf{K}} \neg\neg A\ \mathit{true}}\ \mathsf{Hyp} \quad \dfrac{\dfrac{\dfrac{\dfrac{}{\neg\neg A\ \mathit{true}, A\ \mathit{true}; A\ \mathit{false} \vdash_{\mathsf{K}} A\ \mathit{true}}\ \mathsf{Hyp}}{\neg\neg A\ \mathit{true}, A\ \mathit{true}; A\ \mathit{false} \vdash_{\mathsf{K}} \bot\ \mathit{true}}\ \mathsf{Contra}\downarrow}{\neg\neg A\ \mathit{true}; A\ \mathit{false} \vdash_{\mathsf{K}} \neg A\ \mathit{true}}\ \neg\mathsf{I}}{\neg\neg A\ \mathit{true}; A\ \mathit{false} \vdash_{\mathsf{K}} \bot\ \mathit{true}}\ \neg\mathsf{E}}{\neg\neg A\ \mathit{true}; A\ \mathit{false} \vdash_{\mathsf{K}} A\ \mathit{true}}\ \bot\mathsf{E}}{\neg\neg A\ \mathit{true}; \cdot \vdash_{\mathsf{K}} A\ \mathit{true}}\ \mathsf{Contra}\uparrow}{\cdot; \cdot \vdash_{\mathsf{K}} \neg\neg A \supset A\ \mathit{true}}\ \supset\mathsf{I}$$

1.2 Proof terms

We apply the Curry-Howard isomorphism to the new form of hypothetical judgment $\Gamma; \Delta \vdash_{\mathsf{K}} C\ \mathit{true}$ by first associating variables with judgments in Γ and Δ and then assigning a proof term of type C. For $A\ \mathit{true}$ in Γ, we bind a variable x to type A to obtain $x : A$. For $A\ \mathit{false}$ in Δ, we need a new notation to indicate that a variable x is associated with a falsehood judgment $A\ \mathit{false}$. We choose a new notation $x : A\ \mathit{false}$, in

which $A\ \mathit{false}$ can be thought of as a type instead of a judgment; in the context of type theory, $A\ \mathit{false}$ would stand for the type of *continuations* of type A.

$$\begin{array}{rcl} \Gamma & ::= & \cdot \mid \Gamma, x : A \\ \Delta & ::= & \cdot \mid \Delta, x : A\ \mathit{false} \end{array}$$

As proof terms corresponding to the rules Contra $\uparrow$ and Contra $\downarrow$, we use $\mathsf{callcc}\ x : A\ \mathit{false}.\ M$ and $\mathsf{throw}\ M\ \mathsf{to}\ x$ which are constructs for capturing and throwing continuations in programming languages:

$$\frac{\Gamma; \Delta, x : A\ \mathit{false} \vdash_{\mathsf{K}} M : A}{\Gamma; \Delta \vdash_{\mathsf{K}} \mathsf{callcc}\ x : A\ \mathit{false}.\ M : A}\ \textsf{Callcc} \qquad \frac{\Gamma; \Delta, x : A\ \mathit{false} \vdash_{\mathsf{K}} M : A}{\Gamma; \Delta, x : A\ \mathit{false} \vdash_{\mathsf{K}} \mathsf{throw}\ M\ \mathsf{to}\ x : C}\ \textsf{Throw}$$

In the rule Throw, proof term M is allowed to contain variable x.

A simple case of reducing a proof term $\mathsf{throw}\ M\ \mathsf{to}\ x$ occurs when M does not contain x:

$$\mathsf{callcc}\ x : A\ \mathit{false}.\ \sigma[\![\mathsf{throw}\ M\ \mathsf{to}\ x]\!] \quad \Longrightarrow_R \quad M$$

Here $\sigma[\![\mathsf{throw}\ M\ \mathsf{to}\ x]\!]$ denotes a certain proof term containing $\mathsf{throw}\ M\ \mathsf{to}\ x$ as a subterm. By the rule Callcc, it has type A so that the whole proof term $\mathsf{callcc}\ x : A\ \mathit{false}.\ \sigma[\![\mathsf{throw}\ M\ \mathsf{to}\ x]\!]$ is assigned type A. By the rule Throw, proof term M also has type A, and thus the type of the proof term being reduced is preserved. For a full account of reductions of $\mathsf{callcc}\ x : A\ \mathit{false}.\ M$ and $\mathsf{throw}\ M\ \mathsf{to}\ x$, we need to formalize the definition of σ, which we do not pursue here.

A proof term LEM of type $A \vee \neg A$ is given as follows:

$$\mathit{LEM} \quad = \quad \mathsf{callcc}\ x : A \vee \neg A\ \mathit{false}.\ \mathsf{inr}_A\ \lambda y{:}A.\ \mathsf{throw}\ \mathsf{inl}_{\neg A}\ y\ \mathsf{to}\ x$$

A proof term DNE of type $\neg\neg A \supset A$ is given as follows:

$$\mathit{DNE} \quad = \quad \lambda x{:}\neg\neg A.\ \mathsf{callcc}\ y : A\ \mathit{false}.\ \mathsf{abort}_A\ (x\ (\lambda z{:}A.\ \mathsf{throw}\ z\ \mathsf{to}\ y))$$

1.3 Sequent calculus for classical logic

The sequent calculus for classical logic uses a new form of sequent whose definition is motivated by the principle of proof by contradiction, rather than the notion of normal proof as in constructive logic. We write $A_1, \cdots, A_n \Longrightarrow B_1, \cdots, B_m$ to mean that assumptions of $A_1\ \mathit{true}$, $\cdots$, $A_n\ \mathit{true}$ and $B_1\ \mathit{false}$, $\cdots$, $B_m\ \mathit{false}$ lead to a contradiction. Note that unlike a sequent $\Gamma \longrightarrow C$ for constructive logic, the new sequent allows multiple propositions in the right side. We use Γ for a collection of propositions in the left side and Δ for a collection of propositions in the right side (*e.g.*, $\Gamma \Longrightarrow \Delta$). We assume that Γ and Δ are unordered.

The definition of the new form of sequent justifies the following rule, which is similar to the rule Init in constructive logic but actually expresses the principle of proof by contradiction:

$$\frac{}{\Gamma, A \Longrightarrow A, \Delta}\ \mathit{Contra}$$

The rule Contra says that since assumptions of $A\ \mathit{true}$ and $A\ \mathit{false}$ lead to a contradiction, the proof of $\Gamma, A \Longrightarrow A, \Delta$ is completed immediately.

As in the sequent calculus for constructive logic, we give left and right rules for each connective. Although these rules appear to be mechanically derived from their corresponding rules in the sequent calculus for constructive logic, their interpretation is different because the definition of $\Gamma \Longrightarrow \Delta$ is motivated differently from the definition of $\Gamma \longrightarrow C$. As each rule focuses on a proposition in the left or right side in a given sequent, we choose to reuse the rule names from the sequent calculus for constructive logic.

Figure 1.1 shows all the rules in the sequent calculus for classical propositional logic. As in the sequent calculus for constructive logic, the proof of a sequent always proceeds in a bottom-up way. There are two important observations to make. First, unlike in the sequent calculus for constructive logic, the right side

$$\frac{}{\Gamma, A \Longrightarrow A, \Delta}\ \mathit{Contra}$$

$$\frac{\Gamma, A \wedge B, A \Longrightarrow \Delta}{\Gamma, A \wedge B \Longrightarrow \Delta}\ \wedge L_L \qquad \frac{\Gamma, A \wedge B, B \Longrightarrow \Delta}{\Gamma, A \wedge B \Longrightarrow \Delta}\ \wedge L_R \qquad \frac{\Gamma \Longrightarrow A, A \wedge B, \Delta \quad \Gamma \Longrightarrow B, A \wedge B, \Delta}{\Gamma \Longrightarrow A \wedge B, \Delta}\ \wedge R$$

$$\frac{\Gamma, A \vee B, A \Longrightarrow \Delta \quad \Gamma, A \vee B, B \Longrightarrow \Delta}{\Gamma, A \vee B \Longrightarrow \Delta}\ \vee L \qquad \frac{\Gamma \Longrightarrow A, A \vee B, \Delta}{\Gamma \Longrightarrow A \vee B, \Delta}\ \vee R_L \qquad \frac{\Gamma \Longrightarrow B, A \vee B, \Delta}{\Gamma \Longrightarrow A \vee B, \Delta}\ \vee R_R$$

$$\frac{}{\Gamma \Longrightarrow \top, \Delta}\ \top R \qquad \frac{}{\Gamma, \bot \Longrightarrow \Delta}\ \bot L \qquad \frac{\Gamma, \neg A \Longrightarrow A, \Delta}{\Gamma, \neg A \Longrightarrow \Delta}\ \neg L \qquad \frac{\Gamma, A \Longrightarrow \neg A, \Delta}{\Gamma \Longrightarrow \neg A, \Delta}\ \neg R$$

$$\frac{\Gamma, A \supset B \Longrightarrow A, \Delta \quad \Gamma, A \supset B, B \Longrightarrow \Delta}{\Gamma, A \supset B \Longrightarrow \Delta}\ \supset L \qquad \frac{\Gamma, A \Longrightarrow B, A \supset B, \Delta}{\Gamma \Longrightarrow A \supset B, \Delta}\ \supset R$$

Figure 1.1: Sequent calculus for classical propositional logic

of a sequent should *not* be read as a collection of conclusions to be drawn; rather it should be read as a collection of assumptions (consisting of falsehood judgments). Second the premise in a rule (especially in a right rule) always contains more assumptions than the conclusion: we never cancel an existing assumption because the goal is to elicit a contradiction from a collection of assumptions.

The left rules $\wedge L_L$, $\wedge L_R$, $\vee L$ can be read in the same manner as their counterparts in the sequent calculus for constructive logic. The right rules, however, cannot be read in the same manner because the right side of a sequent should be read not as a collection of conclusions but as a collection of assumptions. In the rule $\wedge R$, for example, we use an assumption $A \wedge B$ *false* to yield a contradiction, rather than deduce a conclusion $A \wedge B$ *false*. Since $A \wedge B$ *false* holds when either A *false* or B *false* holds, we have to show a contradiction in each case. (Using the right side as a collection of conclusions would make it difficult, or even impossible, to make sense of the rule $\wedge R$.) The right rules $\vee R_L$ and $\vee R_R$ show that we never cancel an existing assumption.

The rule $\top R$ makes sense because an assumption $\top$ *false* expresses a contradiction: $\top$ cannot be false. Likewise the rule $\bot L$ makes sense because $\bot$ cannot be true. The rule $\neg L$ states that assuming $\neg A$ *true* is equivalent to assuming A *false*; similarly the rule $\neg R$ states that assuming $\neg A$ *false* is equivalent to assuming A *true*. (Here we do not use the notational definition of $\neg A$ as $A \supset \bot$.)

Although Figure 1.1 includes the rules $\supset L$ and $\supset R$, these rules are in fact derived rules because implication is a derived notion: classical logic has no notion of transforming a proof into another because every proposition denotes just a truth value, and thus $A \supset B$ is *defined* as $\neg A \vee B$. Lack of the notion of implication in classical logic is also the reason why $A \supset B$ is assigned a truth value T whenever A is assigned a truth value F.

The sequent calculus in Figure 1.1 satisfies the weakening and contraction properties which allow us to use a proposition A in Γ or Δ as many times as necessary in a proof of $\Gamma \Longrightarrow \Delta$. It also satisfies the subformula property in the same sense as in the sequent calculus for constructive logic, which in turn implies that the sequent calculus is decidable.

Proposition 1.1 (Structural properties).

(Weakening) *If* $\Gamma \Longrightarrow \Delta$, *then* $\Gamma, A \Longrightarrow \Delta$.
If $\Gamma \Longrightarrow \Delta$, *then* $\Gamma \Longrightarrow A, \Delta$.

(Contraction) *If* $\Gamma, A, A \Longrightarrow \Delta$, *then* $\Gamma, A \Longrightarrow \Delta$.
If $\Gamma \Longrightarrow A, A, \Delta$, *then* $\Gamma \Longrightarrow A, \Delta$.

Proof. By induction on the structure of the proof of $\Gamma \Longrightarrow \Delta$ and $\Gamma, A, A \Longrightarrow \Delta$ and $\Gamma \Longrightarrow A, A, \Delta$. □

As in the sequent calculus for constructive logic, there is a cut rule whose admissibility implies that the sequent calculus is sound (*i.e.*, $\cdot \Longrightarrow \bot$ is not provable). Interestingly it expresses precisely the law of excluded middle:

$$\frac{\Gamma \Longrightarrow A, \Delta \quad \Gamma, A \Longrightarrow \Delta}{\Gamma \Longrightarrow \Delta}\ \mathit{Cut}$$

The two premises cover all possibilities because for every proposition A, either $A\ true$ or $A\ false$ must hold by the law of excluded middle. Hence the provability of the premises implies that a contradiction can always be reached whenever assumptions Γ and Δ are available. It turns out that the rule Cut is indeed admissible, and we can conclude that the law of excluded middle is built into the sequent calculus.

There is another way of interpreting a sequent $\Gamma \Longrightarrow \Delta$, which is called the *multi-conclusion view*. Under the multi-conclusion view, $A_1, \cdots, A_n \Longrightarrow B_1, \cdots, B_m$ means that if assumptions $A_1\ true$, $\cdots$, "and" $A_n\ true$ are available, a conclusion $B_1\ true$, $\cdots$, "or" $B_m\ true$ is provable. While all the rules in Figure 1.1 continue to make sense, they fail to express the essence of classical logic, namely the principle of proof by contradiction (or the law of excluded middle).

1.4 Double-negation translation and CPS transformation

We have seen that classical logic is obtained by augmenting constructive logic with two new rules $\mathsf{Contra}\uparrow$ and $\mathsf{Contra}\downarrow$. As it comes with more inference rules, classical logic allows us to prove more truth judgments than constructive logic. That is, a proof of $A\ true$ in constructive logic is a valid proof in classical logic as well, and therefore, if $A\ true$ is provable in constructive logic, it is also provable in classical logic. The converse is certainly untrue, as evidenced by such judgments as $A \vee \neg A\ true$, $\neg\neg A \supset A\ true$, and $((A \supset B) \supset A) \supset A\ true$.

It is important that more judgments being provable does not necessarily mean more expressive power. For example, a system in which $\bot\ true$ is provable is totally useless (and is said to be inconsistent), even though every truth judgment is provable. In the case of classical logic, it allows more judgments to be provable, but is *less* expressive than constructive logic, which is capable of simulating classical logic via the *double-negation translation* to be explained below. In essence, constructive logic can make a finer distinction between truth and falsehood than classical logic (as it allows not only truth and falsehood but also "excluded middle").

We write A° for the proposition in constructive logic corresponding to proposition A in classical logic under the double-negation translation. The translation is structural except for $A \supset B$, which is translated to $A^\circ \supset \neg\neg B^\circ$:

$$\begin{array}{rcl} (A \wedge B)^\circ & = & A^\circ \wedge B^\circ \\ (A \vee B)^\circ & = & A^\circ \vee B^\circ \\ \top^\circ & = & \top \\ \bot^\circ & = & \bot \\ (A \supset B)^\circ & = & A^\circ \supset \neg\neg B^\circ \\ P^\circ & = & P \end{array}$$

An analogy in programming language theory is that an invocation of a "classical" function of type $A \supset B$ is in effect an invocation of a "constructive" function of type $A \supset \neg\neg B = A \supset ((B \supset \bot) \supset \bot)$, which returns an answer (of type $\bot$) when given an argument of type A and a return address (of type $B \supset \bot$) expecting an argument of type B.

Theorem 1.2 shows that classical logic is embedded in constructive logic via the double-negation translation, where we use the subscript I in a hypothetical judgment to indicate that it is valid only in *I*ntuitionistic logic, which is another name for constructive logic. Antecedents in hypothetical judgments are translated as follows:

$$\Gamma^\circ = \{A^\circ\ true \mid A\ true \in \Gamma\}$$
$$\neg\Delta^\circ = \{\neg A^\circ\ true \mid A\ false \in \Delta\}$$

Theorem 1.2 (Embedding of classical logic in constructive logic). *If* $\Gamma; \Delta \vdash_{\mathsf{K}} C\ true$, *then* $\Gamma^\circ, \neg\Delta^\circ \vdash_{\mathsf{I}} \neg\neg C^\circ\ true$.

An immediate corollary of Theorem 1.2 is that classical logic is relatively consistent with constructive logic in the sense that classical logic is consistent (*i.e.*, $\bot\ true$ is not provable) if and only if constructive logic is consistent, since $\neg\neg\bot$ is logically equivalent to $\bot$. As we have shown that constructive logic is consistent (*i.e.*, $\bot\ true$ is unprovable), we conclude that classical logic is also consistent.

Instead of proving Theorem 1.2 directly, we give another translation that converts a proof term M of type A in classical logic into a proof term M° of type A° in constructive logic. The translation is usually called *the CPS (Continuation-Passing Style) translation*, which enables us to simulate $\mathsf{callcc}\ x : A\ \mathit{false}.\, M$ and $\mathsf{throw}\ M\ \mathsf{to}\ x$ with λ-abstractions. The main idea in the CPS translation is to interpret $\neg A = A \supset \bot$ as the type of a *continuation* for type A, which, when invoked with an argument of type A, initiates the rest of the evaluation. Note that an invocation of a continuation conceptually returns an "answer" but actually never returns, for if it did, it would return a value of type $\bot$, which is impossible.

The CPS translation is obtained by translating a proof of $\Gamma; \Delta \vdash_{\mathsf{K}} M : C$ to a proof of $\Gamma^\circ, \neg\Delta^\circ \vdash_{\mathsf{I}} M^\circ : \neg\neg C^\circ$ where Γ° and $\neg\Delta^\circ$ are defined as follows:

$$\begin{aligned} \Gamma^\circ &= \{x : A^\circ \mid x : A \in \Gamma\} \\ \neg\Delta^\circ &= \{x : \neg A^\circ \mid x : A\ \mathit{false} \in \Delta\} \end{aligned}$$

Theorem 1.3 (CPS translation). *If $\Gamma; \Delta \vdash_{\mathsf{K}} M : C$, there exists a proof term M° such that $\Gamma^\circ, \neg\Delta^\circ \vdash_{\mathsf{I}} M^\circ : \neg\neg C^\circ$.*

Proof. By induction on the structure of the proof of $\Gamma; \Delta \vdash_{\mathsf{K}} M : C$. The proof reuses metavariables M and C.

In each case, we only specify M°, which can be shown to satisfy $\Gamma^\circ, \neg\Delta^\circ \vdash_{\mathsf{I}} M^\circ : \neg\neg C^\circ$ by straightforward structural induction. M° is given as a λ-abstraction $\lambda k{:}C^\circ \supset \bot.\, \cdots$ (or equivalently $\lambda k{:}\neg C^\circ.\, \cdots$), where k can be thought of as a continuation expecting the result of evaluating M. A typical pattern in the CPS translation is that a proof term of type $\bot$ (returning an "answer") is built from a proof term N of type A by applying M° to a continuation $\lambda x{:}A^\circ.\, N'$ (as in $M^\circ\ (\lambda x{:}A^\circ.\, N')$) so that x is bound to the result of evaluating M and the evaluation of N' returns the final "answer."

Case $\dfrac{x : A \in \Gamma}{\Gamma; \Delta \vdash_{\mathsf{K}} x : A}\ \mathsf{Hyp}$

$x^\circ = \lambda k{:}A^\circ \supset \bot.\, k\ x$

Case $\dfrac{\Gamma; \Delta \vdash_{\mathsf{K}} M : A \quad \Gamma; \Delta \vdash_{\mathsf{K}} N : B}{\Gamma; \Delta \vdash_{\mathsf{K}} (M, N) : A \wedge B}\ \wedge\mathsf{I}$

$(M, N)^\circ = \lambda k{:}(A^\circ \wedge B^\circ) \supset \bot.\, M^\circ\ (\lambda x{:}A^\circ.\, N^\circ\ (\lambda y{:}B^\circ.\, k\ (x, y)))$

Case $\dfrac{\Gamma; \Delta \vdash_{\mathsf{K}} M : A \wedge B}{\Gamma; \Delta \vdash_{\mathsf{K}} \mathsf{fst}\ M : A}\ \wedge\mathsf{E_L}$

$(\mathsf{fst}\ M)^\circ = \lambda k{:}A^\circ \supset \bot.\, M^\circ\ (\lambda x{:}A^\circ \wedge B^\circ.\, k\ (\mathsf{fst}\ x))$

Case $\dfrac{\Gamma; \Delta \vdash_{\mathsf{K}} M : A \wedge B}{\Gamma; \Delta \vdash_{\mathsf{K}} \mathsf{snd}\ M : B}\ \wedge\mathsf{E_R}$

$(\mathsf{snd}\ M)^\circ = \lambda k{:}B^\circ \supset \bot.\, M^\circ\ (\lambda x{:}A^\circ \wedge B^\circ.\, k\ (\mathsf{snd}\ x))$

Case $\dfrac{\Gamma; \Delta \vdash_{\mathsf{K}} M : A}{\Gamma; \Delta \vdash_{\mathsf{K}} \mathsf{inl}_B\ M : A \vee B}\ \vee\mathsf{I_L}$

$(\mathsf{inl}_B\ M)^\circ = \lambda k{:}(A^\circ \vee B^\circ) \supset \bot.\, M^\circ\ (\lambda x{:}A^\circ.\, k\ (\mathsf{inl}_{B^\circ}\ x))$

Case $\dfrac{\Gamma; \Delta \vdash_{\mathsf{K}} M : B}{\Gamma; \Delta \vdash_{\mathsf{K}} \mathsf{inr}_A\ M : A \vee B}\ \vee\mathsf{I_R}$

$(\mathsf{inr}_A\ M)^\circ = \lambda k{:}(A^\circ \vee B^\circ) \supset \bot.\, M^\circ\ (\lambda x{:}B^\circ.\, k\ (\mathsf{inr}_{A^\circ}\ x))$

Case $\dfrac{\Gamma; \Delta \vdash_{\mathsf{K}} M : A \vee B \quad \Gamma, x_1 : A; \Delta \vdash_{\mathsf{K}} N_1 : C \quad \Gamma, x_2 : B; \Delta \vdash_{\mathsf{K}} N_2 : C}{\Gamma; \Delta \vdash_{\mathsf{K}} \mathsf{case}\ M\ \mathsf{of}\ \mathsf{inl}\ x_1 \Rightarrow N_1 \mid \mathsf{inr}\ x_2 \Rightarrow N_2 : C}\ \vee\mathsf{I_R}$

$(\mathsf{case}\ M\ \mathsf{of}\ \mathsf{inl}\ x_1 \Rightarrow N_1 \mid \mathsf{inr}\ x_2 \Rightarrow N_2)^\circ = \lambda k{:}C^\circ \supset \bot.\, M^\circ\ (\lambda x{:}A^\circ \vee B^\circ.\, \mathsf{case}\ x\ \mathsf{of}\ \mathsf{inl}\ x_1 \Rightarrow {N_1}^\circ\ k \mid \mathsf{inr}\ x_2 \Rightarrow {N_2}^\circ\ k)$

Case $\dfrac{\Gamma, x : A; \Delta \vdash_{\mathsf{K}} M : B}{\Gamma; \Delta \vdash_{\mathsf{K}} \lambda x{:}A.\, M : A \supset B}\ \supset\mathsf{I}$

$(\lambda x{:}A.\, M)^\circ = \lambda k{:}(A^\circ \supset \neg\neg B^\circ) \supset \bot.\, k\ (\lambda x{:}A^\circ.\, M^\circ)$

Case $\dfrac{\Gamma; \Delta \vdash_{\mathsf{K}} M : A \supset B \quad \Gamma; \Delta \vdash_{\mathsf{K}} N : A}{\Gamma; \Delta \vdash_{\mathsf{K}} M\ N : B}$

$(M\ N)^\circ = \lambda k{:}B^\circ \supset \bot.\, M^\circ\ (\lambda x{:}A^\circ \supset \neg\neg B^\circ.\, N^\circ\ (\lambda y{:}A^\circ.\, x\ y\ k))$

Case $\dfrac{}{\Gamma; \Delta \vdash_{\mathsf{K}} \langle\rangle : \top}$

$\langle\rangle^\circ = \lambda k{:}\top^\circ \supset \bot.\, k\ \langle\rangle$

Case $\dfrac{\Gamma; \Delta \vdash_{\mathsf{K}} M : \bot}{\Gamma; \Delta \vdash_{\mathsf{K}} \mathsf{abort}_C\ M : C}$

$(\mathsf{abort}_C\ M)^\circ = \lambda k{:}C^\circ \supset \bot.\, M^\circ\ (\lambda x{:}\bot.\, x)$

Case $\dfrac{\Gamma; \Delta, x : A\ \mathit{false} \vdash_{\mathsf{K}} M : A}{\Gamma; \Delta \vdash_{\mathsf{K}} \mathsf{callcc}\ x : A\ \mathit{false}.\, M : A}$

$(\mathsf{callcc}\ x : A\ \mathit{false}.\, M)^\circ = \lambda k{:}A^\circ \supset \bot.\, [k/x]M^\circ\ k$

Case $\dfrac{\Gamma; \Delta \vdash_{\mathsf{K}} M : A \quad x : A\ \mathit{false} \in \Delta}{\Gamma; \Delta \vdash_{\mathsf{K}} \mathsf{throw}\ M\ \mathsf{to}\ x : C}$

$(\mathsf{throw}\ M\ \mathsf{to}\ x)^\circ = \lambda k{:}C^\circ \supset \bot.\, M^\circ\ x$ □