Introduction to Logic

1 What is Logic?

The word **logic** comes from the Greek *logos*, which can be translated as *reason*. Logic as a discipline is about studying the fundamental principles of how to reason correctly, i.e., making valid *arguments* (in the sense of making a case for something, not an acrimonious dispute). In other words, logic tells us, given certain assumptions, what conclusions one can reasonably infer. Consider the following example (adapted from The Simpsons):

```
 Lisa: By your logic I could claim that this rock keeps tigers away.
Homer: Oh, how does it work?
 Lisa: It doesn't work...it's just a stupid rock. But I don't see any tigers around, do you?
Homer: Lisa, I want to buy your rock.
```

Clearly (and unsurprisingly) Homer is not reasoning correctly. Logic can tell us exactly why his reasoning is incorrect and thus why Lisa is so exasperated with him. Logic has been studied since ancient times, with schools arising thousands of years ago in Greece, India, and China. In Europe, logic was strongly influenced by Aristotle all the way up through the 19th century. Logic was part of the Trivium, and along with grammar and rhetoric formed the foundations of a medieval liberal arts education. Modern formal logic started taking shape in the mid 1800s due to people like George Boole (boolean algebra), Gottlob Frege (predicate logic), and many others. An important principle of formal logic is **form over content**. That is, the rules of logic are about the form that an argument takes, not the contents of that argument. Consider the following two examples:

```
   Premise: If one is a man then one is mortal
   Premise: Socrates is a man
Conclusion: Socrates is mortal

   Premise: If x > y then x+1 > y
   Premise: 1 > 0
Conclusion: 2 > 0
```

One example is about the mortality of man, the other is about math. They have completely different content, but the *form* of both arguments is the same:

```
   Premise: If A then B
   Premise: A
Conclusion: B
```

This particular form of argument is called *modus ponens*. In formal logic we only study the forms of arguments and ignore content. This makes formal logic seem like just a bunch of symbol pushing, devoid of meaning—and that's exactly what it is! In fact, that is what makes formal logic so powerful. We can study the correct forms of argument without worrying about what those arguments are about, which means that logic applies to *all* kinds of arguments no matter what their subject matter might be. We don't need separate logics for studying, e.g., arguments about mortality versus arguments about math.

1.1 Logical Systems

Logic is not one single thing; there are many different systems of logic with different applications. These systems fall into different categories such as **propositional logics**, **predicate logics**, **modal logics**, etc. Examples of predicate logics include *first-order logic*, *higher-order logic*, *many-sorted logic*, and *infinitary logic*. Examples of modal logics include *temporal logic* (logics of time), *alethic logic* (logics of possibility and necessity), *deontic logic* (logics of obligation and permission), and *doxastic logic* (logics of belief). There are many other logical systems that have been developed. For all logical systems there are three desirable properties, though not all logical systems will have all three:

- **Consistency.** The system cannot be used to prove things that contradict each other, i.e., a consistent system cannot be used to prove for some proposition A that A is true *and* that A is not true.
- **Soundness.** If the system can be used to prove a proposition A, then A is true. In other words, only true things can be proven.
- **Completeness.** If proposition A is true, then the system can be used to prove A. In other words, all true things can be proven.

Consistency and soundness are generally taken as required in order to have a sensible logical system; when a system cannot have all three it is completeness that is dropped.

2 Logical Interpretations

The study of logic is about form, not content. However, we do need some way to connect the abstract logical sentences with meaningful content, or logic would be completely disconnected from reality. This connection is made via an *interpretation*. Consider the following definitions:

- **Domain of discourse** (a.k.a. *universe of discourse*). A domain of discourse is a set of objects that we are interested in making arguments about. This set can consist of anything we want: people, numbers, emotions, planets, nations, colors, etc.
- **Interpretation.** An interpretation maps logical syntax (the symbols used to make logical formulae) to the domain of discourse. Take the modus ponens example above: from 'If A then B' and 'A' we can infer 'B'. We can take the domain of discourse to be statements about people and states of being and the interpretation would map A to "Socrates is a man" and B to "Socrates is mortal". Alternatively, we can take the domain of discourse to be arithmetic expressions and the interpretation would map A to "$1 > 0$" and B to "$2 > 0$". Both of these interpretations provide different meanings to the same original logical formula.

Based on these definitions, we can now characterize logical sentences (or *formulae*) as one of the following:

- **Valid.** A valid formula is guaranteed to be true no matter what interpretation we use. For example, the logical sentence "If A is true then A is true" is valid.
- **Satisfiable.** A satisfiable formula has at least one interpretation that makes it true. For example, the logical sentence "A and B are both true" is satisfiable, but not valid. Under the interpretation A = "triangles have three sides" and B = "squares have four sides", the sentence is true. Under the interpretation A = "triangles have four sides" and B = "squares have three sides" the sentence is not true.
- **Unsatisfiable.** An unsatisfiable formula has no interpretations that make it true. For example, the sentence "A is true and A is not true" cannot be true in any interpretation.

In Computer Science we are often more interested in satisfiability than validity. In particular, we often would like to take a given formula and ask "is this satisfiable?", and further "given a particular domain of discourse, what specific interpretation satisfies the formula?". Finding a satisfying interpretation for a formula is a form of computation.

2.1 More Interpretation Examples

The notions of "domain of discourse" and "interpretation" may be a little difficult to comprehend. Here are some examples to illustrate the concepts. Take the following sentence in first-order logic:

$$\forall x. p(x, f(x, g))$$

Here x is a variable, p is a predicate, f is a binary function (i.e., a function taking two arguments), and g is a nullary function (i.e., a function taking no arguments, also called a *constant*). The $\forall$ symbol is saying that the variable x ranges over all elements of the domain of discourse. If these concepts are not familiar to you, take a look at the section on first-order logic below where they will be explained in more detail and then come back. An interpretation will take all of the predicates and functions in a formula and map them to relations and functions over the given domain of discourse, as in the following examples:

Interpretation 1. We will set the domain of discourse to be people. We will map the predicate p to the relation between people `descendant-of`, the function f to the function on people `least-common-ancestor` (it returns the most closely related person who is an ancestor of both people given as arguments), and the constant g to the person `mary`. Then in this interpretation, x is ranging over the domain of people and the above formula is saying "for all people x, x is a descendant of the least common ancestor of x and `mary`". For convenience, we will assume that everyone has a common ancestor with `mary`.

Interpretation 2. We will set the domain of discourse to be integers. We will map the predicate p to the relation between numbers $<$ (less than), the function f to the function on numbers $+$ (addition), and the constant g to the number 1. Then in this interpretation, x is ranging over the domain of integers and the above formula is saying "for all numbers x, x is less than $x + 1$".

Interpretation 3. We can have multiple interpretations for the same domain. Again set the domain of discourse to be integers. We will map the predicate p to the relation between numbers $>$ (greater than), the function f to the function on numbers $-$ (subtraction), and the constant g to the number 1. Then in this interpretation, x is ranging over the domain of integers and the above formula is saying "for all numbers x, x is greater than $x - 1$".

3 First-Order Logic

One of the most commonly-used systems of logic is *first-order predicate logic*. The syntax of first-order logic formulae is given below:

$$x \in \mathit{Variable} \qquad f \in \mathit{Function} \qquad p \in \mathit{Predicate}$$

$$t \in \mathit{Term} ::= x \mid f(\vec{t})$$

$$A, B \in \mathit{Proposition} ::= p(\vec{t}) \mid \top \mid \bot \mid A \wedge B \mid A \vee B \mid A \supset B \mid \forall x.A \mid \exists x.A$$

In this notation, $\vec{t}$ is a sequence of terms of length equal to the arity of the given function or predicate symbol. When we need multiple variables we'll use letters u, v, w, x, y, z. When we need multiple function symbols we'll use f, g, h, i, j, k. When we need multiple predicate symbols we'll use m, n, o, p, q, r, s. When we need multiple propositions we'll use A, B, C, D, E. As a last resort, we'll use subscripts to distinguish names, so that, for example, x_1 and x_2 indicate different variables.

Arity. Every function and predicate symbol has an *arity*, which is simply the number of arguments that it accepts. For example, a function $f(u, v)$ has arity 2 and a function $g(u, v, w)$ has arity 3. We'll sometimes indicate the arity using notation such as $f/2$ (for function symbol f with arity 2) and $g/3$ (for function symbol g with arity 3). Functions and predicates can have arity 0, in which case we write them without parentheses. For example, if we have a predicate symbol $p/0$ we'll write p instead of $p()$. Function symbols with 0 arity are called *constants* and predicate symbols with 0 arity are called *propositional variables*.

Function Symbols. Function symbols f, g, etc. stand for functions that map objects to other objects. Of course, without an interpretation we don't know what those functions and objects are—an *interpretation* will include a specific domain of objects and map each function symbol to a function over that domain. For example, one interpretation might specify that the objects are people, and connect the function symbol *fatherOf*/1 with the function that takes a person and returns that person's father, e.g., *fatherOf*(Mary) = John. Terms, which as indicated above are made up of variables and functions over terms, are just names for objects in some unknown domain. They don't have any inherent meaning until we give them an interpretation, which will provide a specific domain of objects. Different interpretations may give them different meanings.

Predicate Symbols. Predicate symbols p, q, etc. stand for relations between objects. Recall that a relation is a set of tuples. For example, we might define the relation $\{(grass, green), (sky, blue), (apple, red)\}$. Again, without an interpretation we don't know what the relations or objects are; an interpretation will map each predicate symbol to a specific relation over the domain of objects for that interpretation. For example, one interpretation might map the predicate symbol *color*/2 to the relation defined above.

Example Terms: (denoting some object in a domain)

- x
- $f(c, g(a, b))$
- $f(g(h(x, y), i, j(x, z)), h(y, z))$

Terminology note: a *ground term* is a term that does not contain any variables. Of the three examples above, only the second one is a ground term.

Example Propositions: (having a truth value of *true* or *false*)

- $\forall x.(q(x) \vee \neg p(x))$
- $\exists x.q(x, f(x), g) \wedge s(x) \supset \forall x.r(h, x)$
- $\forall x \exists y.(r(x, y) \supset r(y, x))$

3.1 Informal Meaning of First-Order Logic

Now we will informally describe the meaning of the logical connectives being used in first-order logic formulae, i.e., $\wedge$, $\vee$, $\supset$, $\forall$, and $\exists$. We will give a precise, formal description of these connectives in the section below about natural deduction; the intent here is simply to provide some intuition. You may notice that we have not included a logical connective that is often used in first-order logic: $\neg$, i.e., negation. Rather than defining negation directly, we will assume that a negated proposition $\neg A$ is actually an abbreviation for the formula $A \supset \bot$. This will work out to give exactly the same semantics for negation that we would have defined directly, but simplifies the description of the first-order logic system.

A proposition is a logical statement with a truth value; that is, the statement can be either true or false. The statement "2+2 = 4" is a proposition (as is the statement "2+2 = 5"). The expression $\top$ (pronounced top or true) is the proposition that is trivially true; the expression $\bot$ (pronounced bottom or false or absurd) is the proposition that is trivially false. A predicate $p(\vec{t})$ (e.g., the predicate `color(sky, purple)`) stands for some relation over the domain of discourse. Without an interpretation its truth value depends solely on what assumptions we might have already made (this notion will be made more precise in the section on natural deduction). The logical connectives, then, are ways to combine propositions together into a new proposition:

- $A \wedge B$, pronounced "A and B". This form of proposition is called a *conjunction*; the propositions A and B are the *conjuncts*. The proposition $A \wedge B$ is true iff (if and only if) both proposition A is true and proposition B is true; otherwise it is false.

- $A \vee B$, pronounced "A or B". This form of proposition is called a *disjunction*; the propositions A and B are the *disjuncts*. The proposition $A \vee B$ is true iff at least one of proposition A and proposition B are true, i.e., either A is true, B is true, or both are true. Otherwise it is false.

- $A \supset B$, pronounced "A implies B". This form of proposition is called an *implication*; the proposition A is called the *antecedent* and the proposition B is called the *succedent*. The proposition $A \supset B$ is true iff whenever proposition A is true, proposition B is necessarily true as well. In other words, A cannot be true unless B is also true (though the reverse does not necessarily hold).

- $\forall x \,.\, A$, pronounced "for all x, A". This form of proposition is called *universal quantification*; the $\forall$ symabol is called the universal quantifier. Typically the proposition A will mention the variable x, otherwise there is no point in having the quantifier. The variable x ranges over the entire (unspecified) domain of discourse, so, e.g., the proposition $\forall x \,.\, p(x)$ means that the predicate p is true for all elements in the domain of discourse.

  Another way to think about the universal quantifier is in terms of conjunction. If we label the elements of the (unspecified) domain of discourse $d_1, d_2, \ldots, d_n$ then we can think of the proposition $\forall x \,.\, p(x)$ as the proposition $p(d_1) \wedge p(d_2) \wedge \ldots \wedge p(d_n)$.

- $\exists x \,.\, A$, pronounced "there exists x such that A". This form of proposition is called *existential quantification*; the $\exists$ symabol is called the existential quantifier. Typically the proposition A will mention the variable x, otherwise there is no point in having the quantifier. The variable x represents some element of the (unspecified) domain of discourse, so, e.g., the proposition $\exists x \,.\, p(x)$ means that the predicate p is true for at least one element in the domain of discourse.

  Another way to think about the existential quantifier is in terms of disjunction. If we label the elements of the (unspecified) domain of discourse $d_1, d_2, \ldots, d_n$ then we can think of the proposition $\exists x \,.\, p(x)$ as the proposition $p(d_1) \vee p(d_2) \vee \ldots \vee p(d_n)$.

3.2 What Does "First-Order" Mean?

The term *first-order* in "first-order logic" refers to the fact that we are restricted to quantifying over the objects in a domain of discourse. That is, a variable x in a proposition will always refer to some object. In higher-order logics we are also allowed to quantify over predicates and functions, which means that a variable x may refer to some relation or function over objects in the domain, not just objects themselves. Here is a small example:

$$\forall x \,.\, x(\texttt{foo}) \supset x(\texttt{bar})$$

In this proposition, x is ranging over unary predicates. The proposition states that for any predicate x, if x is true of constant `foo` then it is also true of constant `bar`. We cannot make this statement in first-order logic because we are not allowed to quantify over predicates in this fashion. Higher-order logic is strictly more powerful than first-order logic.

4 Natural Deduction

There are several ways to formally define first-order logic. In Computer Science most people are introduced to it via the Hilbert-style axiomatic formulation; in philosophy most people learn it via the Natural Deduction formulation. There are also other formulations possible, such as the sequent calculus. These all define the same thing, they just provide different perspectives and different ways of getting to the same end. For reasons that will become clear later in the course, we'll use the natural deduction style.

The fundamental notion of natural deduction is a **judgement** on the truth of a **proposition** based on **evidence**. A *proposition* is something that can be either true or false, e.g., "it is raining". A *judgement* says whether a proposition is true based on some evidence (e.g., observation, or a derivation from known facts). For example, we could have the proposition "it is raining" and the judgement "the proposition 'it is raining' is true", based on the evidence that I can see it raining. Propositions are given as formulae in the syntax of first-order logic.

4.1 Making Judgements

Given a proposition, we want to be able to make a judgement about it. There are a number of different kinds of judgements that we could make, but we'll focus on one of the most important: *truth*. We make judgements based on evidence. Some evidence will be given to us as facts (i.e., *axioms*). Other evidence will come from derivations based on *inference rules*. These rules give us guidelines for how to make new judgements based on existing judgements. Judgements will often make use of *hypotheses*, i.e., propositions that we will temporarily assume are true while trying to make the judgement. A hypothesis is just a sequence of propositions; we will symbolize arbitrary hypotheses using the Greek letters Γ and Δ.

A judgement will be of the form '$\Gamma \vdash A$'; this says that if we assume the propositions contained in Γ are true, then we are justified in saying that proposition A is true. For example, we can make the judgement '$A, B \vdash A \wedge B$', i.e., "if we assume that proposition A is true and proposition B is true, then we can infer that proposition $A \wedge B$ is true".

4.2 Axioms

For classical first-order logic there are two axioms, i.e., judgements that we accept as true without further evidence:

- $\Gamma, A \vdash A$, i.e., we always know that under the assumption that A is true, we can conclude that A is true. The notation 'Γ, A' means that we're appending the proposition A to the list of propositions Γ.

- $\Gamma \vdash A \vee \neg A$, i.e., the law of the excluded middle: it is always true, under any set of assumptions (including the empty set), that either A is true or $\neg A$ is true.

Note that we could equivalently replace the law of the excluded middle with the law of double negation: $\Gamma \vdash \neg\neg A \supset A$. It turns out that given either axiom we can infer the other, so it doesn't matter which one we take as fundamental. If we remove this second axiom (in either form), then instead of *classical* first-order logic we have what is known as *intuitionistic* first-order logic. This is a deep philosophical choice with many implications, which we will discuss further in the section below on intuitionistic logic.

4.3 Inference Rules

Inference rules are just a compact way of writing `if..then` statements. They consist of a horizontal line with zero or more judgements on top of the line, called *premises*, and exactly one judgement on the bottom of the line, called the *conclusion*. An inference rule is saying that *if* all of the premises can be proven true, *then* the conclusion must also be true. Each inference rule will also have a name, given immediately to the right of the horizontal line.

For each logical connective $\wedge, \vee, \supset, \forall, \exists$ there are rules that tell us how we can use them to make judgements. We don't include $\neg$ as a logical connective; we could, but it's easier to just say that $\neg A$ is shorthand for writing $A \supset \bot$. Each connective has an *introduction rule* that shows how we can judge that a proposition using that connective is true (i.e., the connective is used in the conclusion judgement). Each connective has an *elimination rule* that shows how we can judge that a proposition is true based on knowing some other proposition using that connective is true (i.e., the connective is used in one of the premise judgements). The rules below mention assumptions Γ, but they don't specify or use the contents of Γ; it is there only to make clear that the rules are valid no matter what assumptions we're making.

If you are having trouble understanding these inference rules, refer back to the informal description of these operators in the section on informal meaning of first-order logic. Having a conceptual understanding of what the connectives mean should help in understanding the formal definitions below.

Logical Connective	Introduction Rule	Elimination Rule
Conjunction ($\wedge$)	$\dfrac{\Gamma \vdash A \qquad \Gamma \vdash B}{\Gamma \vdash A \wedge B}\ (\wedge\mathrm{I})$	$\dfrac{\Gamma \vdash A \wedge B}{\Gamma \vdash A}\ (\wedge\mathrm{E}_1)$ $\dfrac{\Gamma \vdash A \wedge B}{\Gamma \vdash B}\ (\wedge\mathrm{E}_2)$
Disjunction ($\vee$)	$\dfrac{\Gamma \vdash A}{\Gamma \vdash A \vee B}\ (\vee\mathrm{I}_1)$ $\dfrac{\Gamma \vdash B}{\Gamma \vdash A \vee B}\ (\vee\mathrm{I}_2)$	$\dfrac{\Gamma \vdash A \vee B \quad \Gamma, A \vdash C \quad \Gamma, B \vdash C}{\Gamma \vdash C}\ (\vee\mathrm{E})$
Implication ($\supset$)	$\dfrac{\Gamma, A \vdash B}{\Gamma \vdash A \supset B}\ (\supset\mathrm{I})$	$\dfrac{\Gamma \vdash A \supset B \qquad \Gamma \vdash A}{\Gamma \vdash B}\ (\supset\mathrm{E})$
Universal Quantification ($\forall$)	$\dfrac{\Gamma \vdash A[x \mapsto k] \qquad k \text{ fresh}}{\Gamma \vdash \forall x.A}\ (\forall\mathrm{I})$	$\dfrac{\Gamma \vdash \forall x.A}{\Gamma \vdash A[x \mapsto t]}\ (\forall\mathrm{E})$
Existential Quantification ($\exists$)	$\dfrac{\Gamma \vdash A[x \mapsto t]}{\Gamma \vdash \exists x.A}\ (\exists\mathrm{I})$	$\dfrac{\Gamma \vdash \exists x.A \qquad \Gamma, A[x \mapsto k] \vdash B \qquad k \text{ fresh}}{\Gamma \vdash B}\ (\exists\mathrm{E})$

Conjunction. The introduction rule $\wedge$I states that if we know A is true under some set of assumptions Γ and we know B is true under the same set of assumptions Γ, then we can also conclude $A \wedge B$ is true under that same set of assumptions. There are two elimination rules; they state that if we know $A \wedge B$ is true under some set of assumptions Γ then we can conclude both that A is true (using the first rule $\wedge\mathrm{E}_1$) and that B is true (using the second rule $\wedge\mathrm{E}_2$) under the same set of assumptions Γ. Constantly mentioning the assumptions Γ is tedious, so we'll often ignore them in the explanations below, but for all of the inference rules the assumptions used in the premises and conclusions must match up for the rule to be used.

Notice how the introduction and elimination rules are duals of each other. The introduction rule takes two pieces of information (the truth of A and B) and packages them up into a single piece of information (the truth of $A \wedge B$). The elimination rules take a single piece of information (the truth of $A \wedge B$) and extract from it two pieces of information (the truth of A and the truth of B). There is a principle of conservation of information here: information is neither lost nor destroyed when using the introduction and elimination rules. A similar observation will hold for all of the other connectives.

Disjunction. The introduction rules state that if we know A is true then we can conclude that $A \vee B$ is true (using the first rule $\vee\mathrm{I}_1$) and that if we know B is true then we can also conclude $A \vee B$ is true (using the second rule $\vee\mathrm{I}_2$). Notice that we conclude the same thing in either case; thus simply knowing $A \vee B$ is true does not give us any information about which of A or B was true.

The elimination rule $\vee$E shows how to use the fact $A \vee B$ is true without knowing which one of A or B was actually true. The first premise states that we know $A \vee B$. The second premise states that if we assume A is true, we can infer some new proposition C. The third premise states that if we assume B is true, we can infer that same new proposition C. The elimination rule, then, states that if we know $A \vee B$ is true, and we know that if A is true then C is true and also that if B is true then C is true, then we can safely conclude C is true without knowing *which* of A or B was actually true.

Implication. The introduction rule states that if by assuming A is true we can infer B is true, then we can conclude that $A \supset B$ is true. The elimination rule states that if we know $A \supset B$ is true and we can infer A is true, then we can conclude that B is true.

Universal Quantification. We assume that proposition A mentions variable x, otherwise we can trivially remove the quantification and leave A by itself. The notation '$A[x \mapsto k]$' means to return a new version of A such that all mentions of variable x are replaced by k, e.g., $(f(x) \wedge g(x))[x \mapsto h]$ would result in $f(h) \wedge g(h)$. The term *fresh* means that the constant k has never been used anywhere in the current proof, so this is the first time in the current proof that k has been mentioned.

The introduction rule states that if we can infer $A[x \mapsto k]$ is true for some fresh (i.e., never-seen-before) constant k, then we can conclude that $\forall x \,.\, A$ is true. The reasoning behind this rule is that if k is a fresh constant that we've never seen before, then we know nothing about it, which means that it could represent any object in the domain of discourse. If we can prove that $A[x \mapsto k]$ is true, and k can represent any object in the domain, then A must be true no matter what domain object we substitute for x. Therefore, A must be true for all objects in the domain, which is represented as $\forall x \,.\, A$.

The elimination rule states that if $\forall x \,.\, A$ is true, then we can conclude $A[x \mapsto t]$ is true for any arbitrary term t. Recall that terms specify objects in the domain of discourse; if A is true for all objects, then it is true for any one object.

Existential Quantification. Again we assume that A mentions variable x. The introduction rule states that if we know $A[x \mapsto t]$ is true for some term t, then we can conclude that $\exists x \,.\, A$ is true. The reasoning is that we have shown t makes A true, and thus we have demonstrated that there is at least one object in the domain of discourse that makes A true.

The elimination rule states that if $\exists x \,.\, A$ is true and, by assuming $A[x \mapsto k]$ for some fresh constant k we can infer that B is true, then we can conclude that B is true. The reasoning is similar to that of the disjunction elimination rule—we know that there is some object that makes A true, but we don't know which object. Therefore we select a fresh constant k that can represent any object (as discussed in the introduction rule for universal quantification ∀I), assume A is true for that object, and attempt to infer B. If we are successful, then we know that it doesn't matter which object satisfies A, we can safely conclude B.

4.4 Proof Examples

Here we give several examples of how to prove a judgement about a given proposition. Note in the following proofs that the desired conclusion is at the bottom of the proof. We can think of these proofs as *derivation trees* rooted in the conclusion, growing upwards according to the appropriate introduction and elimination rules. The leaves of the tree are judgements that are trivially true; in this case, they are all of the form $\Gamma, A \vdash A$.

4.4.1 Example 1

We will prove the following judgement: $\vdash p \wedge q \supset q \wedge p$.

$$\dfrac{\dfrac{\dfrac{p \wedge q \vdash p \wedge q}{p \wedge q \vdash q}\,{\scriptstyle \wedge E_1} \qquad \dfrac{p \wedge q \vdash p \wedge q}{p \wedge q \vdash p}\,{\scriptstyle \wedge E_2}}{p \wedge q \vdash q \wedge p}\,{\scriptstyle \wedge I}}{\vdash p \wedge q \supset q \wedge p}\,{\scriptstyle \supset I}$$

4.4.2 Example 2

We will prove the following judgement: $\vdash p \supset (q \supset (p \wedge q))$.

$$\dfrac{\dfrac{\dfrac{p, q \vdash p \qquad p, q \vdash q}{p, q \vdash p \wedge q}\,{\scriptstyle \wedge I}}{p \vdash q \supset (p \wedge q)}\,{\scriptstyle \supset I}}{\vdash p \supset (q \supset (p \wedge q))}\,{\scriptstyle \supset I}$$

4.4.3 Example 3

We will prove the following judgement: $\vdash (p \supset q) \wedge (p \supset r) \supset (p \supset (q \wedge r))$. In the following proof, in order to fit the proof on the page we will abbreviate the assumption $(p \supset q) \wedge (p \supset r), p$ as Γ.

$$\dfrac{\dfrac{\dfrac{\dfrac{\Gamma \vdash p \qquad \dfrac{\Gamma \vdash (p \supset q) \wedge (p \supset r)}{\Gamma \vdash p \supset q}\,{\scriptstyle \wedge E_2}}{\Gamma \vdash q}\,{\scriptstyle \supset E} \qquad \dfrac{\Gamma \vdash p \qquad \dfrac{\Gamma \vdash (p \supset q) \wedge (p \supset r)}{\Gamma \vdash p \supset r}\,{\scriptstyle \wedge E_1}}{\Gamma \vdash r}}{(p \supset q) \wedge (p \supset r), p \vdash q \wedge r}\,{\scriptstyle \wedge I}}{(p \supset q) \wedge (p \supset r) \vdash p \supset (q \wedge r)}\,{\scriptstyle \supset I}}{\vdash (p \supset q) \wedge (p \supset r) \supset (p \supset (q \wedge r))}\,{\scriptstyle \supset I}$$

4.4.4 Example 4

In this example we will illustrate the importance of using a fresh k in the universal quantification introduction rule by giving an **incorrect** proof of the following judgement: $\vdash \forall x \forall y. p(x) \supset p(y)$. This judgement is obviously wrong; the proof will be incorrect because it fails to properly use a fresh k.

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{p(k) \vdash p(k)}{p(k) \vdash \forall x.p(x)}\ \forall I}{p(k) \vdash p(j)}\ \forall E}{\vdash p(k) \supset p(j)}\ \supset I}{\vdash \forall y.p(k) \supset p(y)}\ \forall I}{\vdash \forall x \forall y.p(x) \supset p(y)}\ \forall I
$$

The problem is at the top of the derivation tree, where we used the $\forall I$ rule to turn $p(k)$ into $\forall x.p(x)$. We can't do that because k isn't fresh—it's used in the hypothesis of the top judgement.

5 Intuitionistic Logic

From one perspective, intuitionistic logic is a small change from classical first-order logic. We simply remove the law of the excluded middle ($\Gamma \vdash A \vee \neg A$) and the law of double negation ($\Gamma \vdash \neg\neg A \supset A$) as axioms. Everything else remains exactly the same. However, there is a deep philosophical choice being made when we do this.

Intuitionistic logic was developed by Heyting (1898–1980) in order to formalize Brouwer's program of intuitionism. Brouwer (1881–1966) believed that mathematics is completely a creation of the human mind, rather than an external reality that we explore. That is, math is invented, not discovered. Along with this belief comes a new notion of what it means for a mathematical statement to be true: to prove that a mathematical object exists, one must provide a *constructive* proof—that is, a method for constructing said object. This is in contrast to classical mathematics, which allows one to prove something exists by proving that it can't not exist. The constructive viewpoint is interesting from a Computer Science perspective because a constructive proof is akin to an *algorithm*.

If we think in terms of the traditional notions of true and false, intuitionistic logic doesn't seem to make sense. A proposition must be either true or false; if it isn't true then it must be false, and vice-versa. Similarly, if we know that it's false that a proposition is false, then the proposition must be true. This reasoning seems like common-sense, and are exactly what the law of the excluded middle and the law of double negation are stating.

However, intuitionism forces us to think in terms of *provability* rather than truth, or in other words, something is only true if it is provable. Consider the famous statement from algorithmic complexity theory $\mathbf{P} = \mathbf{NP}$. Clearly this statement is either true or false. However, we do not have a *proof* either way—we don't know whether it is true or false. From an intuitionistic standpoint the proposition $A \vee \neg A$ is saying that either we have a proof of A or we have a proof of $\neg A$; however we have just given an example where this is not true (and there are many other examples of statements such that we neither have a proof nor a refutation). Thus, the law of the excluded middle does not universally hold from the standpoint of intuitionistic logic.

Thinking in terms of provability also explains why the law of double negation doesn't hold. The proposition $\neg\neg A$ states that we have a proof that $\neg\neg A$ does not hold. However, this fact does not immediately give us a way to construct A, and therefore we don't have a constructive proof that A holds. Thus, there is no intuitionistic law of double negation.

5.1 Intuitionism and Computer Science

Intuitionism and the philosophy of constructive mathematics is not in the mainstream of the modern mathematical community. However, as hinted at earlier, intuitionism can have a strong impact on Computer Science. For example, as we will see when we discuss type systems, type theory and intuitionistic logic are, in a very real way, exactly the same thing. Whenever we write code in a statically-typed language, we are actually proving a theorem in intuitionistic logic, and when we compile that code the type checker is verifying that our theorem is correct. We will be exploring this connection in-depth during the coming weeks.