arXiv:1601.01483v1 [cs.LO] 7 Jan 2016

Rules and derivations in an elementary logic course

Gilles Dowek
Inria, 23 avenue d'Italie, CS 81321, 75214 Paris Cedex 13, France.
gilles.dowek@inria.fr

When teaching an elementary logic course to students who have a general scientific background but have never been exposed to logic, we have to face the problem that the notions of deduction rule and of derivation are completely new to them, and are related to nothing they already know, unlike, for instance, the notion of model, that can be seen as a generalization of the notion of algebraic structure.

In this note, we defend the idea that one strategy to introduce these notions is to start with the notion of inductive definition [1]. Then, the notion of derivation comes naturally. We also defend the idea that derivations are pervasive in logic and that defining precisely this notion at an early stage is a good investment to later define other notions in proof theory, computability theory, automata theory, ... Finally, we defend the idea that to define the notion of derivation precisely, we need to distinguish two notions of derivation: *labeled with elements* and *labeled with rule names*. This approach has been taken in [2].

1 From inductive definitions to derivations

1.1 A method to define sets: inductive definitions

Inductive definitions are a way to define subsets of a set A. The definition of a subset P is given by functions f_1, from A^{n_1} to A, f_2, from A^{n_2} to A, ... These functions are called *rules*. For example, the function $f_1 = \langle\rangle \mapsto 0$, from $\mathbb{N}^0$ to $\mathbb{N}$, and $f_2 = \langle a \rangle \mapsto a + 2$, from $\mathbb{N}^1$ to $\mathbb{N}$ are rules.

Instead of writing these rules $f_1 = \langle\rangle \mapsto 0$ and $f_2 = \langle a \rangle \mapsto a + 2$, we often write them

$$\frac{}{0}\, f_1$$

$$\frac{a}{a+2}\, f_2$$

but despite this new notation, rules are things the students know: functions.

To define the subset P, we first define a function F from $\mathcal{P}(A)$ to $\mathcal{P}(A)$ as follows

$$F(X) = \bigcup_i \{f_i(a_1, ..., a_{n_i}) \mid a_1, ..., a_{n_i} \in X\}$$

For example, the two rules above define the function

$$F(X) = \{0\} \cup \{a + 2 \mid a \in X\}$$

and, for instance, $F(\{4, 5, 6\}) = \{0, 6, 7, 8\}$, $F(\varnothing) = \{0\}$, $F(\{0\}) = \{0, 2\}$, ...

The function F monotonic and continuous, thus it has a smallest fixed point P which is the inductively defined subset of A. This smallest fixed point can be defined in two ways

$$P = \bigcap_{F(X) \subseteq X} X = \bigcup_i F^i(\varnothing)$$

The first definition characterizes the set P as the smallest set closed by f_1, f_2, ... the second as the set containing all the elements that can be built with these functions in a finite number of steps. The monotonicity and continuity of F and the two fixed points theorems are easy lemmas [2] for mathematically oriented students. They can be admitted without proof otherwise.

Continuing with our example, the set P of even numbers can be characterized as the smallest set containing 0 and closed by the function $a \mapsto a + 2$, or as the union of the sets $F(\varnothing) = \{0\}$, $F^2(\varnothing) = \{0, 2\}$, $F^3(\varnothing) = \{0, 2, 4\}$, ...

1.2 Derivations

Defining a *derivation* as a tree whose nodes are labeled with elements of A and such that if a node is labeled with x and its children with y_1, ..., y_n, then there exists a rule f such that $x = f(y_1, ..., y_n)$, and *a derivation of an element* a as a derivation whose root is labeled with a, we can prove, by induction on i, that all the elements of $F^i(\varnothing)$ have a derivation. The property is trivial for $i = 0$. If it holds for i and $a \in F^{i+1}(\varnothing)$, then by definition $a = f(b_1, ..., b_n)$ for some rule f and $b_1 \in F^i(\varnothing)$, ..., $b_n \in F^i(\varnothing)$, thus, by induction hypothesis, b_1, ..., b_n have derivations. Hence, so does a.

Thus, from the second property $P = \cup_i F^i(\varnothing)$, we get that all elements of P have derivations. Conversely, all elements that have a derivation are elements of P.

Continuing with our example the number 4 has the derivation

$$\cfrac{\cfrac{\overline{\;0\;}}{2}}{4}$$

1.3 Rule names

There are several alternatives for defining the notion of derivation. For instance, when $x = f(y_1, ..., y_n)$, instead of labelling the node just with x, we can label it with the ordered pair formed with the element x and the name of the rule f. For instance, the derivation of 4 above would then be

$$\dfrac{\dfrac{\dfrac{}{\langle 0, f_1\rangle}}{\langle 2, f_2\rangle}}{\langle 4, f_2\rangle}$$

more often written

$$\dfrac{\dfrac{\dfrac{}{0}\,f_1}{2}\,f_2}{4}\,f_2$$

Such a derivation is easier to check, as checking the node

$$\dfrac{2}{4}$$

requires to find the rule f such that $f(2) = 4$, while checking the node

$$\dfrac{2}{4}\,f_2$$

just requires to apply the rule f_2 to 2 and check that the result is 4.

But these rules names are redundant, as soon as the relation $\cup_i f_i$ is decidable. So, in general, they can be omitted.

1.4 Derivations and derivations

Instead of omitting the rule names, it is possible to omit the elements of A. The derivation of 4 is then

$$\dfrac{\dfrac{\dfrac{}{f_1}}{f_2}}{f_2}$$

that can also be written

$$\dfrac{\dfrac{\dfrac{}{\cdot}\,f_1}{\cdot}\,f_2}{\cdot}\,f_2$$

Although it is not explicit in the derivation, the element 4 can be inferred from this derivation with a top-down *conclusion inference algorithm*, because the rules f_i are functions. The conclusion of the rule f_1 can only be $f_1(\langle\rangle) = 0$, that of the first rule f_2 can only be $f_2(\langle 0\rangle) = 2$, and that of the second can only be $f_2(\langle 2\rangle) = 4$.

We can introduce this way two kinds of derivations *labeled with objects* and *labeled with rules names.*

1.5 Making the rules functional

Natural deduction proofs, for instance, are often labeled both with sequents and rule names

$$\dfrac{\dfrac{}{P,Q,R \vdash P}\,\text{axiom} \qquad\qquad \dfrac{}{P,Q,R \vdash Q}\,\text{axiom}}{P,Q,R \vdash P \wedge Q}\,\wedge\text{-intro}$$

but they can be labeled with sequents only

$$\dfrac{\dfrac{}{P,Q,R \vdash P} \qquad\qquad \dfrac{}{P,Q,R \vdash Q}}{P,Q,R \vdash P \wedge Q}$$

and proof-checking is still decidable. They can also be labeled with rule names only, but we have to make sure that all the deduction rules are functional, which is often not the case in the usual presentations of Natural deduction. The rule

$$\dfrac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \wedge B}\,\wedge\text{-intro}$$

is functional: there is only one possible conclusion for each sequence of premises, but the axiom rule

$$\dfrac{}{\Gamma, A \vdash A}\,\text{axiom}$$

is not. To make it functional, we must introduce a different rule $\text{axiom}_{\langle\Gamma,A\rangle}$ for each pair $\langle\Gamma, A\rangle$. Thus, the proof above must be written

$$\dfrac{\dfrac{}{\cdot}\,\text{axiom}_{\langle\{Q,R\},P\rangle} \qquad\qquad \dfrac{}{\cdot}\,\text{axiom}_{\langle\{P,R\},Q\rangle}}{\cdot}\,\wedge\text{-intro}$$

and its conclusion $P, Q, R \vdash P \wedge Q$ can be inferred top-down.

2 Derivations in elementary computability theory

2.1 A pedagogical problem

The set of computable functions is often defined inductively as the smallest set containing the projections, the null functions, and the successor function, and closed by composition, definition by induction, and minimization.

But to study the computability of properties of computable functions, we need a notion of *program*, that is we need a way to express each computable function by a expression, to which a Gödel number can be assigned. A usual solution is to introduce Turing machines at this point.

This solution however is not pedagogically satisfying as, while the students are still struggling to understand the inductive definition, we introduce another, that is based on completely different ideas, letting them think that logic made of odds and ends. Moreover, the equivalence of the two definitions requires a tedious proof. Why do we not base our notion of program on the inductive definition itself?

2.2 Programs already exist

The function $x \mapsto x + 2$ is computable because it is the composition of the successor function with itself

$$\dfrac{\overline{x \mapsto x+1} \qquad\qquad \overline{x \mapsto x+1}}{x \mapsto x+2}$$

But such a derivation labeled with objects cannot be used as a program, because to label its nodes, we would need a a language to express all the functions, and there is no such language.

But if we use a derivation labeled with rule names instead

$$\dfrac{\dfrac{}{.}\,Succ \qquad \dfrac{}{.}\,Succ}{.}\,\circ^1_1$$

and write the trees in linear form: $\circ^1_1(Succ, Succ)$, we obtain a simple functional programming language to express the programs.

For instance, introduce a Gödel numbering $\ulcorner . \urcorner$ for these programs, and assume there is an always defined function h such that

- $h(p, q) = 1$ if $p = \ulcorner f \urcorner$ and f defined at q
- $h(p, q) = 0$ otherwise

then, the function

$$k = \circ^1_1(\mu^1(\pi^2_1), \circ^1_2(h, \pi^1_1, \pi^1_1))$$

is defined at $\ulcorner k \urcorner$ if and only if it is not.

We get this way a proof of the undecidability of the halting problem that requires nothing else than the inductive definition of the set of computable functions.

3 Derivations in elementary automata theory

When introducing the notion of automaton, we often introduce new notions, such as those of transition rules and recognizability. Having introduced the notion of derivation from the very beginning of the course permits to avoid introducing these as new notions.

Consider for instance the automaton

$$\mathit{odd} \xrightarrow{a} \mathit{even}$$

$$\mathit{even} \xrightarrow{a} \mathit{odd}$$

where the state *even* is final. In this automaton, the word *aaa* is recognized in *odd*. Indeed

$$\mathit{odd} \xrightarrow{a} \mathit{even} \xrightarrow{a} \mathit{odd} \xrightarrow{a} \mathit{even}$$

If, instead of introducing a new notion of transition rule, we just define transition rules as deduction rules

$$\frac{\mathit{even}}{\mathit{odd}}\,a \qquad\qquad \frac{\mathit{odd}}{\mathit{even}}\,a \qquad\qquad \frac{}{\mathit{even}}\,\varepsilon$$

then, the element *odd* has a derivation

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\mathit{even}}\,\varepsilon}{\mathit{odd}}\,a}{\mathit{even}}\,a}{\mathit{odd}}\,a$$

If we label this derivation with rule names we obtain

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\cdot}\,\varepsilon}{\cdot}\,a}{\cdot}\,a}{\cdot}\,a$$

which can be written in linear form $a(a(a(\varepsilon)))$, or aaa.

Thus, a word w is recognized in a state s if and only if it is a derivation, labeled with rule names, of s.

This example introduces some points to be discussed: the rules

$$\frac{even}{odd}\,a \qquad \frac{odd}{even}\,a$$

are labeled with the same name. If the automaton is deterministic, we can replace these two rules with one: a function such that $a(even) = odd$ and $a(odd) = even$. But for non deterministic automata, we either need to extend the notion of rule name, allowing different rules to have the same name, or to consider two rule names

$$\frac{even}{odd}\,a_1 \qquad \frac{odd}{even}\,a_2 \qquad \frac{}{even}\,\varepsilon$$

and map the derivation $a_1(a_2(a_1(\varepsilon)))$ to the word $a(a(a(\varepsilon)))$ with the function $|.|$ defined by: $|\varepsilon| = \varepsilon$, $|a_1(t)| = a(|t|)$, and $|a_2(t)| = a(|t|)$.

4 Introducing the Brouwer-Heyting-Kolmogorov correspondence

4.1 A radical change in viewpoint?

The Brouwer-Heyting-Kolmogorov interpretation, and its counterpart, the Curry-de Buijn-Howard correspondence, are often presented as a radical change in viewpoint: proofs are not seen as trees anymore, but as algorithms.

But, of course, these algorithms must be expressed in some language—often the lambda-calculus. Thus, proofs are not really algorithms, but terms expressing algorithms, and such terms are nothing else than trees. So, it is fairer to say that, in the Brouwer-Heyting-Kolmogorov interpretation, proofs are not derivation trees, but trees of a different kind. For instance, the tree

$$\dfrac{\dfrac{\dfrac{\dfrac{}{P \wedge Q \vdash P \wedge Q}}{P \wedge Q \vdash Q} \qquad \dfrac{\dfrac{}{P \wedge Q \vdash P \wedge Q}}{P \wedge Q \vdash P}}{P \wedge Q \vdash Q \wedge P}}{\vdash (P \wedge Q) \Rightarrow (Q \wedge P)}$$

is replaced by the tree

$$\dfrac{\dfrac{\dfrac{}{x}}{fst} \qquad\qquad \dfrac{\dfrac{}{x}}{snd}}{\dfrac{\langle , \rangle}{\lambda x : P \wedge Q}}$$

often written in linear form: $\lambda x : P \wedge Q\ \langle snd(x), fst(x)\rangle$.

4.2 What about derivation trees labeled with rule names?

Instead of following this idea of expressing proofs as algorithms, let us just try to label the derivation above with rule names. Five rules are used in this proof. Three of them are functional

$$\dfrac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \wedge B}\ \wedge\text{-intro}$$

$$\dfrac{\Gamma \vdash A \wedge B}{\Gamma \vdash A}\ \wedge\text{-elim1}$$

$$\dfrac{\Gamma \vdash A \wedge B}{\Gamma \vdash B}\ \wedge\text{-elim2}$$

Let us just give them shorter names: $\langle , \rangle$, fst, and snd. The rule

$$\dfrac{\Gamma, A \vdash B}{\Gamma \vdash A \Rightarrow B}\ \Rightarrow\text{-intro}$$

is functional, as soon as we know which proposition A in the left-hand side of the antecedent is used. So, we need to supply this proposition A in the rule name, let us call this rule λA. Finally, the rule

$$\dfrac{}{\Gamma, A \vdash A}\ \text{axiom}$$

is functional, as soon as we know Γ and A. We could supply Γ and A in the rule name. However, we shall just supply the proposition A and infer the context Γ. Let us call this rule $[A]$. So, the proof above can be written

$$\dfrac{\dfrac{\dfrac{\dfrac{}{P \wedge Q \vdash P \wedge Q}\,[P \wedge Q]}{P \wedge Q \vdash Q}\,snd \qquad \dfrac{\dfrac{}{P \wedge Q \vdash P \wedge Q}\,[P \wedge Q]}{P \wedge Q \vdash P}\,fst}{P \wedge Q \vdash Q \wedge P}\,\langle , \rangle}{\vdash (P \wedge Q) \Rightarrow (Q \wedge P)}\,\lambda P \wedge Q$$

and if we keep rule names only

$$\dfrac{\dfrac{\dfrac{\dfrac{}{.}\,[P \wedge Q]}{.}\,snd \qquad \dfrac{\dfrac{}{.}\,[P \wedge Q]}{.}\,fst}{.}\,\langle , \rangle}{.}\,\lambda P \wedge Q$$

or in linear form $\lambda P \wedge Q\ \langle snd([P \wedge Q]), fst([P \wedge Q])\rangle$. This is the scheme representation [3] of this proof.

Let us show that the conclusion can be inferred, although we have not supplied the context Γ in the axiom rule. The conclusion inference goes in two steps. First we infer the context bottom-up, using the fact that the conclusion has an empty context, and that all rules preserve the context, except λA that extends it with the proposition A

$$\dfrac{\dfrac{\dfrac{\dfrac{}{P \wedge Q \vdash .}\,[P \wedge Q]}{P \wedge Q \vdash .}\,snd \qquad \dfrac{\dfrac{}{P \wedge Q \vdash .}\,[P \wedge Q]}{P \wedge Q \vdash .}\,fst}{P \wedge Q \vdash .}\,\langle , \rangle}{\vdash .}\,\lambda P \wedge Q$$

Then, the right-hand part of the sequent can be inferred with a usual top-down inference algorithm, using the fact that the rules are functional

$$\dfrac{\dfrac{\dfrac{\dfrac{}{P \wedge Q \vdash P \wedge Q}\,[P \wedge Q]}{P \wedge Q \vdash Q}\,snd \qquad \dfrac{\dfrac{}{P \wedge Q \vdash P \wedge Q}\,[P \wedge Q]}{P \wedge Q \vdash P}\,fst}{P \wedge Q \vdash Q \wedge P}\,\langle , \rangle}{\vdash (P \wedge Q) \Rightarrow (Q \wedge P)}\,\lambda P \wedge Q$$

4.3 Brouwer-Heyting-Kolmogorov interpretation: an optional modification

In the rule

$$\dfrac{\Gamma, A \vdash B}{\Gamma \vdash A \Rightarrow B}\ \Rightarrow\text{-intro}$$

instead of supplying just the proposition A, we can supply the proposition A and a name x for it. Then, in the axiom rule

$$\dfrac{}{\Gamma, A \vdash A}\ \text{axiom}$$

instead of supplying the proposition A, we can just supply the name that has been introduced lower in the tree for it. We obtain this way the tree

$$\cfrac{\cfrac{\cfrac{\cfrac{}{\cdot}\;x}{\cdot}\;snd \qquad\qquad \cfrac{\cfrac{}{\cdot}\;x}{\cdot}\;fst}{\cdot}\;\langle,\rangle}{\cdot}\;\lambda x : P \wedge Q$$

in linear form $\lambda x : P \wedge Q\ \langle snd(x), fst(x)\rangle$, which is exactly the representation of the proof according to the Brouwer-Heyting-Kolmogorov interpretation.

So, the Brouwer-Heyting-Kolmogorov interpretation boils down to use of derivations labeled with rule names plus two minor modifications: context inference and the use of variables. These two modifications can be explained by the fact that Natural deduction does not really deal with sequents and contexts: rather with propositions, but, following an idea initiated in [4], some rules such as the introduction rule of the implication dynamically add new rules, named with variables.

References

[1] P. Aczel, An introduction to inductive definitions, *Handbook of Mathematical Logic*, Studies in Logic and the Foundations of Mathematics 90, 1977, pp. 739-201.

[2] G. Dowek, *Proofs and Algorithms: An Introduction to Logic and Computability*, Springer-Verlag, 2011.

[3] G. Dowek and Y. Jiang, On the expressive power of schemes, *Information and Computation*, 209, 2011, pp. 1231-1245.

[4] P. Schroeder-Heister, A natural extension of natural deduction, *The Journal of Symbolic Logic*, 49, 4, 1984, pp. 1284-1300.

Received \jreceived