Pathways of deduction

A. Carbone
Département d'Informatique, Université Pierre et Marie Curie
INSERM U511, 91 Bd de l'Hôpital, 75013, Paris

Abstract

Cyclic structures underlie formal mathematical reasoning, and replication and folding play a crucial role in the complexity of proofs. These two aspects of the geometry of proofs are discussed.

1 Deductions, foldings and the brain

Different models of various regions of the brain have been proposed and they stimulated the discussion on the way our mind works. The essential feature of most of these models is the *hierarchical* structure which is underlying the organization. What we "see" is nevertheless not necessarily the basic mechanism. Recent studies in computational complexity and proof theory reveal that hierarchical organizations, even though structurally appealing, are computationally inefficient. In fact, our brain seems to be "fast" in performing certain tasks (such as perceiving the presence of an animal in the landscape, or intuitively grasping a complicated mathematical idea) and extremely "slow" in performing others (as the construction of a mathematical proof). No hierarchical structure conceived by man displays similar features.

In this paper we would like to show how the usual hierarchical approach to the construction of formal mathematical proofs (introduced with the work of Frege and established through the work in logic until this last decade) is inappropriate to reveal the intricate structures underlying proofs.

There are two basic operations which one finds in every *complex* system: *replication* and *folding* (or *cancellation*). (Based on these two operations one can suggest a formal definition of complexity.) Replication and folding arise in many forms, going from a more abstract to a more concrete nature. On the abstract side, it has been and remains a challenge to study the effect of replication and folding on mathematical structures. For example, the folding in the combinatorial group theory corresponds to word cancellation and it is encoded in the 2-dimensional language of the van Kampen diagrams.

On the concrete side, these two operations underly many biological systems, where the perfection and the symmetry of a mathematical structure is broken. Yet "pseudo-structures" seem to be preserved and one looks for mathematical tools to handle them.

Replication and folding manifest themselves in vastly different situations where they are governed by different laws. In molecular biology for instance, one observes the replication in the production of thousands of copies of the same RNA strand, followed by the physical folding of those. This kind of folding constraints the dynamics of the cell and may interfere with the rate of production of RNA (via the network of protein-DNA interaction).

DNA also folds, but for different purposes. For example, the 2 meters of human DNA must fit into the cell nucleus of 10 microns. This folding is organized (in chromosomes) along very specific structural patterns serving several biological functions.

One more example involves proteins. They fold in the course of translation into a compact 3-dimensional conformation where the geometry of the specific active sites on the boundary determines their bio-chemical activity.

Summing up:

- folding allows a large object to fit into a *small* space,
- the complexity of the functions performed by the object depend on the complexity of the folding, and
- these two properties go along, since reducing the space necessarily produces complicated foldings. Possibly the evolution first folded "primordial DNA" in order to save space and then as a bonus came up with more complicated behavior.

Folding and size reduction make possible to control the object but sometimes make the manipulation of the object a slow process, in particular if a local unfolding needs to be realized (as in the transcription of RNA from DNA). Biologists are searching for rules of folding and unfolding and it remains unclear, for example, in how many ways a (natural) protein can fold. Do foldings follow specific pathways?

Amazingly, this pure biological (and superficial) discussion remains meaningful in the context of formal proofs. In proofs, and as we believe in most complex systems, replication and folding play a crucial role, and the analysis of the interaction of these operations in proofs is the main object of this paper. The biology will remain in the background, to contrast what happens in formal proofs. In proofs the folding and the replication are tide up in intricate ways (the unfolding induces replication and the folding induces identification) while biological systems tend to separate folding and replication in order to operate efficiently. As a result, the dynamics of proofs turns out to be very different (at least to a casual eye) from the dynamics of biological systems.

Dynamics in proofs underlies a process of computation. We are after the structure of this dynamics, or if one prefers, of the computation behind proofs, where we shall see that *short* proofs need to contain *cycles* and that the elimination of these cycles might enlarge the proof in such a way that no human mind could possibly embrace the details.

A proof, as understood in this article, represents a *finite* computation despite the presence of cycles which might create an illusion of infinite computations hidden behind. (There are proofs that might describe infinite processes but we shall not consider them here.) This will be clarified in Section 6 where we shall show that infinite iterations (intrinsic, for instance, to autonomous dynamical systems) are *not* present in proofs. The absence of infinite iterations becomes apparent if one introduces possibilities of parallel computation in a proof (see Section 6), otherwise one necessarily has cycles which bias our perspective towards a dynamical view.

Why do we look at proof theory and at the notion of formal deduction instead of considering the notion of "truth"? Here is a basic fact which might give a clear picture of the reason to the non-logician: if we extend our mathematical theory with an inconsistent axiom, we might end-up with a new theory for which there is a $k > 0$ such that all proofs of length (i.e. the number of formal deductive steps) smaller than k are proofs of true statements. This means that even if we work in an inconsistent setting we can still deduce interesting results. Proof theory allows us to speak about what is going on in the stretch of computational time $< k$ and permits an analysis of the structure of the computation when resources are bounded (e.g. with respect to time and space).

I would like to acknowledge the numerous conversations I have with Misha Gromov in the subjects touched in this paper. This work was partly written during my visit at the Institut des Hautes Études Scientifiques.

2 A formal language to describe proofs

In the thirties, Gentzen introduced a logical system (i.e. a finite set of *rules* for the manipulation of logical formulas) which, nowadays, is used at large in the study of formal proofs. Its success is due to the useful *combinatorial* properties concerning the formulas appearing in the proofs as well as the graph-theoretical features of the structure of the proofs. This logical system allows the manipulation of *sequences* of formulas which, for our purposes, will be simply chains of symbols that one can combine through controlled transformations of the sub-chains as described below. The alphabet out of which the chains are constructed is the logical language containing variables, constant, function and relation symbols, as well as logical connectives $\wedge$ (and), $\vee$ (or) and logical quantifiers $\forall$ (for all), $\exists$ (there exists). Some extra symbols are used as separators, that is (,) (parenthesis) and , (comma). For each relation symbol R there is a unique *complement* $R^\perp$ that represents the *logical negation* of R (often written $\neg R$). The symbols $\wedge, \forall$ are *complementary* to $\vee, \exists$, and the complement of a chain of symbols is the chain obtained by complementing all relation symbols and logical symbols in it. For instance the complement of the formula $p \vee (q^\perp \wedge r)$ is $p^\perp \wedge (q \vee r^\perp)$, and the complement of $(A^\perp(c) \wedge B(d)) \vee \exists x\ C(x)$ is $(A(c) \vee B^\perp(d)) \wedge \forall x\ C^\perp(x)$. For convenience, if A is a formula then we shall denote with the symbol $A^\perp$ its complement. (The reader might notice the analogy with the Watson-Crick complementarity in DNA for which the complement of the sequence $ACGGGGTTTCC$ is $TGCCCCAAAGG$, where the letters A, C are complementary to T, G. One might entertain herself by looking at other examples of duality and parity in mathematics and physics.)

An example of *sequence* of formulas is $A^\perp(c), \forall y\ B(y, d) \wedge C(d), B^\perp(a)$, where $A^\perp(c)$, $\forall y\ B(y, d) \wedge C(d)$ and $B^\perp(a)$ are formulas. An example of *rule* is

$$\frac{\Gamma, A \qquad \Delta, B}{\Gamma, \Delta, A \wedge B}$$

to the effect that given two sequences of formulas Γ, A and Δ, B one can construct a new sequence containing both collections of formulas Γ and Δ (which might be empty) and the formula $A \wedge B$. (The logical meaning of the rule should not worry the reader.) Similar rules are defined for the other logical connectives and quantifiers.

Together with those rules allowing the construction of new formulas lying in a sequence, there are two extra rules:

$$\frac{\Delta, A, A}{\Delta, A}\ \textit{contraction} \qquad\qquad \frac{\Gamma, A \qquad \Delta, A^{\perp}}{\Gamma, \Delta}\ \textit{cut}$$

The *contraction rule* says that if two copies of a formula A lie in the same sequence, then they can be identified and only one remains. The *cut rule* says that if two sequences contain *complementary* formulas then the two formulas cancel out. In logic, the cut rule is a generalization of the well-known rule of *modus ponens*, saying that if the lemma A has been derived, then the sequence of formulas Δ is derivable from the sequence $\Delta, A^{\perp}$. The occurrences A and $A^{\perp}$ in the two sequences considered in the cut rule, are called *cut-formulas* or *lemmas*.

The set of rules which we described, together with sequences of the form $\Gamma, A, A^{\perp}$ (called *axioms*), define the so-called *predicate logic*. If quantifiers are not allowed to occur in the formulas, the logic is called *propositional*.

A *formal proof* is a manipulation of sequences of formulas by means of the rules above, which starts from axioms, creates new sequences which can be combined with any of the previously derived sequences and any of the axioms, and which ends with a sequence called *theorem*. An example of formal proof is illustrated on the left hand side of Fig. 2.

The use of the cut rule in formal proofs allows the construction of *compact* deductive arguments. We give two intuitive examples of the use of cuts in proofs. They illustrate the power of lemmas in deductions.

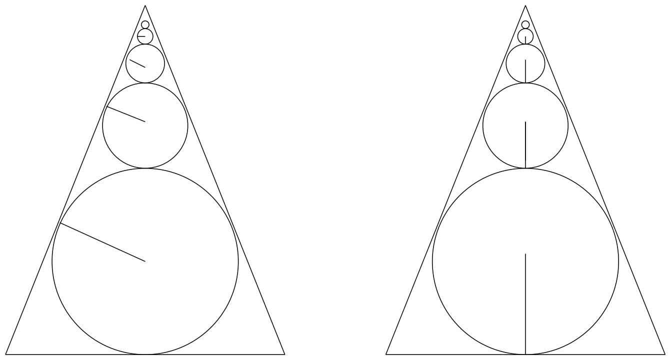

Figure 1: Triangles with inscribed circles.

Example 1 We take an "isoscele" triangle and we inscribe a circle in it as illustrated in Fig. 1. Repeatedly, we inscribe a smaller circle on the top of the previous one and so on. We shall obtain an infinite number of such circles, one on the top of the other. To compute the sum of the radiuses of such circles (see Fig. 1 on the left), we can calculate the radius of each circle and then sum up the values, but this sum is infinite. This approach produces an *explicit* computation in the sense that each radius is explicitly considered in the reasoning and contributes to the sum. A finite but *implicit* solution, is illustrated in the picture on the right of Fig. 1, where one can see that the sum of the radiuses coincides with half of the height of the triangle. (Infinity is present in this solution but suppressed by the implicitness.) The short cut we used to pass from an argument involving an infinite computation to a finite argument corresponds to the presence of lemmas in a formal proof. (For all apparent simplicity, no known automatic deduction system is able to come up with this short cut.)

Example 2 To avoid the explicit calculation of $20+19+18+\ldots+3+2+1$, at the age of seven, Gauss observed that if we add by columns the following additions

$$\begin{array}{ccccccccccc} 20 & + & 19 & + & 18 & + & \ldots & + & 2 & + & 1 \\ 1 & + & 2 & + & 3 & + & \ldots & + & 19 & + & 20 \end{array}$$

the result will be 10 times 21. By using the lemma

$$n+m=p \rightarrow (n-1)+(m+1)=p$$

one can even avoid the addition by columns: one computes $20+1$, and applies 20 times the lemma. The lemma allows to codify the explicit calculation and to deduce the solution faster.

The reader might have a sense now of the shortening induced by certain mathematical arguments contrapposed to the length of the constructions that one would generate if lemmas were not allowed to be used. These are toy examples, but in the next section we discuss some concrete mathematical ones.

3 Unfolding

A precise statement concerning *explicit* and *implicit* constructions, that is constructions allowing or not the use of lemmas, was proved by Gentzen

Theorem 3 (The Cut Elimination Theorem - Gentzen) *If* Π *is a formal proof (with cuts) of a statement* S*, then there is an effective way to transform* Π *into a formal proof* Π' *of* S *which does not make use of the cut rule. This holds for both propositional and predicate logic.*

The transformation is possible by means of *local* manipulations which *unfold* the proof Π into the proof Π', usually much larger than Π. The price to pay for the elimination of cuts may be exponential for propositional logic, and multi-exponential (i.e. a tower of 2's) for predicate logic. From these values, it is clear that there are situations where one might find a small proof with cuts, even though one might never be able to look at the cut-free form because too huge.

But why would we like to look at the cut-free form of a proof? Roughly speaking this form corresponds to the *combinatorial* version of the original proof. In a proof free of cuts, the construction of an "object" in the proof is done "piece by piece" and we might be interested to know how large this object is, or in how many steps we can build it, etc. (We shall comment on a concrete example of such construction in Section 5.) Since the complexity of the object and the length of the proof are related, one hopes to extract bounds from formalized proofs by analysing cut elimination. This is a direction of research proposed by Kreisel.

In mathematical practice proofs with “large” cuts are regarded as less *elementary* than those with “smaller” ones. For instance, the Prime Number Theorem was originally proven with the cutting edge of the Riemann zeta function while the elementary proof was found much later and with a great effort. On the other hand, it took a long time to furnish a short non-elementary proof of the van der Waerden Theorem on arithmetic progressions.

The beauty of the van der Waerden Theorem resides in the simplicity of its formulation

> *given a finite set of points in a finitely colored (partitioned) plane, there is a parallel translation followed by a dilation which moves the set into a monochromatic position.*

Amazingly, there is still no logically simple proof of the result. In 1987, Girard showed that one can formally recover (by cut elimination) the elementary combinatorial argument used by van der Waerden from the dynamical system proof of Furstenberg and Weiss (where the key transcendental ingredient is the fact that the intersection of a decreasing sequence of non-empty compact sets is non-empty). This transformation consists of purely local combinatorial manipulations of formulas (together with some finitarisation of compactness). Thus the proof based on dynamical systems contains, in some codified form, all information needed for the combinatorial proof.

Girard’s theorem is a single result of this kind, and in most significant cases in number theory and algebraic geometry there is no elementary counterpart to analytical proofs. In no single case there is a formal derivation of one kind of proofs from the other.

4 A geometrical view of the unfolding

This section and the following one represent a condensed survey of the subject and the reader should not be surprised if they turn out to be hard to follow in detail.

A formal proof is usually visualized as a “finite tree” of rules (growing downwards), whose root is the theorem, the leaves are axioms, and the internal nodes are intermediate sequences of formulas derived from one or two sequences (which label the antecedents of the node in the tree) through the formal rules. An example is illustrated in Fig. 2.

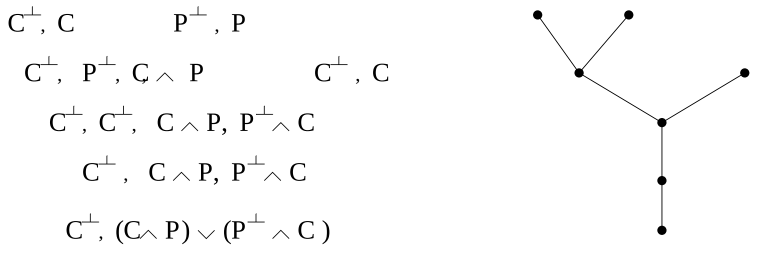

Figure 2: A formal proof and its associated tree of derivation. Each node of the tree corresponds to a sequence of formulas in the proof. There are three axioms and three leaves of the tree, three intermediate sequences and three internal nodes of the tree. The theorem corresponds to the root.

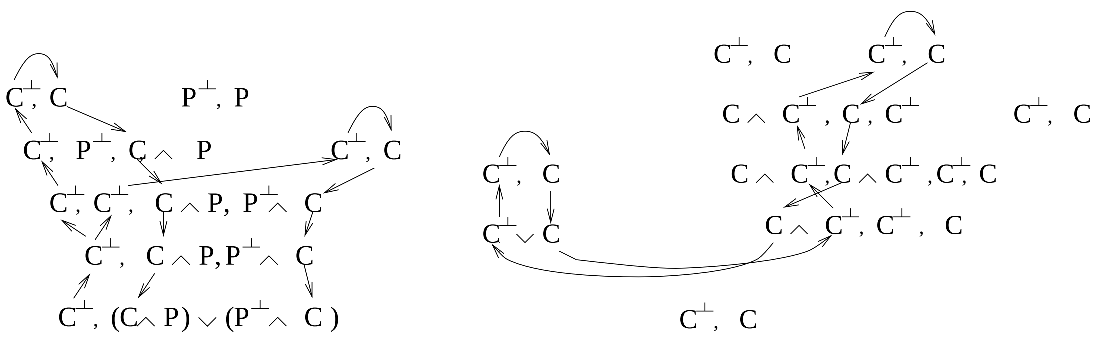

Figure 3: Logical paths between formula occurrences in formal proofs. The proof on the left displays a splitting point of the graph correspondent to the contraction of the formula C; the figure on the right displays a proof whose logical flow graph contains an oriented cycle. (In both pictures, only parts of the logical flow graphs are traced.)

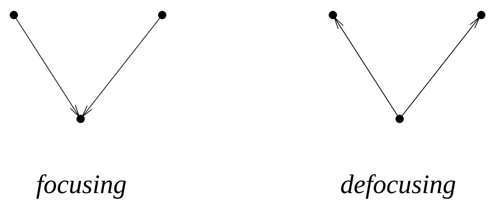

Figure 4: Branching points. When lying in a logical flow graph, they correspond to the use of contractions.

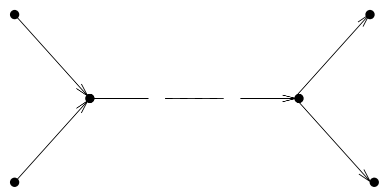

Figure 5: A defocusing point follows a focusing one. Between the two branching points there is a path linking them. This configuration corresponds to the presence in the proof of two contractions, and of a cut in between them.

There is another kind of graph, not necessarily a tree anymore, that can be associated to a formal proof. It represents the "flow of formulas" in the proof, and it is called the *logical flow graph.* This graph carries more information than the tree derivation (for instance, it distinguishes proofs with cuts from those without cuts) and the process of cut elimination can be adequately expressed in terms of combinatorial operations over this graph.

Two examples of logical flow graphs are illustrated in Fig. 3. They show the occurrences of the formula C logically linked within the proof. The left hand side diagram in Fig. 3, displays a proof with links for P as well as for C. By looking at the pictures one observes that contractions correspond to branching points in the graphs. We distinguish the branching points where a directed edge forks in two edges by calling them *defocusing* points, while vertices where two directed edges come together and are followed by a single edge are called *focusing* points (see Fig. 4).

A logical flow graph might be arbitrarily complicated. In fact, every (non-oriented) graph with degree of the vertices ≤ 3 can be topologically embedded into the logical flow graph of some proof.

The more complicated the graph is, the more logical information the

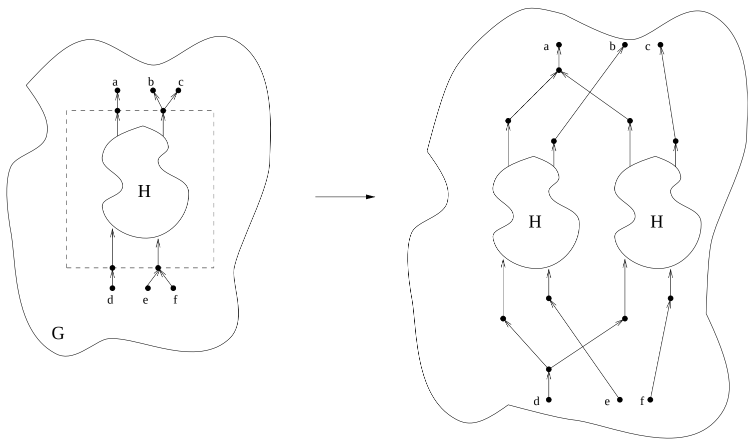

Figure 6: A subgraph H of the logical flow graph G is duplicated. The branching points below the nodes b, c and above the nodes e, f disappear with the duplication. Some new branching point is introduced: one below the node a and the other above the node d.

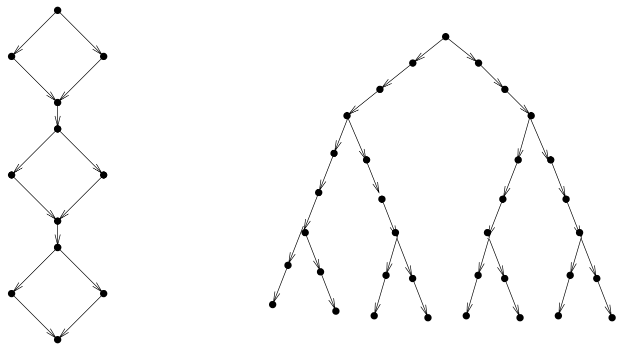

Figure 7: A graph where several directed paths share common parts (left) and its unfolding (right). The number of paths is exponential in the number of diamonds in the graph. Observe that the folding plays the role of the universal covering in the category of oriented graphs.

graph carries. A good combinatorial way to understand this point is to count the number of different paths encoded in a logical flow graph. Let us consider for instance the graph in Fig. 5. It is easy to count 4 different oriented paths coded in it. Each one of the paths shares with the others, some part which might be constituted by a very long chain of nodes. It is the sharing of nodes that reduces the complexity of the representation. In Fig. 7 we can see how an exponential number of paths can be coded in a small graph.

During the process of elimination of cuts, the logical flow graph of a proof undergoes significant *topological* changes and with this, its size blows up. The key idea is to transform a graph which contains paths starting from a focusing branching point and arriving to a defocusing one (see Fig 5) into a graph free of such configurations. The branching points where the graph splits or recombines are *locally* disrupted and this induces *global* effects in the topology of the graph (for instance cycles might be disrupted as illustrated in the second diagram on the right of Fig. 9).

The combinatorial idea which underlies the cut elimination procedure is the following. Given the logical graph of the proof, the procedure chooses a subgraph of it and resolves some of the focusing or defocusing points by *duplicating* the subgraph, as illustrated in Fig. 6. By doing this, some of the branching points are eliminated but some *new* ones might be introduced. All in all, one moves around branching points and pushes them towards the boundary of the graph, until they are absorbed by the boundary and eliminated. Gentzen Cut Elimination Theorem ensures that this can be always done with a finite number of duplications.

Among the effects that the operation of duplication might generate, one can see Fig. 8 and Fig. 9.

5 Cyclic structures

Proofs without cuts are *acyclic* as well as proofs with cuts and no contractions. This means that cycles are built through the *interaction* between cuts and contractions.

As a side effect of the rules for the manipulation of sequences of formulas, cycles in a proof organize following an interesting geometric pattern. In fact, any cyclic structure (i.e. any group of cycles nested together, as illustrated

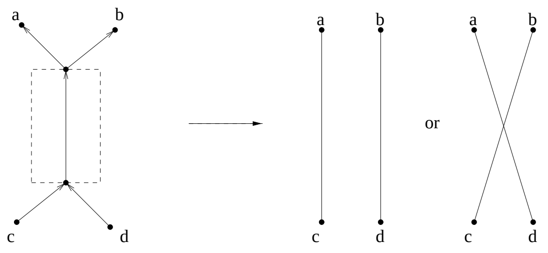

Figure 8: Duplication of a path. This kind of duplication might give two different connections between the nodes a, b and c, d. (This is reminiscent to how the enzyme *recombinase* alters DNA strands by switching one of the connections above (on the right) into the other.)

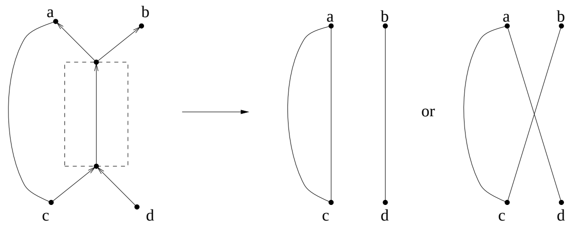

Figure 9: The duplication of a path might induce the splitting of a cycle.

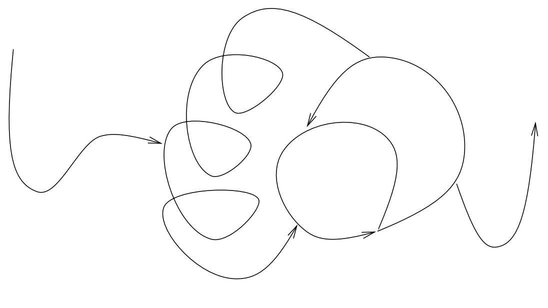

Figure 10: A cyclic structure lying in the logical flow graph of a proof, together with a path going in and a path going out the cyclic structure.

in Fig. 10) is linked to the rest of the graph in such a way that

- there is a point in the graph, which is *external* to the cyclic structure, and from which it starts a path that arrives to *any* point lying in the cyclic structure, and
- there is another point in the graph, which is again *external* to the cyclic structure, and to which it arrives a path that starts from *any* point lying in the cyclic structure.

This means that for each cyclic structure, there is (at least) a path going in the structure and a path going out from it (as in Fig. 10). In other words, cyclic structures in proofs behave essentially as "open systems".

The existence of ways in and of ways out in a cyclic structure can be proved by observing that if no way in or no way out was present in a cyclic structure, then the operation of duplication illustrated in Fig. 6 could not split (some) cycles and therefore reduce the logical flow graph of the proof to a cycle-free graph. Since we know that cycles can always be eliminated by applying the procedure of cut elimination, the above hypothesis leads to a contradiction.

The elimination of cycles through duplication is done at great expense of the proof size. Hence, it is useful to ask, whether cycles are relevant for a proof to be short.

When one considers a logic *with quantifiers* the answer is positive. This can be shown by proving in a *few* steps of deduction, that large integers exist. It turns out that not only one can prove in a very few steps that n, 2^n, 2^{2^n} exist, but that also $2^{2^{2^n}}$ and $e(2,n)$ (i.e. a tower of n 2's) can be constructed with essentially n steps of formal deduction. To construct $2^{2^{2^n}}$ and $e(2,n)$ with such a small complexity of derivation, one *needs* cycles, and it can be shown that any cycle-free proof of size n can construct only objects of size much smaller than $2^{2^{2^n}}$. These short proofs *codify* in a small space the complete description of the object of large size. This codification exploits the high *symmetry* of the object and this latter is reflected in the argument of the proof by a logical description of a repeated substitution of logical "terms" (in essence, given a chain of symbols, some of these symbols are substituted with the chain itself, and this operation is allowed to repeat). In the logical flow graph, this repeated substitution corresponds to the presence of cycles.

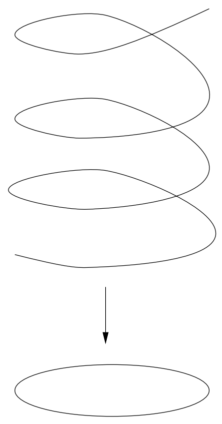

Figure 11: The projection of the spiral back down to the circle. Observe that the spiral is the universal covering of the circle and that the map represented in the figure is the canonical projection.

Are cycles necessary to short cuts in *propositional* logic? Surprisingly the answer is negative. In fact, there is a fine procedure for the manipulation of lemmas in proofs that allows to transform locally the structure of the proof until cycles are eliminated (but cut-formulas will still be present). The resulting proof is of *polynomial* size in the size of the original proof, and the degree of the polynomial is the number of cycles of the original proof.

We do not know whether all true propositional formulas have polynomial size proof in the logical system described in Section 2. If this were the case then NP would be equal to co-NP. This problem is a brother to the famous $P = NP$ question. We believe that an understanding of the folding and unfolding of propositional proofs might turn out to be crucial for complexity issues and possibly shade some light on $P =?\, NP$.

6 Cycles and spirals

Cycles cannot be eliminated silently when quantifiers are involved in the deduction (as seen in Section 5) since an explosive blow up in the size of the proof might occur. Also, cycles suggest a codification of an *infinite* process while we know, by the Theorem of Cut Elimination, that this codification represents only a *finite* number of iterations. Therefore the “intuition” of infinity is an illusion, and one wonders why cycles appear at all in our formal representation of proofs.

There is a set of rules for the manipulation of chains of formulas that creates different geometric structures in proofs, where cycles do not appear but *spirals* replace them instead. A comparison of the two deductive systems shows that cycles are *projections* of spirals, as illustrated in Fig. 11.

Spirals properly represents the dynamics within a proof but the *complexity* of this representation is large: if the size of a proof with cycles equals n, then the size of the corresponding proof where spirals replace cycles is roughly $2^{2^{2^n}}$. Thus the proof with spirals is too big to be handled, and one is obliged to work in the "projected space", where spirals turn into cycles.

One might speculate that computational complexity forces the brain to follow cyclic pathways of deduction, being aware, at the same time, that the "real" dimension of the space of reasoning is bigger than the "descriptive" dimension. The interweaving of these two modes of human reasoning underly the difficulty of our attempts to understand how our mind works. Taking this view, one starts doubting how often what we call "natural" corresponds to "reality".

7 Conclusions

We have seen how proof theory provides a procedure for unfolding a proof with cuts, e.g. turning the trascendental proof of the van der Waerden theorem into the elementary one. Yet there is no procedure for compactifying a proof by folding.

Not every proof can be in principle folded. For this, a proof needs to contain many repetitive patterns, some kind of hidden symmetry. There are at least two situations where such folding is desirable: the first concerns real mathematical proofs such as the above van der Waerden theorem where one has an intuitive feeling for the symmetries but yet the actual process of folding is by no means automatic. The second concerns automatic proofs generated by computers where the symmetry, even if present, might be hard to detect. Moreover, once it is detected, the symmetry does not immediately yield the pattern of folding. Transforming an automated formal proof into one acceptable by a human mind remains an open problem.

The difficulty resides in the fact that local relations between formulas may be lost during the unfolding (this is due to duplication) and in order to fold one has to "guess" where to insert these relations. As a comparison, look at the following geometric picture: start with the universal (infinite) covering of a compact space and consider a bounded domain in this covering which projects onto the underlying space. The problem is how to reconstruct this map of the domain together with the underlying space where it goes, by using the information encoded in the (partial) local transformation of the domain which comes from the Deck transformation group of the covering. One should be aware that the combinatorics of proofs is by far more complicated than that of the coverings.

8 Bibliographical guide

An introduction to the combinatorics and complexity of cut elimination can be found in [CS97a]. An exhaustive source (but more technical) is [Gir87b]. For a survey on propositional proof systems and their relations with complexity theory see [Pud96, Kra96].

Gentzen's original paper on the cut elimination theorem appeared in [Gen34] and, more recently, in [Sza69]. The bounds on the complexity of cut elimination are due to Statman, Orevkov and Tseitin [Sta78, Ore79, Tse68]. Some references for the work of Georg Kreisel on the importance of extracting bounds from proofs using cut elimination as the main tool are [Kre77, Kre81a, Kre81b].

The original paper of van der Waerden with the combinatorial proof of his theorem on arithmetic progressions is [VdW27]. The proof by Furstenberg and Weiss appeared in [FW78]. Girard's proof of the transformation (through cut elimination) between the combinatorial proof and the proof in dynamical systems is included in [Gir87b].

Other examples of formal proofs analysed through cut elimination (and Kreisel's counterexample interpretation) can be found in [Bel90, Kol93a, Kol93b, Kol94].

The notion of a logical flow graph is introduced in [Bus91] and an analysis of its properties appears in [Car97, Car99a]. The constructions of numbers as $2^{2^{2^n}}$, $e(2, n)$ together with the cyclic structure underlying these constructions are studied in [Car98a], and the relation of these cyclic structures to group presentations is developped in [Car99b]; cyclic structures with their ways in and ways out are analysed in [Car99a, Car97a]; the evolution of the logical flow graph during cut elimination is studied in [Car97a, CS97b]; cycles and their spiral representations are the object of study in [Car98b]. The cost of the elimination of cycles for predicate logic is computed in [Car98a], and for propositional logic in [Car97b]. The notion of symmetry and quasi-symmetry in proofs and mathematical structures in general is the main theme of [CS97b, CS96a].

References

[Bel90] G. Bellin. Ramsey interpreted: a parametric version of Ramsey's Theorem. *Contemporary Mathematics*, 106:17–37. AMS Publications, 1990.

[Bus91] S. Buss. The undecidability of k-provability. *Annals of Pure and Applied Logic*, 53:72–102, 1991.

[Car97] A. Carbone. Interpolants, cut elimination and flow graphs for the propositional calculus. *Annals of Pure and Applied Logic*, 83:249–299, 1997.

[Car97a] A. Carbone. Duplication of directed graphs and exponential blow up of proofs. *Annals of Pure and Applied Logic*, 100:1-76, 1999.

[Car97b] A. Carbone. The cost of a cycle is a square. *The Journal of Symbolic Logic*, 2000. To appear.

[Car98a] A. Carbone. Cycling in proofs and feasibility. *Transactions of the American Mathematical Society*, 5:2049–2075, 2000.

[Car98b] A. Carbone. Turning cycles into spirals. *Annals of Pure and Applied Logic*, 96:57–73, 1999.

[Car99a] A. Carbone. Streams and strings of formal proofs. *Theoretical Computer Science*, 2000. To appear.

[Car99b] A. Carbone. Asymptotic cyclic expansion and bridge groups of formal proofs. *Journal of Algebra*, 2000. To appear.

[CS96a] A. Carbone and S. Semmes. Looking from the inside and the outside. *Synthèse*, 2000. To appear.

[CS97a] A. Carbone and S. Semmes. Making proofs without modus ponens: An introduction to the combinatorics and complexity of cut elimination. *Bulletin of the American Mathematical Society*, 34:131–159, 1997.

[CS97b] A. Carbone and S. Semmes. *A Graphic Apology for Symmetry and Implicitness*. Mathematical Monographs, Oxford University Press, 2000.

[FW78] H. Furstenberg and B. Weiss. Topological dynamics and combinatorial number theory. *Journal d'Analyse Mathématique*, 34:61–85, 1978.

[Gen34] G. Gentzen. Untersuchungen über das logische Schließen I–II. *Math. Z.*, 39:176–210, 405–431, 1934.

[Gir87b] J-Y. Girard. *Proof Theory and Logical Complexity.* volume 1 of *Studies in Proof Theory, Monographs* – Bibliopolis, Napoli, Italy, 1987.

[Kol93a] U. Kohlenbach. Effective moduli from ineffective uniqueness proofs. An unwinding of de La Vallee Poussin's proof for Chebycheff approximation. *Annals of Pure and Applied Logic*, 64:27–94, 1993.

[Kol93b] U. Kohlenbach. New effective moduli of uniqueness and uniform a priori estimates for constants of strong unicity by logical analysis of known proofs in best approximation theory. *Numer. Funct. Anal. and Optimiz.*, 14(5&6):581–606, 1993.

[Kol94] U. Kohlenbach. Analyzing proofs in analysis. *Proceedings of the* Logic Colloquium 93 – *Keele*, 1994.

[Kra96] J. Krajíček. *Bounded Arithmetic, Propositional Logic and Complexity Theory.* Cambridge University Press, 1996.

[Kre77] G. Kreisel. From foundations to science: justifying and unwinding proofs. *Symposium: Set theory. Foundations of mathematics*, dans Recueil des travaux de l'Institut Mathématique, Nouvelle serie (Belgrade) 2(10):63–72, 1977.

[Kre81b] G. Kreisel. Extraction of bounds: interpreting some tricks on the trade. In P. Suppes, editor, *University-level computer assisted instruction at Stanford: 1968–1980*, number tome 2 (10) in Institute for Mathematical Studies in the Social Sciences, 1981.

[Kre81a] G. Kreisel. Neglected possibilities of processing assertions and proofs mechanically: choice of problems and data. In P. Suppes, editor, *University-level computer assisted instruction at Stanford: 1968–1980*, number tome 2 (10) in Institute for Mathematical Studies in the Social Sciences, 1981.

[Ore79] V.P. Orevkov. Lower bounds for increasing complexity of derivations after cut elimination. *Journal of Soviet Mathematics*, 20(4), 1982.

[Pud96] P. Pudlák. The lengths of proofs. In S. Buss, editor, *Handbook of Proof Theory*. North Holland, 1996.

[Sta78] R. Statman. Bounds for proof-search and speed-up in predicate calculus. *Ann. Math. Logic*, 15:225–287, 1978.

[Sza69] M.E. Szabo (Editor). *The Collected Papers of Gerhard Gentzen.* North Holland, Amsterdam, 1969.

[Tse68] G.S. Tseitin. Complexity of a derivation in the propositional calculus. *Zap. Nauchn. Sem. Leningrad Otd. Mat. Inst. Akad. Nauk SSSR*, 8:234–259, 1968.

[VdW27] B. L. van der Waerden. Beweis einer baudetschen vermutung. *Nieuw Archiv Wiskunde*, 212–216, 1927.