Lemma Extraction Criteria Based on Properties of Theorem Statements*

Karol Pąk

Institute of Informatics,
University of Białystok, Białystok, Poland
pakkarol@uwb.edu.pl

Abstract. Both easily readable and obscure proof scripts can be found in the Mizar Mathematical Library (MML). Many authors do not want to invest additional efforts in improving readability of their deductions, once the computer accepts their scripts. In their opinion, text of the proofs are ignored by most of the readers and the readability is essential only at the top level of scripts where statements of theorems are formulated. However, the analysis of such scripts is unavoidable if we have to rebuild some theorems to make them stronger or more easily applicable. Therefore, it is important to develop tools that can improve legibility of proofs, in particular those that shorten reasoning by removing technical sub-deductions, and this requires development of criteria that lead to extraction of statements which, in the opinion of human readers, describe well the extracted reasoning.
We propose characteristics of formula complexity that can be applied to determine which sub-deductions should be extracted so that resulting lemmas are more comprehensible. To better understand their significance we study the distribution of these characteristics on statements of theorems that are collected in the MML.

Key words: Lemma extraction, Complexity of formulas, Legibility of proofs

1 Introduction

1.1 Motivations

The legibility of proof scripts might be considered as one of the most important factors of formalization quality, but in practice the growth of proof databases is not always accompanied by the improvement of the formalization quality of the articles. Analyzing proof scripts developed with proof assistants, especially the longer and more complex deductions, leads to a conclusion that their legibility often seems to be of the secondary importance to their authors since computer assisted proof development frameworks can check the correctness of such deductions. According to the opinion of some proof authors any attempt to analyze

* The paper has been financed by the resources of the Polish National Science Center granted by decision n°DEC-2012/07/N/ST6/02147.

details of the proofs scripts created in this way is extremely difficult or even impossible. However, analysis of such proofs is unavoidable if we try to adapt or modify them [5].[1]

This concerns especially systems such as Mizar [2, 10], where the proof script language is close to the natural language, which makes it possible to create legible deductions. Therefore, authors of Mizar proof scripts can manually try to improve the comprehensibility of their work spending a lot of time over their readability [13, 6], similar as it is done for informal mathematical proofs. However, many authors do not want to invest additional efforts in this process and assume that the task can be handled automatically for them, since having a digital form of structured formal proof, a computer can not only automatically verify the correctness, but also automatically enhance proof scripts. Therefore, it comes as no surprise that Mizar is being developed in many directions to meet these needs of users, similar to the way people work on informal mathematical proofs.

We can identify three main directions of development to improve legibility. The first one is based on improving the representation of proof scripts in HTML format [14] by adding selected information that is automatically generated by Mizar and also by bringing the formal mathematical language to the informal one by introduction to the formal language idioms that stem from informal mathematical practice [7–9]. The second direction is based on the simplification of deductions in proof scripts by finding and removing irrelevant parts of reasoning or by elimination of redundant premises from the justification of steps, preserving the correctness of the modified proof scripts. The third direction is based on rebuilding the deduction structure in proof scripts by changing the order of independent steps in reasoning and also by detecting reasoning passages (called *packets*) that are, e.g., technical and repeated many times in a reasoning, and extracting them as lemmas or encapsulating them on a deeper level of a proof in the form of a nested lemma.

The last direction is still the least developed. SMT solvers can be used to choose a better order of independent proof steps [12]. A method of extracting packets as external or nested lemmas so that the correctness of proof scripts is preserved has also been developed [11]. However, an additional challenge remains to formulate packet extraction criteria so that resulting new lemmas can be accepted as ones that deserve readers' attention and are worth extracting. In this paper we concentrate on this aspect.

1.2 Proposed approach

The ability to find such passages automatically is crucial, since the extraction of carelessly selected packets can drastically reduce the proof scripts legibility even if the modification reduces the length of the proof. Additionally, the statements associated with the reasoning in a packet has usually a more complicated form

[1] Actually this is mentioned in Page 3 of an unpublished preliminary version of the article [4].

than a plain implication *premises* $\implies$ *conclusions* preceded by a sequence of universal quantifiers. It turns out that a mathematical statement can be perceivably more complex than another one, even if both have the same number of assumptions and theses; or their prenex normal forms are at the same level of the same arithmetic hierarchy [1, 11].

In this paper we analyze the selected characteristic of sentence complexity of theorems and lemmas in the Mizar Mathematical Library (MML) to get the most typical values. and then apply them in the context of lemma extraction based upon the notion of proof graph. In Section 2 we introduce the notion of an abstract model of proofs and packets. In Section 3 we discuss selected indicators of the packet's statement complexity that describe a structure generated by premises and conclusion in a packet. We discuss also the impact of the positions of quantifiers in a formula for the complexity of a deduction that justifies the formula. Then in Section 4 we report the statistical results obtained on statements of theorems and lemmas occurring in the MML. Finally, Section 5 concludes the paper and discusses future work.

2 Packets in Abstract Proof Graph

To formulate the notion of a packet, we have to fix the terminology and notation.

Let $G = \langle V, E\rangle$ be a DAG, E_1 be a subset of E, V_1 be as subset of V. An arc is called E_1*–arc* if it belongs to E_1. A path $P = \langle u_1, u_2, \ldots, u_n\rangle$ of G is called an E_1*–path* if $\langle u_i, u_{i+1}\rangle$ is an E_1–arc for $i = 1, 2, \ldots, n-1$. Additionally, we say that P *passes* V_1 if $u_2, u_3, \ldots, u_{n-1}$ belong to V_1. For vertices u, v in V, the notation $u \underset{E_1}{\to} v$ means that $\langle u, v\rangle$ is an E_1–arc. Moreover, the notation $u \underset{E_1}{\to}{}^* v$ means that there exists an E_1–path that leads from u to v. Additionally, we say that vertices u, v are connected only by passing V_1 and denoted this by $u \underset{V_1}{\rightsquigarrow}{}^* v$, if there exists a path that leads from u to v and every path that leads from u to v passes V_1.

An abstract model of proofs was considered in detail in an earlier work [11]. For our purposes we recall only the part of its definition that is the most relevant for our purposes. We illustrate it with an example on Fig. 1, where `P`, `Q`, `R`, `S`, `T` represent predicate symbols with two arguments and `F` represents a functor with one argument. Note also that the additional packet $\mathcal{P}$ is analyzed in the further part of this article. Generally, we call a DAG $\mathfrak{P} = \langle \mathbb{V}, \mathbb{O} \cup \mathbb{M}\rangle$ an *abstract proof graph* if $\mathbb{O}$ and $\mathbb{M}$ are disjointed families of arcs, called *ordered arcs* and *meta–edges* respectively, $\langle \mathbb{V}, \mathbb{M}\rangle$ is a forest, and $\mathbb{O}$ contains a distinguished set of arcs $\mathfrak{R}(\mathfrak{P}) \subseteq \mathbb{O}$, the elements of which are called *references*. Additionally, the contents of a nested reasoning are closed outside, i.e., introduced variables and formulated statements in a sub-reasoning cannot be used outside the sub-reasoning, i.e., from areas of the proof in which the sub-reasoning is nested (for each $u, v, w \in V$ if $u \underset{\mathbb{O}}{\to} v$, $u \underset{\mathbb{M}}{\to} w$ then $v \underset{\mathbb{M}}{\to}{}^* w$ and $v \neq w$). The vertices of $\mathfrak{P}$ represent steps of the reasoning, $\mathbb{O}$–arcs represent the flow of information between different steps of the reasoning, and $\mathbb{M}$–arcs represent the dependence

```
α : let x be set;
β : A1: P[x];
γ : consider y be set such that
      A2:y=F(x) by A1;
δ : A3: Q[y] by A2;
ϵ : consider z be Subset of y such that
      A4: R[z] by A3;
ζ : consider r be Relation of y,z such that
      A5: S[r] by A4;
η : A6: T[r] by A5;
```

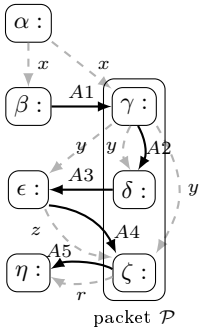


Fig. 1. The example of reasoning sequence written in the Mizar style and the corresponding fragment of the abstract proof graph that represents the reasoning.

between each step from areas of the proof and a proving fact. $\mathfrak{R}(\mathfrak{P})$–arcs that correspond to solid arrows represent the information flow between a premise (e.g., the fact labeled by A1 in β) and a place of its use (γ). The other $\mathbb{O}$-arcs that correspond to dashed arrows represent all kinds of additional constraints that force one step to precede another one, e.g., the dependence between a step that introduces a variable (the variable v in α) into the reasoning and a step that uses this variable in its expression (β, γ). Note also that arcs and vertices of abstract proof graphs are not labeled (arcs and nodes in Fig. 1 are labeled only to simplify their identification).

Using the notion of meta-edges we can define formally the notion of packet introduced in an earlier study [11]. Let us fix the notation $\mathcal{D} = \langle V_{\mathcal{D}}, E_{\mathcal{D}}\rangle$ for a subgraph of $\mathfrak{P}$ induced by a set of vertices. We call $\mathcal{D}$ a *packet*, if $\mathcal{D}$ is induced by the set of all roots in the forest $\langle \mathbb{V}, \mathbb{M}\rangle$ or every vertex of $\mathcal{D}$ has a common successor in $\langle \mathbb{V}, \mathbb{M}\rangle$. Note that in an earlier definition (see [12]) the packet was a subset of steps in a one-level deduction, where we ignored each nested local lemma that was a justification of a step in the deduction. To consider the set of a packet steps together with steps of such supplementing lemmas we define an *area* of the packet by $\mathcal{A}(\mathcal{D}) := \{v \in \mathbb{V} : \exists_{u\in V_{\mathcal{D}}} v \underset{\mathbb{M}}{\rightarrow}^{*} u\}$. A vertex v of $\mathfrak{P}$ is called a $\mathcal{D}$*–premise*, if $v \notin V_{\mathcal{D}}$ and $\langle v, u\rangle$ is a $\mathfrak{R}(\mathfrak{P})$–arc for some $u \in \mathcal{A}(\mathcal{D})$. Similarly, we call a vertex v of $\mathfrak{P}$ a $\mathcal{D}$*–conclusion*, if $v \in V_{\mathcal{D}}$ and $\langle v, u\rangle$ is a $\mathfrak{R}(\mathfrak{P})$–arc for some $u \in V_{\mathcal{D}} \setminus \mathcal{A}(\mathcal{D})$. Let v be a vertex of $\mathbb{V}$. A $\mathcal{D}$–premise p is called v*–necessary* if there exists $\mathbb{O} \cup \mathbb{M}$–path that passes $\mathcal{A}(\mathcal{D})$ and connects p with v. Note that to explore every dependency between a premise and a conclusion, we cannot be limited only to $\mathfrak{R}(\mathfrak{P})$–paths, even if reference arcs are sufficient to define premises and conclusions. As an illustration note that the step β presented in Fig. 1 is ζ–necessary, even if the reference arcs $\langle\delta, \epsilon\rangle$, $\langle\epsilon, \zeta\rangle$ do not occur in the proof graph of the packet, since there exists ordered arc $\langle\gamma, \zeta\rangle$. Note that the packet $\mathcal{P}$ has also ζ–necessary premises ϵ, and β is also δ–necessary.

In our research, we distinguished also a set of vertices $\mathcal{V}(\mathfrak{P})$ that correspond to steps that introduce variables into a reasoning in both cases of steps that introduce an universal or an existential quantifier and of steps that introduce

a new constant. A vertex v of $\mathcal{V}(\mathfrak{P})$ is called a $\mathcal{D}$–*universal*, if $v \notin V_{\mathcal{D}}$ and $\langle v, u\rangle$ is a $\mathbb{O} \setminus \mathfrak{R}(\mathfrak{P})$–arc for some $u \in \mathcal{A}(\mathcal{D})$. Similarly, we call a vertex v of $\mathcal{V}(\mathfrak{P})$ $\mathcal{D}$–*existential*, if $v \in V_{\mathcal{D}}$ and $\langle v, u\rangle$ is a $\mathbb{O} \setminus \mathfrak{R}(\mathfrak{P})$–arc for some $u \in V_{\mathcal{D}} \setminus \mathcal{A}(\mathcal{D})$. In Fig. 1, the packet $\mathcal{P}$ has two universal vertices: α, ϵ and also two existential vertices: γ, ζ. Additionally, the order of these steps in graph suggests that the packet's statement must have the following form $\forall_x \exists_y \forall_z \exists_r \ldots$.

3 Properties of the Packet's Statement

A method of extracting an arbitrary packet as an external or a nested lemma has been described in an earlier work [11]. However, the question of the packet's property which determines the choice of the extraction method is omitted. In this Section we describe the characteristic of the packet's statement complexity that describes a packet considered as a subgraph position in an abstract proof graph, i.e., the information flow between the packet and the rest of a reasoning that contains the packet.

Let us focus on the packet $\mathcal{P}$ presented in Fig. 1, the reachability relation between packet's premises and conclusions, and also the reachability relation between ones that are connected only by passing the packet's area. Using the reasoning in a packet we can provide a packet's statement in the form "*quantifiers premises* $\rightarrow$ *conclusions*" (e.g., $\ldots (P(x) \wedge R(z)) \implies (Q(y) \wedge S(r))$. However, we cannot preserve the correctness of the modified reasoning, since δ $(Q(y))$ is ϵ–necessary $(R(z))$ or more precisely, there exists an outgoing path $\langle \gamma, \epsilon, \zeta\rangle$ that generates a circle if we replace the packet by a single step (for more detail see the definition of lemma extraction procedure [11]). Since, the packet's statement has to preserve the *necessary* relation, we have at least two options for the formulation of the packet's statement:

```
... (P[x] implies Q[y]) & (R[z] implies S[r])
... P[x] implies (Q[y] & (R[z] implies S[r]))
```
(1)

Obviously, the first proposition is more general than the second one. However we can extract $\mathcal{P}$ preserving the correctness in both cases. Analyzing the structure of implications we can observe that a deduction justifying the first statement should have a form of two disjoint proofs (corresponding to each implication) and the skeleton of a deduction justifying the second one has the form "`assume P[x]; thus Q[y]; assume R[z], thus S[r]`", (see Fig. 3).

Analyzing the structure of implications we assume that the formula contains only negation, conjunction, implication; where negation can precede only a literal. Additionally, we eliminate every formula of the form $\alpha \implies (\beta \implies \gamma)$ using the *Exportation low* $(((\alpha \wedge \beta) \implies \gamma) \iff (\alpha \implies (\beta \implies \gamma)))$ and also we eliminate repetitions of formulas as $(\alpha \implies \beta) \wedge (\alpha \implies \gamma)$. We say that such formula is *implicational*. Generally, we can transform a formula to several expected forms. However, this nondeterministic does not occur in majority statements of theorem occuring in the MML. Note that the equivalence in the Mizar system is a syntactic sugar for two implications, the disjunction is used

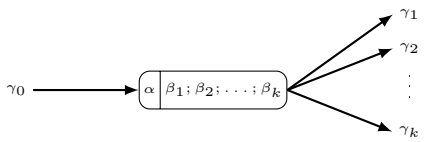
only in 2% of all theorem statements in the MML, and the negation precedes a non-literal formula only in justified cases where it facilitates the understanding of a theorem's statement in the Mizar reviewer's opinion.

Note also that when proving an implication, the most natural proof is the one where we first assume the antecedent. Obviously, we can conclude the consequent directly or using the *reductio ad absurdum* method. However, in both cases, the complexity of the antecedent has a negligible impact on the proof graph that describes the information flow in a reasoning. The only exception is when the antecedent is a conjunction of several facts, but even in that case the number of facts plays an important role than the complexity of each of them. Therefore, we omit the complexity of antecedents (accordingly, packet's premises) in our structure that describes implicational formulas (see `P[x]`, `R[z]` in Fig. 1).

Let us consider an implicational formula ϕ. Then to each sub-formula of ϕ that has the form:

$$(\gamma_0 \implies (\alpha \wedge (\beta_1 \implies \gamma_1) \wedge (\beta_2 \implies \gamma_2) \wedge (\beta_k \implies \gamma_k))$$

we can associate a vertex of a directed rooted tree as follows



The height of obtained tree is called the *depth* of ϕ and the number of leafs there the *breadth* of ϕ. The first formula presented in (1) has depth 1 and breadth 2; and the second one has depth 2 and breadth 1 (see Fig. 2). We show that

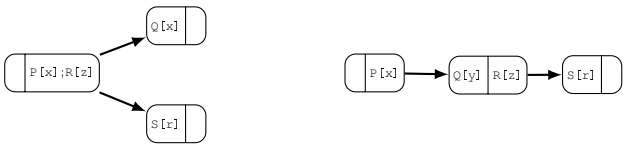


Fig. 2. The structure of implicational formulas presented in (1)

depth and breadth represent the generality level of formula that distinguishes statements of lemmas and theorems. Note that in the most general packet's statement, the formula is equivalent to the basic packet's statement (for more detail see [11]), i.e., a conjunction of implications, where the consequent of a given implication is one of the packet's conclusions c, and the antecedent is a conjunction of c–necessary the packet's assumptions. In consequence, the breadth of the packet's statement should be close to the number of packet's assumptions and the depth of the packet's statement should be close to one or even 0 if the packet has no assumptions. However, the modification of the reasoning part left after

the extraction of a packet with such statement is more complicated and a generated proof of the statement based upon the packet's reasoning is extremely long and has repeated passages that do not occur in existing proof scripts in the MML. Therefore, we analyze the depth and the breadth of the statement of a theorem that is collected in the MML in the context of a situation, where the statement is as general as possible and a generated proof does not have repeated passages. Obviously, the optimal values of depth and breadth are determined by properties of packets. Note also that we can indicate the formula that has breadth 1 (see (3)). However, the generality level of the formula is low. As an illustration, let us denote by $\mathcal{P}re(\mathcal{D})$ the set of $\mathcal{D}$–premises and similarly denote by $\mathcal{C}on(\mathcal{D})$ the set of $\mathcal{D}$–conclusions. We define two recursive families of vertices $\{\mathcal{P}re(\mathcal{D})^i\}_{i=1}^{\infty}$, $\{\mathcal{C}on(\mathcal{D})^i\}_{i=1}^{\infty}$ as follows:

$$\begin{aligned}
\mathcal{P}re(\mathcal{D})^0 &= \{p \in \mathcal{P}re(\mathcal{D}) : \neg \underset{u\in\mathcal{C}on(\mathcal{D})}{\exists} u \underset{\mathbb{M}\cup\mathbb{O}}{\rightarrow^*} p\},\\
\mathcal{C}on(\mathcal{D})^i &= \{c \in \mathcal{C}on(\mathcal{D}) : \underset{p\in\mathcal{P}re(\mathcal{D})^i}{\exists} p \underset{\mathcal{A}(\mathcal{D})}{\rightsquigarrow^*} c\},\\
\mathcal{P}re(\mathcal{D})^{i+1} &= \{p \in \mathcal{P}re(\mathcal{D}) : \underset{c\in\mathcal{C}on(\mathcal{D})^i}{\exists} c \underset{(\mathcal{A}(\mathcal{D}))'}{\rightsquigarrow^*} p\}.
\end{aligned} \tag{2}$$

Note that only a finite number of elements in the families can be non-empty, since $\mathcal{P}re(\mathcal{D}) \cup \mathcal{C}on(\mathcal{D})$ is finite. Additionally, there exists a number d such that $\mathcal{C}on(\mathcal{D})^0 \neq \emptyset$, $\mathcal{P}re(\mathcal{D})^i \neq \emptyset$, $\mathcal{C}on(\mathcal{D})^i \neq \emptyset$, for $i = 1, \ldots, d$, and $\mathcal{P}re(\mathcal{D})^i = \mathcal{C}on(\mathcal{D})^i = \emptyset$, for $i > d$, since for each $\mathcal{D}$–premise p there exists at least one $\mathcal{D}$–conclusion c that p is c–necessary. Then the following formula has breadth 1.

$$\begin{aligned}
&\mathcal{P}re(\mathcal{D})^0 \texttt{ implies (}\mathcal{C}on(\mathcal{D})^0 \texttt{ \& (}\\
&\quad \mathcal{P}re(\mathcal{D})^1 \texttt{ implies (}\mathcal{C}on(\mathcal{D})^1 \texttt{ \& (}\\
&\qquad \ldots\\
&\qquad\quad \texttt{(}\mathcal{P}re(\mathcal{D})^m \texttt{ implies }\mathcal{C}on(\mathcal{D})^d \texttt{) ...))))},
\end{aligned} \tag{3}$$

Additionally, the formula has the minimal depth among these statements of packet $\mathcal{D}$ that have the breadth equal 1.

So far in this section we ignore information about variables. Generally, the packet's statement has to be preceded by a sequence of quantifiers that bind occurring variables. The packet extraction methods that take into consideration variables has been described in an earlier paper [11]. Obviously, the number of universal and existential vertices of a packet suggest two natural characteristics that correspond to the number of variables, which are bound by universal and existential quantifiers in the packet's statement. The arithmetical hierarchy of a theorem's statements, considered in the work of Alama [1], takes into consideration the complexity of the antecedent that we omit in considerations. It is important to note that in [11] only packet's statements not in prenex normal form were considered. Moreover, there is an *a priori* assumption that the existential quantifier corresponding to a $\mathcal{D}$–existential vertex e has to be preceded by every corresponding statement of e–necessary $\mathcal{D}$–premises. In this section we present only a simple justification of this assumption, i.e., without this assumption modification of reasoning is extremely complicated and generates unnatural proof scripts.

```
α0   Lemma_P: for x be set st P[x] ex y be set st Q[y] &
        for z be Subset of y st R[z] holds ex r be Relation of y,z st S[r]
     proof
α1 :    let x be set;
α2 :    assume A1: P[x];
γ' :    consider y be set such that A2:y=F(x) by A1;
α3 :    take y;
δ' :    thus Q[y] by A2;
α4 :    let z be Subset of y;
α5 :    assume A3: R[z];
ζ' :    consider r be Relation of y,z such that A4: S[r] by A3;
α6 :    take y;
ζ'' :   thus S[r] by A4;
     end;
α :  let x be set;
β :  A1: P[x];
θ1 : consider y be set such that B1: Q[y] & for z be Subset of y st R[z] holds
        ex r be Relation of y,z st S[r] by A1, Lemma_P;
δ :  A3: Q[y] by B1;
ϵ :  consider z be Subset of y such that A4: R[z] by A3;
θ2 : consider r be Relation of y,z such that B2: S[r] by B1;
η :  A6: T[r] by B2;
```

Fig. 3. The modification of the proof script (presented in Fig. 1) which represents the extraction of the packet $\mathcal{P}$, where the packet's statement is presented in (4).

According to the *a priori* assumption, in every statement of the packet $\mathcal{P}$ presented in Fig. 1, the premise `P[x]` (β) has to precede both existential quantifiers `ex y be set`; `ex r be Relation of y,z` (γ, ζ) and the premise `R[z]` (ϵ) has to precede only the second one. In consequence, we obtain the following formula:

```
for x be set st P[x] ex y be set st Q[y] & for z be Subset of y st
   R[z] holds ex r be Relation of y,z st S[r]                        (4)
```

that corresponds to the second formula in (1). A generated deduction that demonstrates the formula and a modified part of reasoning remaining after the packet's extraction is presented in Fig. 3. Note that the generated deduction has to contain six skeleton steps that correspond to universal (α_1, α_4) and existential (α_3, α_6) quantifiers; and correspond to antecedents (α_2, α_5). Similarly, variables (`y`, `r`) that are introduced to a reasoning in the area of a packet have to be reintroduced in the remaining parts of the reasoning after the packet's extraction (θ_1, θ_2). It is important to note that the resulting proof script, see Fig. 3, is the shortest of all proofs where the packet $\mathcal{P}$ is extracted as a lemma.

4 Statistical Results and Discussion

In our study we analyze 55160 theorems and 53839 lemmas that are collected in the MML version 5.32.1234. For us, a "*theorem*" is only this item in the MML that is explicitly called `theorem` in the Mizar syntax. Every other step that

has a nested deduction as a justification, not only on the top level of its proof script, is called a *preselected lemma*. We call a preselected lemma "*lemma*" if the statement does not constitute a correctness condition or a property, where the statement is imposed by the Mizar syntax (for more detail see the current overview of the Mizar system [2]), e.g., in every definition of a functor, we have to demostrate the existence and the uniqueness conditions, if the functor value is not defined by a term.

4.1 Premises and Conclusions

As we have expected, the analysis of the number of explicitly formulated premises in theorems and lemmas shows that lemmas have on average 1.8 times less premises than theorems do. Note that the average number of premises in lemmas is equal 0.76. Additionally, more than half of lemmas does not have premises and only 6.25% do not have more then 2, whereas for comparison 19.82% of theorems have more than 2 (see Fig. 4). The results for the average number of conclusions are not significantly different for lemmas and theorems (1.15 and 1.38 respectively), and in both cases the percentage of statements dramatically decreases with the increase of the number of conclusions. Obviously, not all premises are visible in the statement of a lemma. Indeed, if the statement of a step (1) is used as a premise in one of the steps of the nested reasoning that is the justification of a step (2), and both steps (1) and (2) are in the same level of nesting, then the premise does not occur in the statement of the step (2). Such hidden premises (1) are called *implicit premises*, in contrast to these that are formulated in the statement of the step (2), and these are called *explicit premises*. However, a similar situation occurs in the case of theorems if we use the previously proven facts that are formulated on the top level of proof scripts. Obviously, these facts can also be used in lemmas. But if we sum up *explicit* and *implicit* premises, then the percent of formulas in the MML with the same number of such premises does not distinguish statements of lemmas and theorems (see the ratio of percents (solid line) at the diagram that represent number of all premises presented at Fig. 4). However, the set of *implicit* premises can be defined in terms of the notion of packet $\mathcal{D}$ and is equal to $\mathcal{P}re(\mathcal{D})^0$ (see (2)). Therefore, as an appropriate numerical characteristic of a packet we choose the cardinality of $\mathcal{P}re(\mathcal{D}) \setminus \mathcal{P}re(\mathcal{D})^0$.

We can also analyze the arithmetical hierarchy of *explicit* and *implicit* premises. However, the main part of premises contains identifiers of local constants (mainly lemmas) or reserved variables (theorems). In consequence, the main part of *implicit* lemma's premises is at 0–level on the arithmetic hierarchy. Therefore, in our study, we preceded by a sequence of necessary universal quantifiers every *implicit* premise that contains such identifers. But then the percent of *implicit* premises on the level of the arithmetic hierarchy that occurs in lemmas and theorems is almost identical.

A bit different is the case of *implicit* premises, since according to the assumptions of Section 3, we should analyze *explicit* premises as they were originally formulated by the author of a proof script. In that case the average level of

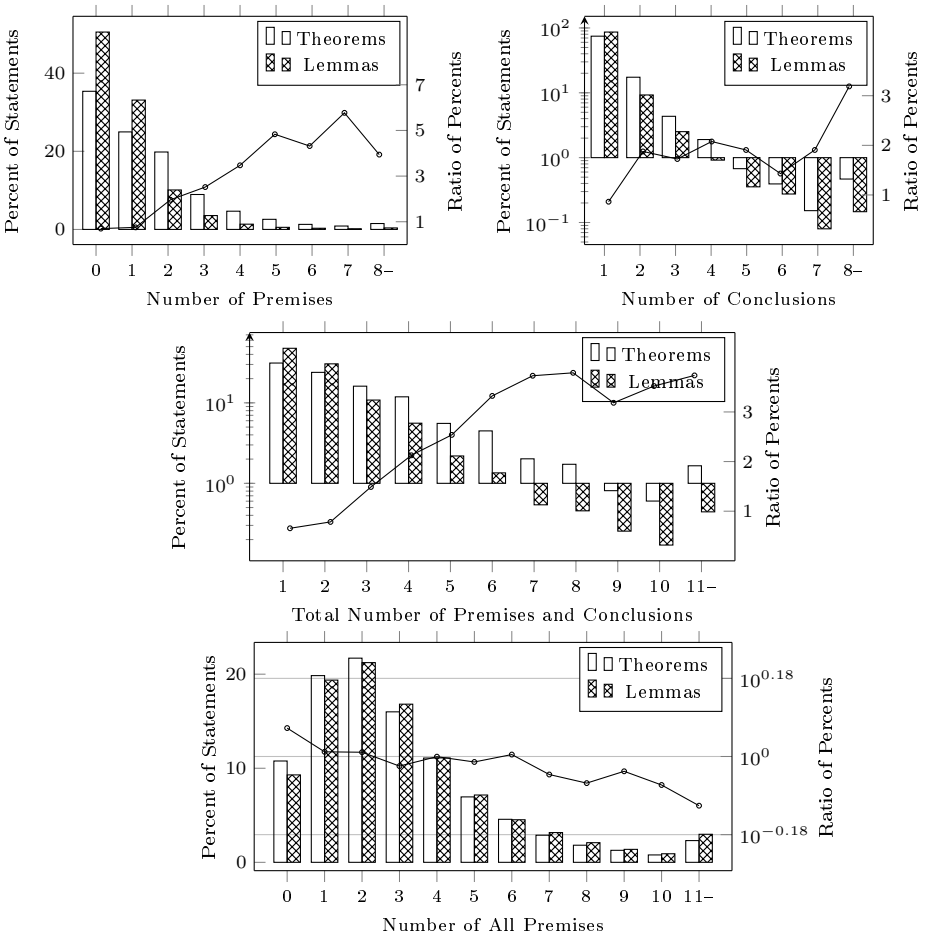


Fig. 4. The percent of statements (bars) with the same number of premises; conclusions; premises and conclusions together; *explicit* and *implicit* premises together; and also the ratio of these percents (solid lines) for theorems to lemmas.

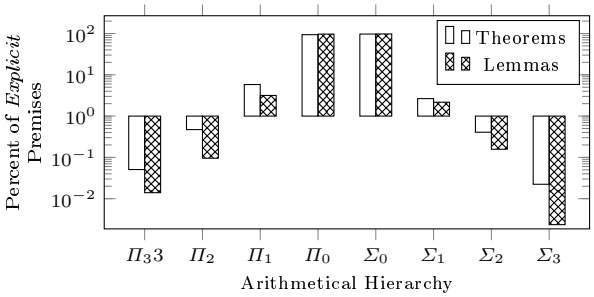


Fig. 5. The percent of *explicit* premises on the same level of the arithmetical hierarchy that occurs in lemmas and theorems.

arithmetic hierarchy is 0.116 and 0.027 for Π_*; 0.0572 and 0.0196 for Σ_* in *explicit* premises occurring in lemmas and theorems, respectively, where for simplicity of notation a formula ϕ is Π_* (Σ_*) if ϕ is Π_i (Σ_i) formula for some i. Note also that non-literal premises, if they are Π_* (Σ_*), then they occur 1.92 (1.32) times more often in theorems then in lemmas. Additionally, the number of premises drastically decreases with increasing level of arithmetical hierarchy, on average 10.9 and 18.2 times in lemmas and theorems, respectively. Therefore, as a characteristic of a packet $\mathcal{D}$ we should take into consideration not only the cardinality of $\mathcal{P}re(\mathcal{D}) \setminus \mathcal{P}re(\mathcal{D})^0$, but also the level of arithmetical hierarchy of statements formulated in vertices of $\mathcal{P}re(\mathcal{D}) \setminus \mathcal{P}re(\mathcal{D})^0$, to reduce the number of more complex one.

4.2 Variables bounded by universal and existential quantifiers

As in the previous section, we assume that every formula is preceded by a sequence of necessary universal quantifiers, which fix all local constants (mainly lemmas) or reserved variables, called together *parameters*. Note that the average number of parameters in lemmas and theorems is equal 3.40 and 1.54, respectively (see Fig. 6). Additionally, 54.26% of theorems have less than 2 parameters and only 4.64% have more than 4 parameters, whereas for comparison 48.36% of lemmas have 3 ± 1 parameters and 36.75% have more than 4 parameters.

For simplicity of notation, we called a variable *universally quantified* if it is a parameter or it is bounded by a universal quantifier and we call a variable *existentially quantified* if is bounded by an existential quantifier. Analyzing the number of variables bounded by universal quantifiers in formula, we obtain that percent of lemmas and theorems with the same value is almost identical, for the most popular values (3, 4, see Fig. 6). However, the ratios of these percents for theorems to lemmas is less then 1 for popular values (2–5). Note also that existentially quantified variables occur very rarely in formulas and the rations of the statement percents with the same number of such variable for theorems to lemmas is less then 1 if and only if the number is less then 2.

4.3 The Depth and Breadth

As we have expected, the main part of statements have depth less than or equal to 1. Additionally only 7 theorems in the MML have the depth greater than 3 (3 of them describe differentiability in higher dimensions, for more detail see the relevant Mizar article [3]). Similarly, the main part of statements have breadth equal 1 and 42% of them are formulated without premises (37.25% theorems and 51.34% lemmas with breadth equal 1). Moreover, only 23.42% of formulas that have the depth greater than 0 or the breadth greater than 1 are used as lemmas (see Fig. 7).

We are aware that the described result is only a small step forward. However this is the first promising result concerning this questions. Analysis of human readers' opinions, especially the Mizar's users, does not make it possible to obtain more important results, which can be summarized by the statement: a lemma

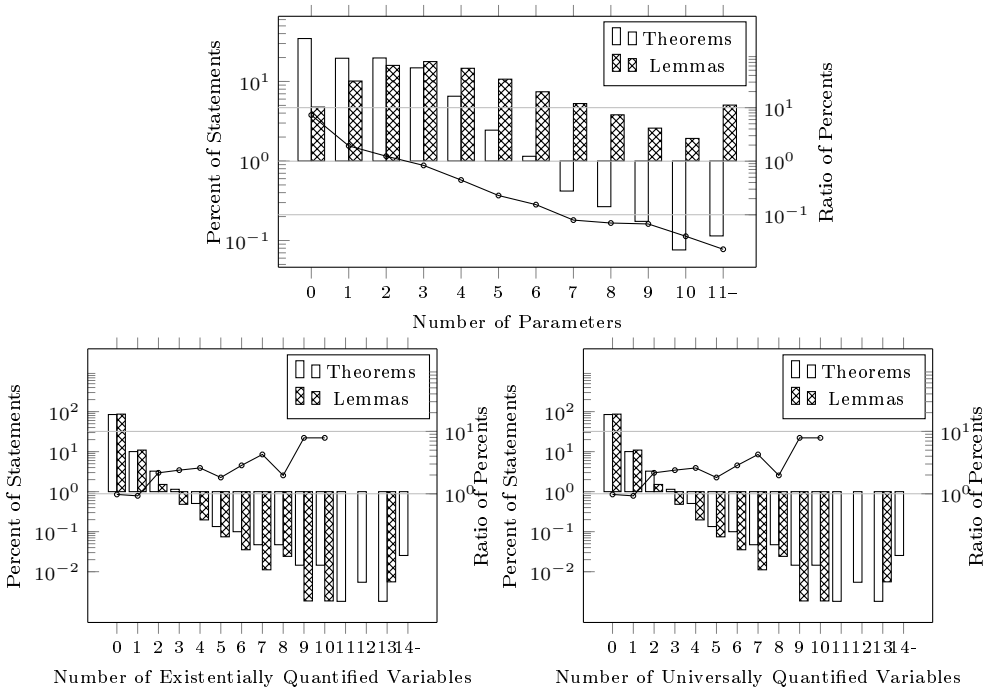


Fig. 6. The percent of statements (bars) with the same number of parameters; existentially quantified variables; universally quantified variables that occurs in lemmas and theorems, and also the ratio of these percents (solid lines) for theorems to lemmas.

corresponds to a separate fragment of reasoning between "two throats" in the proof that correspond to the premise and the thesis of the lemma (A.Trybulec). Additionally in [11] an artificial property of the package (the closeness of packets with respect to directed paths) has been described.

It is crucial in the process of packets' extraction, but the determination of the influence of this property on the statement of existing lemmas was extremely counterintuitive. This impact is our "small step forward" and is described in Section 3 as a depth and breadth of formula. These two parameters seem to be also non-intuitive but they can be easily calculated for formulas and they well improve the result of a method that distinguishes formulas that occur in lemmas and theorems, and base only on the number of premises, conclusions and bound variables.

5 Conclusions

In this paper we describe a next stage in the research on methods that improve proof legibility realized in [11] that base on rebuilding the proof structure, either

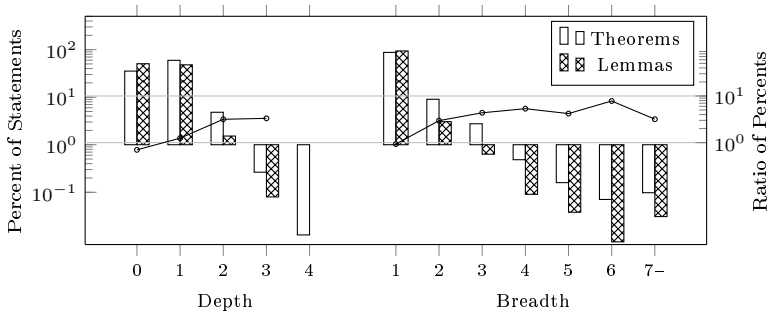


Fig. 7. The percent of statements (bars) with the same depth; breadth that occurs in lemmas and theorems, and also the ratio of these percents (solid lines) for theorems to lemmas.

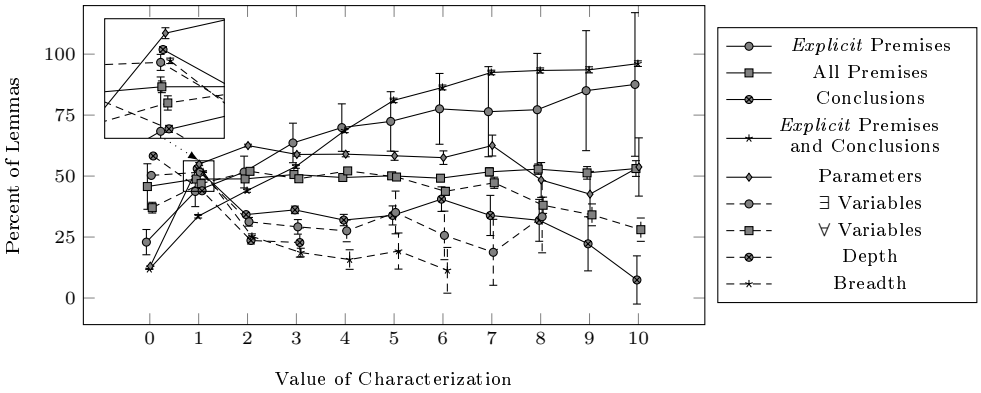


Fig. 8. The percent of statements with the same value of a characteristic that are used as lemmas and also 95% confidence intervals of the percent.

encapsulating sub-deductions (packets) in the form of a nested lemma or extracting them as lemma. We define characteristics of the packet's statement and also we indicate the most appropriate values that are helpful in deciding whether to extract a packet as a lemma or not. Moreover, the proposed characteristics based on the packet's statement determine a packet position in an abstract proof graph. This dependence is crucial, since in our approach we analyze the probability that in human readers' opinions, a package should be extracted basing on the lemmas and theorems statement collected in the MML that are not generated from packets.

This research showed that for every packet we can calculate the probability of using in the MML the that the packet' statement is used in the MML to formulate a theorem or a lemma; and also it indicates which of these two types is more probable, in respect to each proposed characteristic. The two non-intuitive characteristics of a statement have proved to be especially important. The depth and the breadth well improve distinction that based only on the num-

ber of premises, conclusions and bound variables. However, still an open problem to investigate is to determine impact of individual characteristics on the final qualitative assessment of a packet.

References

1. J. Alama. Sentence Complexity of Theorems in Mizar. *CoRR*, abs/1311.1915, 2013.
2. G. Bancerek, C. Bylinski, A. Grabowski, A. Korniłowicz, R. Matuszewski, A. Naumowicz, K. Pąk, and J. Urban. Mizar: State-of-the-art and Beyond. In M. Kerber, J. Carette, C. Kaliszyk, F. Rabe, and V. Sorge, editors, *Intelligent Computer Mathematics - International Conference*, vol. 9150 of *LNCS*, 261–279, 2015.
3. N. Endou, H. Okazaki, and Y. Shidama. Higher-Order Partial Differentiation. *Formalized Mathematics*, 20(2):113–124, 2012.
4. G. Gonthier. A Computer-Checked Proof of the Four Colour Theorem. `http://research.microsoft.com/en-us/um/people/gonthier/4colproof.pdf`, 2005. [Online; accessed 19-May-2016].
5. G. Gonthier. Formal Proof—The Four-Color Theorem. *Notices of the AMS*, 55(11):1382—1393, 2008.
6. A. Grabowski. On the Computer-assisted Reasoning About Rough Sets. In B. Dunin-Kęplicz, A. Jankowski, A. Skowron, and M. Szczuka, editors, *International Workshop on Monitoring, Security, and Rescue Techniques in Multiagent Systems Location*, vol. 28 of *Advances in Soft Computing*, 215–226, 2005.
7. A. Korniłowicz. Tentative Experiments with Ellipsis in Mizar. In J. Jeuring, J. A. Campbell, J. Carette, Gabriel G. Dos Reis, P. Sojka, M. Wenzel, and V. Sorge, editors, *Intelligent Computer Mathematics 11th International Conference*, vol. 7362 of *LNAI*, 453–457, 2012.
8. A. Kornilowicz. On Rewriting Rules in Mizar. *Journal of Automated Reasoning*, 50(2):203–201, 2013.
9. A. Naumowicz and C. Byliński. Improving Mizar Texts with Properties and Requirements. In A. Asperti, G. Bancerek, and A. Trybulec, editors, *Mathematical Knowledge Management, Third International Conference*, vol. 3119 of LNCS, 290–301, 2004.
10. A. Naumowicz and A. Korniłowicz. A Brief Overview of Mizar. In S. Berghofer, T. Nipkow, C. Urban, and M. Wenzel, editors, *Proceedings of the 22nd International Conference on Theorem Proving in Higher Order Logics*, vol. 5674 of *LNCS*, 67–72, 2009.
11. K. Pąk. Methods of Lemma Extraction in Natural Deduction Proofs. *Journal of Automated Reasoning*, 50(2):217–228, 2013.
12. K. Pąk. Automated Improving of Proof Legibility in the Mizar System. In S. M. Watt, J. H. Davenport, A. P. Sexton, P. Sojka, and J. Urban, editors, *Intelligent Computer Mathematics - International Conference*, vol. 9150 of *LNCS*, 373–387, 2014.
13. K. Pąk. Readable Formalization of Euler's Partition Theorem in Mizar. In M. Kerber, J. Carette, C. Kaliszyk, F. Rabe, and V. Sorge, editors, *Intelligent Computer Mathematics - International Conference*, vol. 9150 of *LNCS*, 211–226, 2015.
14. J. Urban. XML-izing Mizar: Making Semantic Processing and Presentation of MML Easy. In M. P. Bonacina, editor, *4th International Conference Mathematical Knowledge Management 2005*, vol. 3863 of *LNCS*, 346–360, 2005.