Trends in Contemporary Computer Science, Podlasie 2014, ISBN 978-83-62582-58-7
by A. Gomolińska, A. Grabowski, M. Hryniewicka, M. Kacprzak, E. Schmeidel (Eds.)

Improving Legibility of Proof Scripts Based on Quantity of Introduced Labels*

Karol Pąk

Institute of Computer Science,
University of Bialystok, Poland
pakkarol@uwb.edu.pl

Abstract. Formal proof checking systems such as Mizar or Isabelle/Isar can verify the correctness of proof scripts, both easily readable and obscure. However for humans, e.g., those who analyse the main idea of a formal proof or redevelop fragments of reasoning to make them stronger, the legibility has substantial significance. Furthermore, proof writers create still more and more complex deductions that cannot be shortened to several steps by any tools currently available. Therefore, it is important to better understand how we can facilitate the work of script readers modifying the order of independent deduction steps or reorganise the proof structure by extracting lemmas that are obtained automatically.
In this paper we present experimental result obtained with a method that improves proof legibility based on human short-term memory and we explore its impact for realisation of other, also popular methods.

Keywords: Operations on languages, Legibility of proofs, Proof assistants

1 Introduction

1.1 Motivations

One of the most important tasks of proof assistants such as Mizar [17, 23] or Isabelle/Isar [24] is checking proof step correctness. Clearly, inferences that are obvious for proof writers should be also obvious for checkers and vice versa. However, the notion of "obviousness" has often different meaning for humans and checkers [5, 21]. Generally, this difference is not so important problem if we do not need to analyze, adapt or modify the existing formal proofs scripts [10, 12]. According to the opinion of some proof writers, repairing a justification in re-developed fragments of reasoning that ceased to be acceptable, can be very difficult. We can observe similar situation in the area of proof script legibility. A proof assistant can check correctness of every syntactically correct formal proof scripts, even automatically generated, excessively large, or written in a chaotic way. However, any attempt to analyse such scripts by a human is extremely difficult or even impossible. The main reason for this situation is often the fact that proof scripts are created in an artificial language which "tries" to be similar to the one

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

that occurs in the traditional mathematical practice. Another reason is that readability does not come of zero cost on the side of proof script developers. Namely, the legibility strongly depends on the amount of effort invested in enhancing the readability by the author of a given proof script. Unfortunately, authors often do not care about legibility of their proof scripts. It is often a consequence of their assumption that no one, with the exception of a proof checker, will want to analyse their proof scripts. Furthermore if somebody would want to do it, then some tool rather than themselves should improve the legibility of their scripts. The experience with proof assistants shows that reading proof script is often unavoidable, e.g., if we adapt or modify existing proofs [8, 11], or if several authors cooperate on a common formalisation. The important point to note here is that creating such a tool that enhances legibility can in general be NP-complete [20].

1.2 Proposed approach

In this paper we present a next stage [18, 20] in the enhancing legibility of proof scripts based on a modification of the order of independent deduction steps. We concentrate on a modification that minimises the number of steps decorated by labels. Note that we need to refer to a premise, if we want to use it in the justification of a step. But in some cases to do this in the Mizar system we do not have to use the label of this premise. Indeed, referencing a step by its label may be replaced in the Mizar system by the `then` construction, if the step is located in the directly preceding step that refers to it (for more detail see [9, 17]). Additionally, if each reference to a fixed step can be realised by the `then` construction, then the label that decorates this step in unnecessary and can be removed from the proof script. Therefore, it is possible to minimise the number of labels that are introduced in proof scripts.

Analysing deductions contained in proof scripts of the Mizar Mathematical Library (MML), one may reasonably conclude that the number of introduced labels in a reasoning can be rarely minimised so that a proof reader can keep them in human short-term memory. However, minimisation to such a number is often possible in one-level deductions (i.e., substructures of a proof where nested lemmas are ignored) that are located on all levels of nesting (see Fig. 2). Note that only 4.5% of one-level deductions that occur in MML (Version `5.22.1191`) have more than 7 labels. Additionally, G. A. Miller shows that the capacity of human short-term memory is 7 ± 2 elements [15]. This limitation is also recognised in modern scientific literature that concerns human perception [3, 4]. Clearly, the capacity of memory decreases quickly with time and it is smaller in the case of similar information [25]. However, this capacity can be extended through training [6]. Therefore, small departure beyond the number 7 should be acceptable and this is the case for MML where the number of labels is in the range 5-10 [14].

In this paper we represent experimental results obtained with minimisation of the number of introduced labels. We combined this result with other criteria that improve proof scripts legibility and have been already recognised by the scientific community of people who write proof scripts in Mizar [18, 19] as well as in other systems [2, 13, 22]. Since, optimisation of legibility criteria in most cases is NP-hard [20], we present readability enhancements obtained with the help of the SMT-solver Z3 [16].

2 Labeled steps in terms of proof graphs

To formulate a criterion that minimises the number of introduced labels and the influence of this criterion implementation for the realisation of other similarly popular

criteria, we need to set the terminology and notation. Let $G = \langle V, A\rangle$ be a DAG and a vertex $u \in V$. We use the following notation:

$$\begin{array}{ll} N_G^-(u) :- \{v \in V : \langle v, u\rangle \in A\} & \text{(incoming arcs)}, \\ N_G^+(u) :- \{v \in V : \langle u, v\rangle \in A\} & \text{(outgoing arcs)}. \end{array} \tag{1}$$

Let A_1 be a subset of A. An arc $\langle u, v\rangle \in A$ is called an A_1*-arc* if $\langle u, v\rangle \in A_1$. A sequence $p = \langle u_1, u_2, \ldots, u_n\rangle$ of vertices of G is called an A_1*–path* if $\langle u_1, u_{i+1}\rangle$ is an A_1-arc for each $i = 1, 2, \ldots, n{-}1$. We identify here topological sortings of G, called also *linearisations*, with one-to-one functions $\tau : V \to \{1, 2, \ldots, |V|\}$, where $\tau(u) < \tau(v)$ for each A–arc $\langle u, v\rangle$.

An abstract model of a proof graph that represents the structure of natural deduction proofs, even with potentially nested subreasonings is fully explained in [18, 20]. However for our purposes it is enough to consider a simplified model which is represented by a DAG with the structure that ignores nested lemmas (i.e., one-level deductions). It is worth pointing out that the number of introduced labels on a one-level deduction in a proof script is independent of the number of introduced labels on another one-level deduction of such script. A DAG $\mathcal{D} = \langle \mathcal{V}, \mathcal{A}\rangle$ with a distinguished set of arcs $\mathfrak{R}(\mathcal{D}) \subseteq \mathcal{A}$ is called a *simple abstract proof graph*. The vertices of $\mathcal{D}$ represent steps of the reasoning and $\mathcal{A}$–arcs represent the flow of information between different steps of the reasoning. A $\mathfrak{R}(\mathcal{D})$–arc, called a *reference arc*, represents the information flow between a premise (the tail of the arc) and a place of its use (the head of the arc). The other $\mathcal{A}$-arcs 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 into the reasoning and a step that uses this variable in its expression.

```
1: reserve i, n, m for Nat;

   theorem
2:    i in Seg n implies i+m in Seg(n+m)
   proof
3:    assume A1: i in Seg n;
4:    then A2: 1 <= i by FINSEQ_1:1;
5:    i <= i+m by NAT_1:11;
6:    then A3: 1 <= i+m by A2,XXREAL_0:2;
7:    i <= n by A1,FINSEQ_1:1;
8:    then i+m <= n+m by XREAL_1:7;
9:    hence thesis by A3,FINSEQ_1:1;
   end;
```

```
   theorem
1:    fixes n m i::nat
2:    shows "i ∈ {k::nat. 1<=k & k<=n}==>
         i+m ∈ {k::nat. 1 <= k & k <= n+m}"
   proof -
3:    assume A1: "i ∈ {k::nat. 1 <= k & k <= n}"
4:    then have A2: "1 <= i" by simp
5:    have "i <= i+m" by simp
6:    then have A3: "1 <= i+m" using A2 by simp
7:    have "i <= n" using A1 by simp
8:    then have "i+m <= n+m" by simp
9:    then show ?thesis using A3 by simp
   qed
```

Fig. 1. An example proof script written in the Mizar language that is contained in [1] and its reformulation to the Isabelle/Isar language.

As an illustration, let us consider an example shown in Fig. 2 that represents the structure of proof scripts presented in Fig. 1, where solid arrows correspond to reference arcs that are also $\mathcal{A}$-arcs, and dashed arrows correspond to $\mathcal{A}$-arcs that are not reference arcs. Additionally, the term `Seg` n that occurs in Fig. 1 represents the set $\{1, 2, \ldots, n\}$. Clearly, both arcs and nodes of the abstract proof graph are not labeled. However we label each element in the actual graph only to simplify their identification. Note that this graph contains two one-level deductions (vertices 1–2 and vertices 3–9) and additionally $\twoheadrightarrow$ arrows that correspond to meta-edges of proof graphs,

which do not occur in our simplified model. We only recall that meta-edges represent dependencies between one-level deductions, i.e., between a step (e.g., the vertex 2) that as a justification contains the nested reasoning and each step of this reasoning (e.g., vertices 3–9). It is easily seen that in such a simplified model we have to take into consideration additional hidden dependencies that can occur between steps in one-level deductions. As an illustration note that the 1st step has to occur before the 2nd step, even if variables introduced in the 1st step do not occur in the statement of the 2nd step. Indeed, e.g., the variable `i` is used in the statement of the 3rd step that occurs in the nested reasoning that justify the 2nd step.

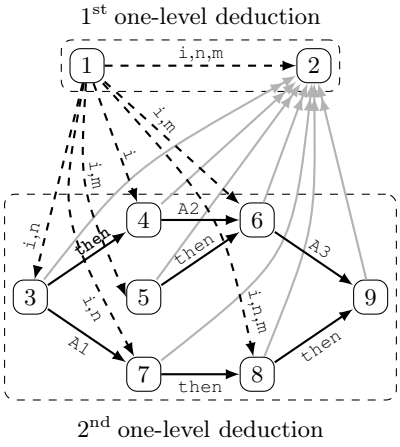


Fig. 2. The abstract proof graph illustrating the structure of the reasoning presented in Fig. 1.

Let $\mathcal{D} = \langle \mathcal{V}, \mathcal{A} \rangle$ be a one-level deduction. For simplicity, we assume that $\mathcal{A}$ contains additionally every hidden dependence between vertices of $\mathcal{V}$, and denote by $\mathfrak{R}(\mathcal{D})$ the set of reference arcs and hidden ones. We mark by $\texttt{then}(\mathcal{D})$ the set of references that can be replaced by the `then` construction. However, to study the general case, without the Mizar context, we will assume only the relation between distinguished sets of arcs in $\mathcal{D}$ that $\texttt{then}(\mathcal{D}) \subseteq \mathfrak{R}(\mathcal{D}) \subseteq \mathcal{A}$. Therefore, in further considerations we mean $\texttt{then}(\mathcal{D})$, $\mathfrak{R}(\mathcal{D})$ simply as two sets $\mathcal{A}_1$, $\mathcal{A}_2$, respectively, where $\mathcal{A}_1 \subseteq \mathcal{A}_2 \subseteq \mathcal{A}$.

Recall that we identify the arrangement of reasoning steps that correspond to $\mathcal{V}$ in a proof script with a topological sortings of $\mathcal{D}$. Let us consider $\sigma \in TS(\mathcal{D})$. We define a metric $d_\sigma : \mathcal{V} \times \mathcal{V} \to \mathbb{N}$ that is called σ–distance and is given by $d_\sigma(v,u) = |\sigma(v) - \sigma(u)|$ for all $v, u \in \mathcal{V}$. We call a vertex $v \in \mathcal{V}$ a $\texttt{then}_{\mathcal{A}_1}(\sigma)$–step if v corresponds to a step that refers to the directly preceding step using a $\mathcal{A}_1$–arc (e.g., the vertex 4). We denote by $\texttt{then}_{\mathcal{A}_1}(\sigma)$ the set of such steps given by

$$v \in \texttt{then}_{\mathcal{A}_1}(\sigma) \iff \big(\sigma(v) \neq 1 \wedge \langle \sigma^{-1}(\sigma(v)-1), v \rangle \in \mathcal{A}_1\big)\,. \tag{2}$$

We call a vertex $v \in \mathcal{V}$ σ–*labeled* if at least once we have to use a label to refer to the statement of a step that corresponds to $\mathcal{V}$ (e.g., the vertex 3). The set of all σ–labeled

vertices, denoted by $\mathtt{lab}_{\mathcal{A}_1,\mathcal{A}_2}(\sigma)$, is defined as follows:

$$v \in \mathtt{lab}_{\mathcal{A}_1,\mathcal{A}_2}(\sigma) \iff \underset{u\in\mathcal{V}}{\exists}\langle v,u\rangle \in \mathcal{A}_2 \ \wedge \big(\langle v,u\rangle \in \mathcal{A}_1 \Longrightarrow d_\sigma(v,u) > 1\big)\,. \tag{3}$$

However, according to an additional syntax restriction of Mizar that prohibits referring to steps that introduce variables into the reasoning, we have to consider also the following set of σ–labeled vertices:

$$v \in \mathtt{lab}^{\mathtt{MIZ}}_{\mathcal{A}_1,\mathcal{A}_2}(\sigma) \iff \underset{u\in\mathcal{V}}{\exists}\langle v,u\rangle \in \mathcal{A}_2 \wedge \Big(\big(\langle v,u\rangle \in \mathcal{A}_1 \Longrightarrow d_\sigma(v,u) > 1\big) \vee \big(\underset{w\in\mathcal{V}}{\exists}\langle v,w\rangle \in \mathcal{A}\setminus\mathcal{A}_1\big)\Big)\,. \tag{4}$$

We call $|\mathtt{lab}^{\mathtt{MIZ}}_{\mathcal{A}_1,\mathcal{A}_2}|$ the *lab–parameter*.

Based on the notions described above we can formulate the method of improving legibility that corresponds to the lab-parameter as the following decision problems:

The 1st Method of Improving Legibility ($\mathbf{MIL}_{\mathtt{lab}}$):
INSTANCE: A DAG $\mathcal{D}=\langle\mathcal{V},\mathcal{A}\rangle$, subsets $\mathcal{A}_1\subseteq\mathcal{A}_2\subseteq\mathcal{A}$, and a positive integer $K\leqslant|\mathcal{V}|$.
QUESTION: Does there exist $\sigma\in TS(\mathcal{D})$ for which $|\mathtt{lab}_{\mathcal{A}_1,\mathcal{A}_2}(\sigma)|\leqslant K$?

The 1st Method of Improving Legibility limited to the Mizar system ($\mathbf{MIL}^{\mathtt{MIZ}}_{\mathtt{lab}}$):
INSTANCE: A DAG $\mathcal{D}=\langle\mathcal{V},\mathcal{A}\rangle$, subsets $\mathcal{A}_1\subseteq\mathcal{A}_2\subseteq\mathcal{A}$, and a positive integer $K\leqslant|\mathcal{V}|$.
QUESTION: Does there exist $\sigma\in TS(\mathcal{D})$ for which $|\mathtt{lab}^{\mathtt{MIZ}}_{\mathcal{A}_1,\mathcal{A}_2}(\sigma)|\leqslant K$?

3 Optimisation of the lab–parameter

The complexity problem of improving legibility methods that corresponds to the lab–parameter optimisation has been studied in [20]. It has been shown that $\mathrm{MIL}_{\mathtt{lab}}$ is NP-complete and $\mathrm{MIL}^{\mathtt{MIZ}}_{\mathtt{lab}}$ is solvable in polynomial time. Here we concentrate first on properties of the polynomial time procedure that optimises the lab–parameter for Mizar proof scripts. Then we show that the $\mathrm{MIL}_{\mathtt{lab}}$ method for one-level deductions that potentially occur in Isabelle proof scrips is NP-hard.

3.1 The lab–parameter in the Mizar system

Let us fix a one-level deduction DAG $\mathcal{D}=\langle\mathcal{V},\mathcal{A}\rangle$ with two distinguished subsets of arcs $\mathcal{A}_1\subseteq\mathcal{A}_2\subseteq\mathcal{A}$. First note that some of the vertices of $\mathcal{D}$ have to be σ–labeled regardless of the σ choice. Indeed, every $v\in\mathcal{V}$ for which at last one of the following properties holds:
(i) $|N^+_{\langle\mathcal{V},\mathcal{A}_1\rangle}(v)|>1$,
(ii) $|N^+_{\langle\mathcal{V},\mathcal{A}_2\rangle}(v)|>|N^+_{\langle\mathcal{V},\mathcal{A}_1\rangle}(v)|$,
(iii) $|N^+_{\mathcal{D}}(v)|>|N^+_{\langle\mathcal{V},\mathcal{A}_2\rangle}(v)|>0$,
has to be σ–labeled in all $\sigma\in TS(\mathcal{D})$. Mark the set of such vertices by $\mathcal{L}^{\mathtt{MIZ}}_{\mathcal{A}_1,\mathcal{A}_2}$. Note also that if we remove all $\mathcal{A}$–arcs outgoing from vertices of $\mathcal{L}^{\mathtt{MIZ}}_{\mathcal{A}_1,\mathcal{A}_2}$ then the digraph obtained in this way, denoted by $\mathcal{D}'$, is a forest where every connected maximal tree is an arborescence (i.e., a rooted tree where all arcs are directed from leaves to the root). Additionally, every arc of $\mathcal{D}'$ is simultaneously $\mathcal{A}_2$–arc, $\mathcal{A}_1$–arc, and $\mathcal{A}$–arc, hence

$\mathtt{lab}^{\mathtt{MIZ}}_{\mathcal{A}_1,\mathcal{A}_2}(\sigma) \setminus \mathcal{L}^{\mathtt{MIZ}}_{\mathcal{A}_1,\mathcal{A}_2}$ has to contain at least $|N^+_{\mathcal{D}'}(v)| - 1$ elements of $N^+_{\mathcal{D}'}(v)$ if only $N^+_{\mathcal{D}'}(v)$ is nonempty for each $v \in \mathcal{V}$. As it has been proven in [20], for each set of vertices that contain $\mathcal{L}^{\mathtt{MIZ}}_{\mathcal{A}_1,\mathcal{A}_2}$ and exactly $|N^+_{\mathcal{D}'}(v)| - 1$ elements of every nonempty $N^+_{\mathcal{D}'}(v)$, there exists a topological sorting $\sigma \in TS(\mathcal{D})$ for which $\mathtt{lab}^{\mathtt{MIZ}}_{\mathcal{A}_1,\mathcal{A}_2}(\sigma)$ is equal to this set. Clearly, in this topological sorting every non-selected vertex $u \in N^+_{\mathcal{D}'}(v) \setminus \mathtt{lab}^{\mathtt{MIZ}}_{\mathcal{A}_1,\mathcal{A}_2}(\sigma)$ has to be located in the directly preceding step v, since u is not decorated by a label. Additionally, this holds for each choice of $|N^+_{\mathcal{D}'}(v)| - 1$ elements of $N^+_{\mathcal{D}'}(v)$. Therefore, we can modify this choice in such a way that an arbitrary step of $N^+_{\mathcal{D}'}(v)$ can become not labeled. Hence from this we can conclude that the `lab`–parameter is minimal if each vertex v that "refers" to at least one "premise" with exactly one incoming $\mathcal{A}$-arc, has to contain at least one such premise that is located directly before v or more precisely:

Proposition 1. *Let $\mathcal{D} = \langle \mathcal{V}, \mathcal{A} \rangle$ be a DAG with two distinguished sets of arcs $\mathcal{A}_1 \subseteq \mathcal{A}_2 \subseteq \mathcal{A}$. Then $\mathtt{lab}^{\mathtt{MIZ}}_{\mathcal{A}_1,\mathcal{A}_2}(\sigma)$ has the smallest possible size if and only if, for every $v \in \mathcal{V}$ it holds that*

$$N^-_{\mathcal{D}}(v) \cap L \neq \emptyset \Longrightarrow \sigma^{-1}(\sigma(v)-1) \in L, \tag{5}$$

where $\sigma \in TS(\mathcal{D})$ and $L = \{v \in \mathcal{V} : |N^+_{\langle \mathcal{V}, \mathcal{A}_1 \rangle}(v)| = |N^+_{\langle \mathcal{V}, \mathcal{A} \rangle}(v)| = 1\}$.

3.2 The `lab`–parameter in the Isabelle/Isar system

Now we show that the minimisation of the `lab`–parameter for Isabelle/Isar proof scripts is NP-hard. To achieve this we indicate a family of correct proof scripts for which the minimisation of the `lab`–parameter is equally hard as the minimisation of the size of a vertex cover.

In this paper, we do not concentrate on a full explanation of how the known NP-complete problem Vertex Cover (see GT1 in [7]) is reducible to the $\mathrm{MIL}_{\mathtt{lab}}$ problem (for more details see [20]). We present only a way to create proofs written in the Isabelle/Isar system that have structures described by graphs obtained in this reduction. In this way we show that difficult proof structures are indeed representable there.

Vertex Cover (**VC**):

Instance: An (undirected) graph $G = \langle V, E \rangle$ and a positive integer $K \leqslant |V|$.

Question: Is there a vertex cover of size at most K, i.e., a subset $V' \subseteq V$ of size at most K such that for each edge $\{u, v\} \in E$ at least one of u or v belongs to V'?

Let $G = \langle V, E \rangle$, $K \leqslant |V|$ be an instance of VC. For simplicity we assume that $V = \{1, 2 \ldots, |V|\}$. The instance of $\mathrm{MIL}_{\mathtt{lab}}$ that is used in the reduction of VC to $\mathrm{MIL}_{\mathtt{lab}}$ is defined as follows. We construct a digraph $\mathcal{D} = \langle \mathcal{V}, \mathcal{A} \rangle$ and subsets $\mathcal{A}_1 \subseteq \mathcal{A}_2 \subseteq \mathcal{A}$ given by:

$$\begin{aligned} \mathcal{V} &:= V \times \{0, 1\}, \\ \mathcal{A}_1 &:= \{\langle \langle v, 0 \rangle, \langle v, 1 \rangle \rangle : v \in V\}, \\ \mathcal{A}_2 &:= \{\langle \langle v, 0 \rangle, \langle v, 1 \rangle \rangle : v \in V\} \cup \{\langle \langle v, 0 \rangle, \langle u, 1 \rangle \rangle : \{v, u\} \in E\}, \\ \mathcal{A} &:= \mathcal{A}_2. \end{aligned} \tag{6}$$

Obviously, $\mathcal{D}$, $\mathcal{A}_1$, and $\mathcal{A}_2$ determine a one-level deduction with two distinguished subsets of arcs. Additionally, this deduction together with K is an instance of $\mathrm{MIL}_{\mathtt{lab}}$ problem. Let us remind that the main idea of this reduction based on the fact that to obtain a vertex cover, for every edge $\{v, u\} \in E$, at least one of $\langle v, 0 \rangle$, $\langle u, 0 \rangle$, has to belong to $\mathtt{lab}_{\mathcal{A}_1,\mathcal{A}_2}(\sigma)$ for each $\sigma \in TS(\mathcal{D})$.

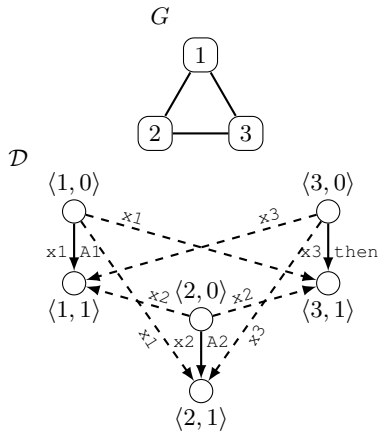
To create Isabelle/Isar proof scripts that correspond to the constructed deduction, we associate:

```
obtain xi::nat where Ai: "xi=i" by simp
```

with every vertex of the form $\langle i, 0\rangle$ and

```
have "xi=i & (xj1=xj1 & ... & xjn=xjn)" using Ai by simp
```

with every vertex of the form $\langle i, 1\rangle$, where $i \in V$, $\{j_1, j_2, \ldots, j_n\} = N^-_{\langle \mathcal{V}, \mathcal{A}_2 \setminus \mathcal{A}_1\rangle}(v)$. For illustration, an example of a reasoning that follows this pattern is presented in Fig. 3. It is simple to observe that every topological sorting of $\mathcal{D}$ organises such steps in the reasoning acceptable by the proof checker, since every occurring statement is "obvious". Additionally, this linearisation ensures that none of the variables and label identifers is used before their introduction in the reasoning. This completes the justification that computationally difficult instances can potentially occur in Isabelle/Isar proof scrips.




```
⟨1,0⟩: obtain x1::nat where
         A1: "x1=1" by simp
⟨2,0⟩: obtain x2::nat where
         A2: "x2=2" by simp
⟨3,0⟩ : obtain x3::nat where
          "x3=3" by simp
⟨3,1⟩ : then have"x3=3 & (x1=x1 & x2=x2)"
         by simp
⟨1,1⟩ : have "x1=1 & (x2=x2 & x3=x3)"
         using A1 by simp
⟨2,1⟩ : have "x2=2 & (x1=x1 & x3=x3)"
         using A2 by simp
```

Fig. 3. An example that illustrates the construction from Section 4, where considered vertex cover is equal to $\{1, 2\}$ and corresponds to vertices $\langle 1, 0\rangle, \langle 2, 0\rangle$, represented as steps decorated by `A1` and `A2`, respectively.

4 The `lab`-parameter in the process of improving other determinants of legibility

Our research is focused on the impact of the methods presented above on other popular methods such as increasing the number of **`then`** constructions (called **`then`**–parameter) or reducing the sum of all σ–distances of references (called **`dist`**–parameter). These methods of improving legibility of proofs can be formulated as the following two problems:

The 2nd Method of Improving Legibility ($\mathbf{MIL}_{\texttt{then}}$):
INSTANCE: A DAG $\mathcal{D} = \langle \mathcal{V}, \mathcal{A}\rangle$, a subset $\mathcal{A}_1 \subseteq \mathcal{A}$, and a positive integer $K \leqslant |\mathcal{V}|$.

QUESTION: Does there exist $\sigma \in TS(\mathcal{D})$ for which $|\texttt{then}_{\mathcal{A}_1}(\sigma)| \geqslant K$?

The 3rd Method of Improving Legibility ($\mathbf{MIL}_{\texttt{dist}}$):

INSTANCE: A DAG $\mathcal{D} = \langle \mathcal{V}, \mathcal{A} \rangle$, a subset $\mathcal{A}_2 \subseteq \mathcal{A}$, and a positive integer $K \leqslant \binom{|\mathcal{V}|+1}{3}$.

QUESTION: Does there exist $\sigma \in TS(\mathcal{D})$ for which $\sum_{\langle v,u \rangle \in \mathcal{A}_2} \sigma(u) - \sigma(v) \leqslant K$?

This impact has been studied on the MML database Version `5.22.1191` that includes 208590 one-level deductions. To obtain the result we use a brute-force method to check the existence of a solution for simple instances of this problem (e.g., one-level deductions that have at most 1000000 of possible linearisations) and the SMT-solver Z3 [16] to check more computationally complex ones, since both problems, $\mathrm{MIL}_{\texttt{then}}$ and $\mathrm{MIL}_{\texttt{dist}}$, are NP-complete [20]. There was a time limit of 10 minutes set for each combination of the transformation strategies. With this threshold only 1.92% and 0.03% remained untransformed in `then` and `dist` parameters optimisation, respectively.

Using a polynomial time algorithm that is sketched in Prop. 1, we reduced the number of labeled steps only in 4.49% of deductions. Additionally, these deductions were mainly located in proof scripts recently added to the MML database. This observation is a simple consequence of the fact that the `lab`–parameter in older scripts was reduced in a revision of MML database Version `4.121.1054` and obtained results were introduced to the Version `4.127.1060` [18]. Note that this situation was not intentional, since the main aim of this revision was not to minimise the `lab`–parameter, but generally to increase the `then`–parameter in proof scripts based on a greedy algorithm. However, topological sortings obtained by this algorithm fulfil the conditions (5).

	`then`–parameter	`dist`–parameter
Improved	1.19%	8.27%
Unchanged	76.36%	74.42%
Worsened	22.45%	17.31%

Table 1. Modification of `then` and `dist` parameters obtained by a polynomial time algorithm, sketched in Section 4, in comparison to the initial situation.

	`then`–parameter	`dist`–parameter
Improved	6.02%	18.66%
Unchanged	93.89%	80.89%
Worsened	0.09%	0.45%

Table 2. Modification of `then` and `dist` parameters obtained by a brute-force method or Z3 solver, if we restrict the search to the linearisation with optimal `lab`–parameter, in comparison to the initial situation. Clearly, we limited results to cases, in which at least one strategy solved the problem.

Analysing the impact of this polynomial time algorithm application for `then` and `lab` parameters, we observe that this parameters are more often worsened than improved. These results are summarised in Tab. 1. However, since we can determine efficiently the `lab`–parameter, we explored also the improvement of these parameters among such topological sortings that have optimal `lab`–parameter (see Tab. 2).

It is important to pay attention to the 0.45% percent of deductions for which it is certainly impossible to obtain the optimal value for both, `lab` and `dist` parameters simultaneously. More precisely, the analysis shows also that these 0.45% of deductions constitute only part (18.91%) of 2.37% cases where we cannot obtain the optimal `dist`-parameter if we have optimal `lab`-parameter. The other 79.36% and 1.73% of 2.37% cases are obtained for deductions where, despite optimal value of `lab`-parameter, we can improve or unchange respectively the `dist`-parameter in comparison to the initial situation.

As an illustration of such a conflict between `lab` and `dist` parameters, let us consider a simple abstract proof graph that is presented in Fig. 4. It is easy to see that this graph has exactly two possible linearisations σ: $1, 2, 3, 4$ and $2, 1, 3, 4$. Additionally, the number od σ-labeled steps is equal to 2 and 1 respectively, but the sum of all σ–distances of references is equal to 6 and 7, respectively.

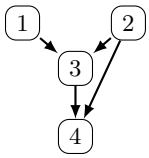


Fig. 4. A simple abstract proof graph for which the conflict between `lab` and `dist` parameters occurs.

Let us note also that the situation where we have to reduce `then`-parameter to obtain optimal `lab`-parameter is a rare situation that occurs mainly in a complex one-level deductions of MML. Additionally, existing examples of this conflict have more than one million of possible linearisations. However, to illustrate the conflict between `lab` and `dist` parameter, we can consider an artificial simple abstract proof graph, presented in Fig. 5 that can potentially occur in Mizar proof scripts. Based on the analysis carried out in Section 4 and Prop. 1 we infer that every topological sorting of this deduction has to have at last 5 labeled steps and this value is obtained, e.g., for an arrangement $1, 2, 4, 5, 7, 8, 3, 6, 9$. Indeed, vertices $1, 2, 4, 5$ have to be labelled, since there exist at least two tail endpoints references arc adjacent to these vertices. Additionally, at most one of the references to premises that correspond to vertices $6, 8$ can be realised without a label. The analysis of all possible, 42, topological sortings shows that the maximal value of `then`-parameter is equal to 6 and it is obtained in exactly two arrangements $1, 2, \ldots, 9$ and $1, 4, 7, 2, 5, 8, 3, 6, 9$ where there exist exactly 6 labeled steps. This shows that the conflict between `lab` and `then` parameters can occur even in "small" one-level deductions.

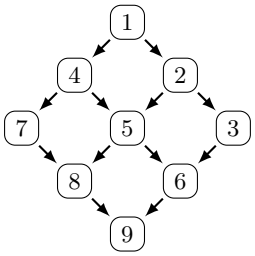


Fig. 5. A simple abstract proof graph for which the conflict between `lab` and `then` parameter occurs.

5 Conclusion

In this paper we have focused on reducing the number of labels introduced to formal reasoning in the process that improves legibility of proof scripts. We have showed that such a reduction can be NP-hard in a formal proof checking environment, even if it is computationally easy in another. Additionally, initial experiments with proof scripts occurring in the MML database show that optimisation of the labeled steps number can be in a conflict with other frequently employed methods of improving proof legibility. However, the presented research shows that this conflict occurs not so often and it appears mainly in deductions that have complex structures.

References

1. Bancerek, G., Hryniewicki, K.: Segments of Natural Numbers and Finite Sequences. Formalized Mathematics (1), 107–114 (1990)
2. Blanchette, J.C.: Redirecting Proofs by Contradiction. In: Third International Workshop on Proof Exchange for Theorem Proving, PxTP 2013. EPiC Series, vol. 14, pp. 11–26. EasyChair (2013)
3. Cowan, N.: Attention and Memory: An Integrated Framework. Oxford University Press (1998)
4. Cowan, N.: The magical number 4 in short-term memory: A reconsideration of mental storage capacity. Behavioral and Brain Sciences 24(1), 87–114 (2001)
5. Davis, M.: Obvious Logical Inferences. In: Proc. of the 7th International Joint Conference on Artificial Intelligence. pp. 530–531. William Kaufmann (1981)
6. Ericsson, K.A.: Analysis of memory performance in terms of memory skill, Advances in the psychology of human intelligence, vol. 4. Hillsdale, NJ: Lawrence Erlbaum Associates Ins. (1988)
7. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. A Series of Books in the Mathematical Science, W. H. Freeman and Company, New York (1979)
8. Gonthier, G.: Formal Proof—The Four-Color Theorem. Notices of the AMS 55(11), 1382–1393 (2008)
9. Grabowski, A., Korniłowicz, A., Naumowicz, A.: Mizar in a Nutshell. Journal of Formalized Reasoning 3(2), 153–245 (2010)

10. Grabowski, A., Schwarzweller, C.: Revisions as an Essential Tool to Maintain Mathematical Repositories. In: Proceedings of the 14th symposium on Towards Mechanized Mathematical Assistants: 6th International Conference, Lecture Notes in Computer Science, vol. 4573. pp. 235–249. Springer-Verlag (2007)
11. Grabowski, A., Schwarzweller, C.: Improving Representation of Knowledge within the Mizar Library. Studies in Logic, Grammar and Rhetoric 18(31), 35–50 (2009)
12. Grabowski, A., Schwarzweller, C.: On duplication in mathematical repositories. In: Autexier, S., Calmet, J.e.a. (eds.) Intelligent Computer Mathematics, 10th International Conference, AISC 2010, 17th Symposium, Calculemus 2010, and 9th International Conference, MKM 2010, Lecture Notes in Artificial Intelligence, vol. 6167, pp. 300–314. Springer-Verlag (2010)
13. Kaliszyk, C., Urban, J.: PRocH: Proof Reconstruction for HOL Light. In: Bonacina, M.P. (ed.) 24th International Conference on Automated Deduction, CADE-24. Lecture Notes in Computer Science, vol. 7898, pp. 267–274. Springer-Verlag (2013)
14. Matuszewski, R.: On Automatic Translation of Texts from Mizar-QC language into English. Studies in Logic, Grammar and Rhetoric 4 (1984)
15. Miller, G.A.: The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information. Psychological Review 63, 81–97 (1956)
16. de Moura, L., Bjørner, N.: Z3: An effcient SMT solver. In: Ramakrishnan, C.R., Rehof, J. (eds.) Tools and Algorithms for the Construction and Analysis of Systems, 14th International Conference, TACAS 2008. Lecture Notes in Computer Science, vol. 4963, pp. 337–340. Springer-Verlag (2008)
17. Naumowicz, A., Korniłowicz, A.: A Brief Overview of Mizar. In: TPHOLs'09, Lecture Notes in Computer Science, vol. 5674. pp. 67–72. Springer-Verlag (2009)
18. Pąk, K.: The Algorithms for Improving and Reorganizing Natural Deduction Proofs. Studies in Logic, Grammar and Rhetoric 22(35), 95–112 (2010)
19. Pąk, K.: Methods of Lemma Extraction in Natural Deduction Proofs. Journal of Automated Reasoning 50(2), 217–228 (2013)
20. Pąk, K.: The Algorithms for Improving Legibility of Natural Deduction Proofs. Ph.D. thesis, University of Warsaw (2013)
21. Rudnicki, P.: Obvious Inferences. Journal of Automated Reasoning 3(4), 383–393 (1987)
22. Smolka, S.J., Blanchette, J.C.: Robust, Semi-Intelligible Isabelle Proofs from ATP Proofs. In: Third International Workshop on Proof Exchange for Theorem Proving, PxTP 2013. EPiC Series, vol. 14, pp. 117–132. EasyChair (2013)
23. Trybulec, A., Korniłowicz, A., Naumowicz, A., Kuperberg, K.: Formal mathematics for mathematicians. Journal of Automated Reasoning 50(2), 119–121 (February 2013), http://dx.doi.org/10.1007/s10817-012-9268-z
24. Wenzel, M.: The Isabelle/Isar Reference Manual. University of Cambridge (2013), http://isabelle.in.tum.de/dist/Isabelle2013-2/doc/isar-ref.pdf
25. Wicknes, D.D., Born, D.G., Allen, C.K.: Proactive Inhibition and Item Similarity in Short-Term Memory. Journal of Verbal Learning and Verbal Behavior 2, 440–445 (1963)