LaTeX profiling of author submissions — completeness & usability checking

Apu V, Rishi T, Aravind Rajendran

Abstract

Due to the permissive nature of LaTeX, authors who prepare their manuscripts in LaTeX for publishing their research articles in academic journals often knowingly or unknowingly indulge in non-standard markup practices. Since article submission systems of most publishers are primarily designed for Microsoft Word-based articles, problems in LaTeX manuscripts go undetected during submission and review processes, and later cause avoidable delays and hardships in processing their submissions.

A tool for pre-submission check followed by requests to fix as much as possible at their end before submission will thus have benefits of earlier publication and reducing turnaround time considerably. TeXFolio Alpha is such a web-based tool for pre-submission profiling, completeness and usability checking of LaTeX manuscript submissions.

1 Introduction

LaTeX is not just a markup language for document preparation. The programmable nature of LaTeX allows authors to customize every aspect of the document. When typesetting documents for personal use, this flexibility of LaTeX is a great advantage. But in the case of academic publishing, journal publishers prefer author-submitted documents to follow a common template and style so that the article publication process, which involves reviewing, editing, typesetting, proofing and final online and print deliverables, can be done within the turnaround time, and as cost effectively as possible.

Journal publishers often provide authors document templates in the preferred layouts and style for both LaTeX and Microsoft Word. Word being a WYSIWYG document preparation system with limited customization capabilities compared to LaTeX, manuscript processing at the typesetters' end is easier for a non-math-intensive document prepared using Word templates. Processing of LaTeX manuscripts depends on many factors, such as the author's expertise and coding style in LaTeX, the journal submission system's compatibility with LaTeX, the journal typesetter's expertise in handling LaTeX manuscripts, etc. Novice LaTeX authors may ignore compilation errors or may not follow instructions given in LaTeX templates, while authors expert in LaTeX may use fancy or cutting-edge packages that may not work with the submission system or that break journal house style. This results in much communication between the author and typesetter which can cause avoidable delays and difficulties in processing authors' submission within a normal turnaround time. A pre-submission check tool that will ensure the requirements for fast article publishing are met would be a solution for these problems.

There are a number of reasons why a pre-submission profiling tool for LaTeX manuscripts is needed. First, it can help to ensure that the manuscript is properly formatted and meets all of the submission requirements by providing the facility to edit and compile the manuscript. Second, it can help to identify any potential problems with the manuscript before it is submitted by walking the authors through a checklist of the problems detected and providing instructions to fix them. Finally, it can help to save time and effort in the article production process by allowing the author to fix any problems before the submission process begins.

2 LaTeX submission profiling process

LaTeX profiling is the process of analyzing an author's submitted manuscript files for errors, missing files, use of recommended class files and packages, use of unsupported LaTeX packages, author definitions of macros, use of mandatory items in the submitted manuscript required by the publisher, and then generating a consolidated checklist based on the analysis. The author needs to review the checklist, which reports both the items that are safe to move forward and the items that need action from the author to fix. Thus the profiling process checks the completeness and usability (C&U) of the manuscript.

TeXFolio Alpha is a cloud LaTeX profiling tool which can be used as a web application and a microservice. Fig. 1 shows a high level block diagram of TeXFolio Alpha's workflow as a web application. Instead of directly uploading the manuscript files to the submission system, an author first uploads to TeXFolio Alpha. The C&U server on which TeXFolio Alpha is running processes the submitted files. This involves checking submitted LaTeX manuscripts and associated files using TeXFolio's analyzing scripts, written in Python and Perl, to detect the class files and packages used in the manuscript, missing input files and figures. If the author is not using the class file preferred by the publisher, TeXFolio Alpha will alert the author about the advantages of using that class file. If figures or input files are used in the manuscript but not uploaded to TeXFolio Alpha, they will be listed and the author prompted to upload them.

doi.org/10.47397/tb/43-2/tb134apu-submissions

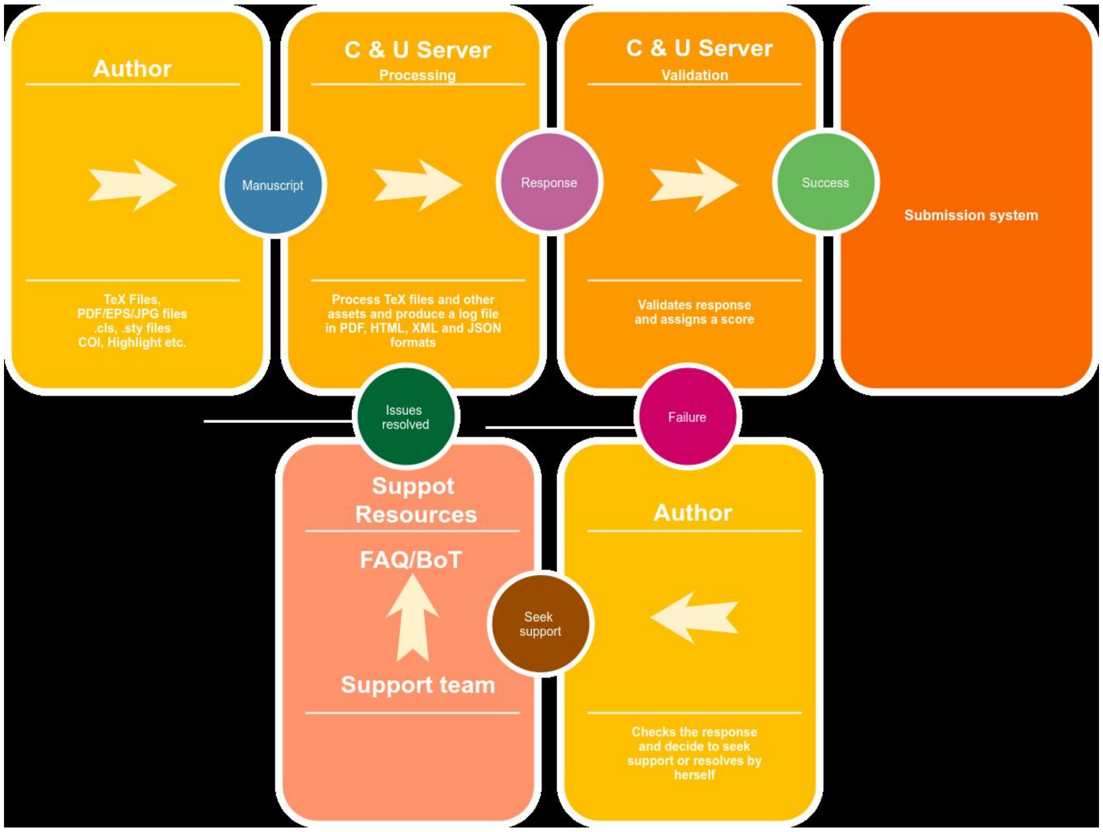
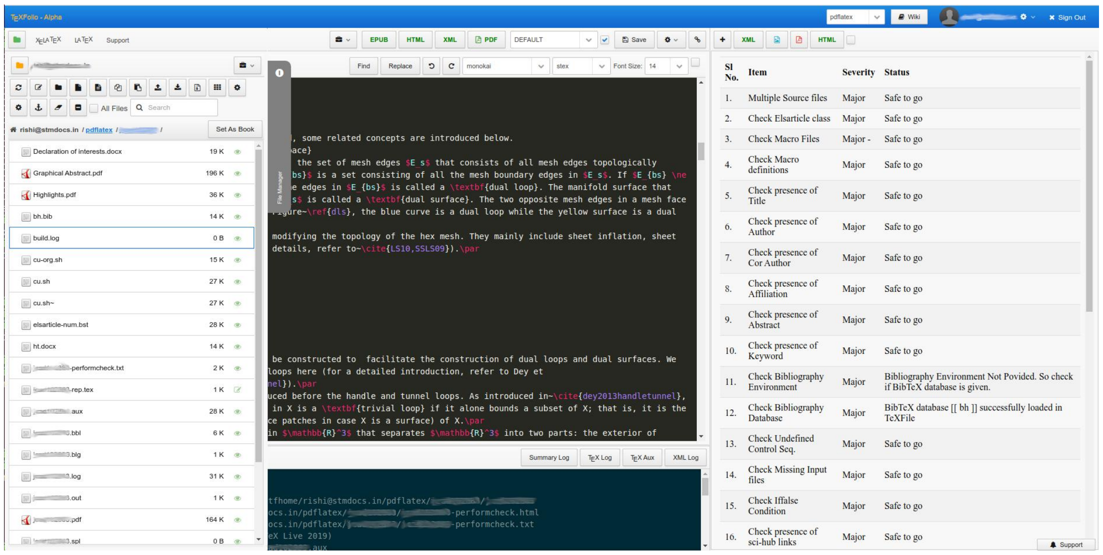

Author
C & U Server
Processing
C & U Server
Validation
Manuscript
Response
Success
Submission system
TeX Files, PDF/EPS/JPG files .cls, .sty files COI, Highlight etc.
Process TeX files and other assets and produce a log file in PDF, HTML, XML and JSON formats
Validates response and assigns a score
Issues resolved
Failure
Suppot Resources
FAQ/BoT
Support team
Seek support
Author
Checks the response and decide to seek support or resolves by herself

Figure 1: High level block diagram of TEXFolio Alpha.

Once the initial analysis and checking are done, the C&U server compiles the manuscript. If LATEX compilation errors are encountered, the author will be prompted with details of the errors and possible solutions to solve the errors. The author can edit the manuscript in TEXFolio Alpha's LATEX editor (Fig. 2). On the left side of the LATEX editor interface there is a file manager. Authors can view the list of files



rishi@stmdocs.in / pdflatex / [redacted] /

Declaration of interests.docx	19 K	
Graphical Abstract.pdf	196 K	
Highlights.pdf	36 K	
bh.bib	14 K	
build.log	0 B	
cu-org.sh	15 K	
cu.sh	27 K	
cu.sh~	27 K	
elsarticle-num.bst	28 K	
ht.docx	14 K	
[redacted]-performcheck.txt	2 K	
[redacted]-rep.tex	1 K	
[redacted].aux	28 K	
[redacted].bbl	6 K	
[redacted].blg	1 K	
[redacted].log	31 K	
[redacted].out	1 K	
[redacted].pdf	164 K	
[redacted].spl	0 B	

```
, some related concepts are introduced below.
ace}
 the set of mesh edges $E_s$ that consists of all mesh edges topologically
bs}$ is a set consisting of all the mesh boundary edges in $E_s$. If $E_{bs} \ne
e edges in $E_{bs}$ is called a \textbf{dual loop}. The manifold surface that
s$ is called a \textbf{dual surface}. The two opposite mesh edges in a mesh face
igure~\ref{dls}, the blue curve is a dual loop while the yellow surface is a dual

modifying the topology of the hex mesh. They mainly include sheet inflation, sheet
details, refer to~\cite{LS10,SSLS09}).\par




 be constructed to  facilitate the construction of dual loops and dual surfaces. We
loops here (for a detailed introduction, refer to Dey et
nel}).\par
uced before the handle and tunnel loops. As introduced in~\cite{dey2013handletunnel},
 in X is a \textbf{trivial loop} if it alone bounds a subset of X; that is, it is the
ce patches in case X is a surface) of X.\par
in $\mathbb{R}^3$ that separates $\mathbb{R}^3$ into two parts: the exterior of
```

```
tfhome/rishi@stmdocs.in/pdflatex/[redacted]/[redacted]
ocs.in/pdflatex/[redacted]/[redacted]-performcheck.html
ocs.in/pdflatex/[redacted]/[redacted]-performcheck.txt
eX Live 2019)
[redacted].aux
```

Sl No.	Item	Severity	Status
1.	Multiple Source files	Major	Safe to go
2.	Check Elsarticle class	Major	Safe to go
3.	Check Macro Files	Major -	Safe to go
4.	Check Macro definitions	Major	Safe to go
5.	Check presence of Title	Major	Safe to go
6.	Check presence of Author	Major	Safe to go
7.	Check presence of Cor Author	Major	Safe to go
8.	Check presence of Affiliation	Major	Safe to go
9.	Check presence of Abstract	Major	Safe to go
10.	Check presence of Keyword	Major	Safe to go
11.	Check Bibliography Environment	Major	Bibliography Environment Not Povided. So check if BibTeX database is given.
12.	Check Bibliography Database	Major	BibTeX database [[bh]] successfully loaded in TeXFile
13.	Check Undefined Control Seq.	Major	Safe to go
14.	Check Missing Input files	Major	Safe to go
15.	Check Iffalse Condition	Major	Safe to go
16.	Check presence of sci-hub links	Major	Safe to go

Figure 2: TEXFolio Alpha's LATEX editor.

Table 1: List of checks performed by TEXFolio Alpha while profiling LATEX manuscripts.

No.	Items	Severity
1	Multiple source files	Major*
2	Check usage of Elsarticle class	Major
3	Check missing Macro packages	Major
4	Check missing Macro definitions	Major
5	Check presence of Title	Major
6	Check presence of Author	Major
7	Check presence of Corresponding Author	Major
8	Check presence of Affiliation	Major
9	Check presence of Abstract	Minor†
10	Check presence of Keyword	Minor
11	Check Bibliography Environment	Minor
12	Check Bibliography Database	Minor
13	Check Undefined References	Minor
14	Check Undefined Control Seq.	Major
15	Check Multiply Def. Labels	Minor
16	Check Missing Input files	Major
17	Check Iffalse Condition	Minor
18	Check Nomenclature	Minor
19	Check Appendix/Supplementary materials	Minor
20	Check possible Overfull content	Minor
21	Check private email address(es)	Minor
22	Check Bibliographic citations — Crossref	Minor
23	Check other cross-references	Minor
24	Check presence of Highlights	Minor
25	Check presence of Conflict of Interest	Minor
26	Check presence of sci-hub links	Major
27	Check presence of notes	Minor
28	Check non-standard Math coding	Minor
29	Check missing Bibliographic citation	Minor
30	Check missing other cross-references	Minor

* Major: Cannot proceed with submission. Author must correct the issue.
† Minor: Can proceed with submission. Typesetter will take care to correct the issue.

submitted and upload, and delete, rename or copy files. TEX files can be opened in the editor in the middle of the interface and changes can be made. Below the editor, the log window will show compilation errors and prompt authors with instructions to fix the errors. On the right-hand side, the document viewer will display the pdf document generated from the manuscript.

Once the author corrects all compilation errors, the C&U server will start validating the manuscript. The document viewer will display the checklist generated by TEXFolio Alpha after the validation process. This checklist is generated by several operations, such as processing the log and auxiliary files using Python and Perl scripts, and a checklist in XML format that is generated during LATEX compilation. This checklist can be seen in the document viewer in Fig. 2. A few examples and more details about this XML checklist generation are given in Section 3. Items that are already satisfied by the manuscript will be listed with status 'Safe to Go'. Items that need action by the author will be listed with details of the problem. Table 1 lists the current checks performed by TEXFolio Alpha.

TEXFolio Alpha will also assign a score to the manuscript relating on the status of the items and its severity. If the score does not meet the threshold set by the publisher, the submission will not be moved to the publishers' submission system; instead, the author will be asked to correct the critical problems.

To help the author solve problems, FAQs, chatbot or human support can be integrated into the TEXFolio interface. Since the issues and instructions to solve them are presented in a checklist format it will often be easier for the authors to correct the problems themselves. After the author completes the validation process with a sufficient validation score, the manuscript will be moved to the publisher's submission system and a preprint pdf will be generated.

TEXFolio Alpha can be configured as a microservice as well, which can be integrated with other applications and used with API calls.

3 Role of LATEX in the manuscript validation process

As discussed above, TEXFolio Alpha uses Python and Perl for processing and analyzing LATEX source files, logs and auxiliary files to generate the C&U checklist. Details like missing input files, checking for the use of recommended class files and packages, uncited references etc., can be detected using these methods. But a few critical ingredients for the checklist can only be extracted with the help of LATEX during a compilation. TEXFolio Alpha uses LATEX's hook mechanisms and the `etoolbox` package's patching commands to extract checklist details during compilation. All hooks and patching commands are kept in a configuration file and this file is loaded in the manuscript before `\documentclass` using `\input`. The example in this article uses a manuscript using the `elsarticle.cls`; this is a popular document class for preparing preprint pdfs and it is the recommended document class by one of the largest academic publishers.

In this example, `elsarticle-pre-hooks.tex` contains the hook macros and patching commands. This pre-hooks file contains publisher and journal specific configuration for documents generated using `elsarticle.cls`:

```
\input{elsarticle-pre-hooks}
\documentclass[final]{elsarticle}
```

This is the only change in the manuscript made for profiling in TEXFolio Alpha. By using hooks and patching there is no need to add any additional packages or macro definition in the preamble part of author manuscript for profiling. This makes sure the author's manuscript is kept intact during TEXFolio Alpha's profiling process.

We will next discuss three examples where we used LATEX hook mechanism in TEXFolio Alpha.

3.1 Detection of sensitive macro redefinitions

An author may redefine macros, either intentionally or unintentionally, that are defined in the class file. Although LATEX normally reports 'already defined' errors, the author may use `\def` or `\renewcommand` to skip this error without considering the reasons for the error. In some cases a package loaded by the author may redefine macros without showing any errors.

We use a hook in the configuration which check for definitions by the author or by loaded packages that override already-defined macros in the class file, per publisher requirements. If such redefinitions go undetected during submission they will raise style error flags later, during journal production at the typesetter, causing delays in the article production as the typesetter must contact the journal and authors to query whether to keep the custom style. If such cases are detected at the submission stage, the author can explain the rationale behind changing the macro, or revert to the original macro defined in the class if the redefinition was unintentional and does not have any particular significance.

Let's look at the implementation of the hook. We create a `clist` (comma-separated list) that holds the names of sensitive macros. In this example, two commands `\textmarker` and `\author` are the sensitive macros.

```
\clist_new:N \cu_macros_for_verification
\clist_gset:Nn \cu_macros_for_verification
    {textmarker,author}
```

Two property lists are created to hold a hash value of the macro definitions.

```
\prop_new:N \cu_elsarticle_macro_hash
\prop_new:N \cu_document_macro_hash
```

This hash value will be used to verify if the macro definition has been modified.

Using the hook mechanism, we add macro calls after the class file is loaded which compute the hash value of the meaning of the macros listed in `\cu_macros_for_verification` to the property list `\cu_elsarticle_macro_hash`.

```
\AddToHook{class/after}[elsarticle]{
 \clist_map_inline:Nn
   \cu_macros_for_verification
 { \str_if_eq:nnTF { #1 } { }
    { \clist_map_break: }
    { \prop_gput:Nnx
      \cu_elsarticle_macro_hash { #1 }
      { \tex_mdfivesum:D{\csmeaning{#1}} }
    }
 }
}
```

Next, to capture any cases of redefinition of the macros listed in \g_macros_for_verification we add an \enddocument hook to save hash values for the macro definitions to \g_document_macro_hash.

```
\AddToHook{enddocument}{
 \clist_map_inline:Nn
   \cu_macros_for_verification
 { \str_if_eq:nnTF { #1 } { }
    { \clist_map_break: }
    { \prop_gput:Nnx
        \cu_document_macro_hash { #1 }
        { \tex_mdfivesum:D{\csmeaning{#1}}
        }
    }
 }
```

Continuing in the same hook, we compare the hash values on both property lists. If hash values are different, authors need to verify if the macro redefinition is necessary.

```
 \prop_map_inline:Nn
  \cu_elsarticle_macro_hash {
    \str_if_eq:nnTF { #1 } { }
     { \prop_map_break: }
     { \str_if_eq:eeTF { #2 }
      { \prop_item:Nn
         \cu_document_macro_hash { #1 } }
      { }
```

When we identify such a difference in hash value, we add a checklist item with description of the problem to a custom XML file.

```
     { \iow_now:Nx \cu_report
       {<checklist
         type="redefined-macros">}
      \iow_now:Nx \cu_report
       {<description id="redefined-#1">
         \cumacrodescription{#1}
        </description>}
      \iow_now:Nx \cu_report {</checklist>}
     }
   }
 }
}
```

The XML elements are given attributes @type and @id to simplify generation of a report which can be viewed in the TEXFolio Alpha web interface.

3.2 Check for mandatory items in the manuscript

Here we look at another case, this time checking for the use of mandatory items in the document. When using the class file suggested by the publisher, some commands defined in the class file, for example, macros for keywords, corresponding author, etc., will be mandatory for submission. One way to detect if authors have not omitted these commands is to add error messages if they are not used. But in the case of popular class files such as elsarticle.cls, authors use it for typesetting documents for personal use like lecture notes too. The implementation of errors specific for publisher requirements will create difficulties in those type of non-academic use cases. So using a hook configuration in the profiling tool is a better solution.

To ensure whether authors use these mandatory commands in manuscripts, a list of such macros are maintained:

```
\clist_new:N \cu_mandatory_macros
\clist_set:Nn \cu_mandatory_macros
  {corref,cortext}
```

\corref and \cortext are commands used to tag corresponding authors in elsarticle.cls.

We iterate through the list, adding a hook to each macro itself which sets a definition to indicate the macro has been used.

```
\clist_map_inline:Nn
 \cu_mandatory_macros
  {\str_if_eq:nnTF { #1 } { }
   { \clist_map_break: }
   { \AddToHook{cmd/#1/before}
      { \csgdef{cu_macro_#1_done}{1}} }
  }
```

Then a hook at \enddocument verifies if the command has been used, and writes to the checklist XML.

```
\AddToHook{enddocument}{
 \clist_map_inline:Nn
   \cu_mandatory_macros {
     \str_if_eq:nnTF { #1 } { }
      { \clist_map_break: }
      { \ifcsundef { cu_macro_#1_done }
       { \iow_now:Nx \cu_report
         {<checklist
            type="mandatory-macros">}
         \iow_now:Nx \cu_report
         {<description
           id="mandatory-macro-#1">
           \cumandatorymacrodescription{#1}
          </description>}
         \iow_now:Nx \cu_report
           {</checklist>}
       } { }
     }
  }
}
```

An analogous hook can be used to check for use of mandatory environments: instead of hooking to `cmd/`⟨*command name*⟩`/before`, we hook to `env/`⟨*environment name*⟩`/before`.

3.3 Preventing the use of incompatible or troublesome packages

Packages authors load in their manuscript may be incompatible with the LaTeX submission system of the publisher. For instance, packages that require the `--shell-escape` option will not be allowed with these online submission systems, as they poses a security risk. A commonly-used example of such a package is `minted.sty`.

Via the `package/`⟨*package name*⟩`/before` hook we can stop the compilation at the point where the package is loaded and instruct author to use a similar package that is compatible with the submission system, or suggest some alternate methods. A hook as below can be used to stop compilation at the point where `minted` is detected.

```
\AddToHook{package/minted/before}{
 \AddToHook{enddocument}{
   \iow_now:Nx \cu_report
     {<checklist
       type="problem-package">}
   \iow_now:Nx \cu_report
     {<description
       id="problem-pkg-minted">
       \cuproblempkgdescription{minted}
      </description>}
   \iow_now:Nx \cu_report
      {</checklist>}
  }
  \AddToHook{begindocument/end}
    {\enddocument}
  \endinput
}
```

4 Objectives and scope of TeXFolio Alpha

The main objective of TeXFolio Alpha is to ensure that authors submit the latest, single version source material right the first time. The reports generated by the checklists can be used by the publisher and supplier workflow management systems to improve customer experience. FAQs and chatbots can be integrated into the system to walk authors through the process. Being a cloud-based web service or micro-service, authors and publishers can use TeXFolio Alpha without the need of a local TeX installation.

References

[1] LaTeX's hook management. `https://mirror.ctan.org/macros/latex/base/lthooks-doc.pdf`

[2] The LaTeX3 Sources. `https://mirror.ctan.org/macros/latex/contrib/l3kernel/source3.pdf`

◇ Apu V
STM Document Engineering
Trivandrum, Kerala, India
apu.v (at) stmdocs.in
https://stmdocs.in

◇ Rishi T
STM Document Engineering
Trivandrum, Kerala, India
rishi (at) stmdocs.in
https://stmdocs.in

◇ Aravind Rajendran
Independent Consultant,
STM Document Engineering
Trivandrum, Kerala, India
aravind (at) stmdocs.in
https://stmdocs.in