

# Spark-BDD : A Big Data Debugger

Muhammad Ali Gulzar  
UCLA

Tianyi Zhang  
UCLA

Seunghyun Yoo  
UCLA

## ABSTRACT

Apache Spark has become a key platform for Big Data Analytics, yet it lacks complete support for debugging analytics programs. As a result, the development of a new analytical toolkit can be a painstakingly long process. To fill this gap, we developed Spark-BDD (Big Data Debugger), which brings a traditional interactive debugger experience to the Spark platform. Over the course of this project we integrated current version of Spark-BDD with some interactive fault localization and optimization mechanisms. Analytic programmers (e.g., data scientists) can leverage Spark-BDD interactive debugging capabilities to set breakpoints and watchpoints, investigate crashes and failures, identifying straggling data partitions. Along with debugging this tool provides some useful runtime information that can be helpful in improving the performance of spark in an unpredictable cluster deployment. One of these optimizations is inter-stage repartitioning to avoid data skew during data shuffling where performance improvement is reflected in the spark job completion time. Most of these debugging features pose minimum overhead and in turn provide valuable debugging with lowest number of iterations.

## Keywords

Debugging, Apache Spark, Fault Recovery & Localization, Performance

## 1. INTRODUCTION

Large scale data processing is the hottest topic these days. As the accessibility of the technology and methods to collect is improved, the amount of data is also increasing exponentially. Processing these massive amount of datasets are both difficult and necessary. As scene from the experience single computer super machines are capable enough to analyze that scale of data. To overcome that cluster computing frameworks are the right path to perform these kind of analysis. Large companies like Google and Facebook have their own data center which they use to process their data with small companies lease subset of inter connected machines from third party resource provider like amazon. Since the availability of these kind infrastructure is not an issue anymore, the need for software framework that be ran on it is imminent. Apache Spark is one of these large scale data processing framework that can be run cluster computing frameworks with thousands of nodes. Spark is built on MapReduce programming paradigm from LISP but also provides data processing operation like GroupBy and Joins. Several other frameworks before Sparks also encountered the

same problem in the same distributed way but most of them were either slow (Hadoop) or complicated (Dryad). Spark cater both of these issues by providing an in-memory model for lightning fast speed and simple Scala, java and python API for easier understanding. The choice of Spark for this project is because of its growing popularity in the community, impressive performance and research support as a lot of research lab are working in or on top of spark. In this project we present a debugging tool called SparkBDD build on top of Spark to debugging feature that are usually available to user in single process programs. The basic aim was to develop a tool corresponds to the features available to users on conventional debuggers as the users of Spark are mostly ML and Data Science that are not very familiar with the spark internals. By hiding the internal details, SparkBDD introduces a layer of debugging build in the same API and a web based UI to perform debugging both physical and data layer. We support features like breakpoint, Watchpoint, stragglers profiling, crash culprit identification and recovery. All of these features supports wide array of use cases and drives a new optimizations and performance improvements from the measurements collected during debugging runtime. Later in the report we also show the overhead of this frameworks and one of the performance improvements based on the stragglers profiling information.

## 2. MOTIVATION

Typically, the body of a paper is organized into a hierarchical structure, with numbered or unnumbered headings for sections, subsections, sub-subsections, and even smaller sections. The command `\section` that precedes this paragraph is part of such a hierarchy.<sup>1</sup>  $\LaTeX$  handles the numbering and placement of these headings for you, when you use the appropriate heading commands around the titles of the headings. If you want a sub-subsection or smaller part to be unnumbered in your output, simply append an asterisk to the command name. Examples of both numbered and unnumbered headings will appear throughout the balance of this sample document.

Because the entire article is contained in the `document` environment, you can indicate the start of a new paragraph with a blank line in your input file; that is why this sentence forms a separate paragraph.

<sup>1</sup>This is the second footnote. It starts a series of three footnotes that add nothing informational, but just give an idea of how footnotes work and look. It is a wordy one, just so you see how a longish one plays out.

## 2.1 Type Changes and *Special Characters*

We have already seen several typeface changes in this sample. You can indicate italicized words or phrases in your text with the command `\textit`; emboldening with the command `\textbf` and typewriter-style (for instance, for computer code) with `\texttt`. But remember, you do not have to indicate typestyle changes when such changes are part of the *structural* elements of your article; for instance, the heading of this subsection will be in a sans serif<sup>2</sup> typeface, but that is handled by the document class file. Take care with the use of<sup>3</sup> the curly braces in typeface changes; they mark the beginning and end of the text that is to be in the different typeface.

You can use whatever symbols, accented characters, or non-English characters you need anywhere in your document; you can find a complete list of what is available in the *L<sup>A</sup>T<sub>E</sub>X User's Guide*[?].

## 2.2 Math Equations

You may want to display math equations in three distinct styles: inline, numbered or non-numbered display. Each of the three are discussed in the next sections.

### 2.2.1 Inline (In-text) Equations

A formula that appears in the running text is called an inline or in-text formula. It is produced by the **math** environment, which can be invoked with the usual `\begin. . . \end` construction or with the short form `$. . . $`. You can use any of the symbols and structures, from  $\alpha$  to  $\omega$ , available in L<sup>A</sup>T<sub>E</sub>X[?]; this section will simply show a few examples of in-text equations in context. Notice how this equation:  $\lim_{n \rightarrow \infty} x = 0$ , set here in in-line math style, looks slightly different when set in display style. (See next section).

### 2.2.2 Display Equations

A numbered display equation – one set off by vertical space from the text and centered horizontally – is produced by the **equation** environment. An unnumbered display equation is produced by the **displaymath** environment.

Again, in either environment, you can use any of the symbols and structures available in L<sup>A</sup>T<sub>E</sub>X; this section will just give a couple of examples of display equations in context. First, consider the equation, shown as an inline equation above:

$$\lim_{n \rightarrow \infty} x = 0 \quad (1)$$

Notice how it is formatted somewhat differently in the **displaymath** environment. Now, we'll enter an unnumbered equation:

$$\sum_{i=0}^{\infty} x + 1$$

and follow it with another numbered equation:

$$\sum_{i=0}^{\infty} x_i = \int_0^{\pi+2} f \quad (2)$$

just to demonstrate L<sup>A</sup>T<sub>E</sub>X's able handling of numbering.

<sup>2</sup>A third footnote, here. Let's make this a rather short one to see how it looks.

<sup>3</sup>A fourth, and last, footnote.

Table 1: Frequency of Special Characters

Non-English or Math	Frequency	Comments
Ø	1 in 1,000	For Swedish names
π	1 in 5	Common in math
\$	4 in 5	Used in business
Ψ <sub>1</sub> <sup>2</sup>	1 in 40,000	Unexplained usage

## 2.3 Citations

Citations to articles [?, ?, ?, ?], conference proceedings [?] or books [?, ?] listed in the Bibliography section of your article will occur throughout the text of your article. You should use BibTeX to automatically produce this bibliography; you simply need to insert one of several citation commands with a key of the item cited in the proper location in the **.tex** file [?]. The key is a short reference you invent to uniquely identify each work; in this sample document, the key is the first author's surname and a word from the title. This identifying key is included with each item in the **.bib** file for your article.

The details of the construction of the **.bib** file are beyond the scope of this sample document, but more information can be found in the *Author's Guide*, and exhaustive details in the *L<sup>A</sup>T<sub>E</sub>X User's Guide*[?].

This article shows only the plainest form of the citation command, using `\cite`. This is what is stipulated in the SIGS style specifications. No other citation format is endorsed.

## 2.4 Tables

Because tables cannot be split across pages, the best placement for them is typically the top of the page nearest their initial cite. To ensure this proper “floating” placement of tables, use the environment **table** to enclose the table's contents and the table caption. The contents of the table itself must go in the **tabular** environment, to be aligned properly in rows and columns, with the desired horizontal and vertical rules. Again, detailed instructions on **tabular** material is found in the *L<sup>A</sup>T<sub>E</sub>X User's Guide*.

Immediately following this sentence is the point at which Table 1 is included in the input file; compare the placement of the table here with the table in the printed dvi output of this document.

To set a wider table, which takes up the whole width of the page's live area, use the environment **table\*** to enclose the table's contents and the table caption. As with a single-column table, this wide table will “float” to a location deemed more desirable. Immediately following this sentence is the point at which Table 2 is included in the input file; again, it is instructive to compare the placement of the table here with the table in the printed dvi output of this document.

## 2.5 Figures

Like tables, figures cannot be split across pages; the best placement for them is typically the top or the bottom of the page nearest their initial cite. To ensure this proper “floating” placement of figures, use the environment **figure** to enclose the figure and its caption.

Table 2: Some Typical Commands

Command	A Number	Comments
<code>\alignauthor</code>	100	Author alignment
<code>\numberofauthors</code>	200	Author enumeration
<code>\table</code>	300	For tables
<code>\table*</code>	400	For wider tables



Figure 1: A sample black and white graphic (.eps format).



Figure 2: A sample black and white graphic (.eps format) that has been resized with the epsfig command.

This sample document contains examples of .eps and .ps files to be displayable with L<sup>A</sup>T<sub>E</sub>X. More details on each of these is found in the *Author's Guide*.

As was the case with tables, you may want a figure that spans two columns. To do this, and still to ensure proper “floating” placement of tables, use the environment **figure\*** to enclose the figure and its caption.

Note that either .ps or .eps formats are used; use the `\epsfig` or `\psfig` commands as appropriate for the different file types.

## 2.6 Theorem-like Constructs

Other common constructs that may occur in your article are the forms for logical constructs like theorems, axioms, corollaries and proofs. There are two forms, one produced by the command `\newtheorem` and the other by the command `\newdef`; perhaps the clearest and easiest way to distinguish them is to compare the two in the output of this sample document:

This uses the **theorem** environment, created by the `\newtheorem` command:

**THEOREM 1.** *Let  $f$  be continuous on  $[a, b]$ . If  $G$  is an antiderivative for  $f$  on  $[a, b]$ , then*

$$\int_a^b f(t)dt = G(b) - G(a).$$

The other uses the **definition** environment, created by the `\newdef` command:

Figure 3: A sample black and white graphic (.ps format) that has been resized with the psfig command.

*Definition 1.* If  $z$  is irrational, then by  $e^z$  we mean the unique number which has logarithm  $z$ :

$$\log e^z = z$$

Two lists of constructs that use one of these forms is given in the *Author's Guidelines*.

and don't forget to end the environment with `figure*`, not `figure`!

There is one other similar construct environment, which is already set up for you; i.e. you must *not* use a `\newdef` command to create it: the **proof** environment. Here is an example of its use:

**PROOF.** Suppose on the contrary there exists a real number  $L$  such that

$$\lim_{x \rightarrow \infty} \frac{f(x)}{g(x)} = L.$$

Then

$$l = \lim_{x \rightarrow c} f(x) = \lim_{x \rightarrow c} \left[ g(x) \cdot \frac{f(x)}{g(x)} \right] = \lim_{x \rightarrow c} g(x) \cdot \lim_{x \rightarrow c} \frac{f(x)}{g(x)} = 0 \cdot L = 0,$$

which contradicts our assumption that  $l \neq 0$ .  $\square$

Complete rules about using these environments and using the two different creation commands are in the *Author's Guide*; please consult it for more detailed instructions. If you need to use another construct, not listed therein, which you want to have the same formatting as the Theorem or the Definition[?] shown above, use the `\newtheorem` or the `\newdef` command, respectively, to create it.

## A Caveat for the T<sub>E</sub>X Expert

Because you have just been given permission to use the `\newdef` command to create a new form, you might think you can use T<sub>E</sub>X's `\def` to create a new command: *Please refrain from doing this!* Remember that your L<sup>A</sup>T<sub>E</sub>X source code is primarily intended to create camera-ready copy, but may be converted to other forms – e.g. HTML. If you inadvertently omit some or all of the `\defs` recompilation will be, to say the least, problematic.

## 3. CONCLUSIONS

This paragraph will end the body of this sample document. Remember that you might still have Acknowledgments or

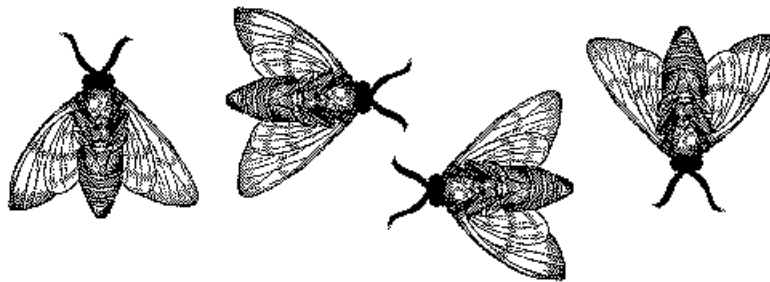


Figure 4: A sample black and white graphic (.eps format) that needs to span two columns of text.

Appendices; brief samples of these follow. There is still the Bibliography to deal with; and we will make a disclaimer about that here: with the exception of the reference to the L<sup>A</sup>T<sub>E</sub>X book, the citations in this paper are to articles which have nothing to do with the present subject and are used as examples only.

## 4. ACKNOWLEDGMENTS

This section is optional; it is a location for you to acknowledge grants, funding, editing assistance and what have you. In the present case, for example, the authors would like to thank Gerald Murray of ACM for his help in codifying this *Author's Guide* and the .cls and .tex files that it describes.

## APPENDIX

### A. HEADINGS IN APPENDICES

The rules about hierarchical headings discussed above for the body of the article are different in the appendices. In the `appendix` environment, the command `section` is used to indicate the start of each Appendix, with alphabetic order designation (i.e. the first is A, the second B, etc.) and a title (if you include one). So, if you need hierarchical structure *within* an Appendix, start with `subsection` as the highest level. Here is an outline of the body of this document in Appendix-appropriate form:

#### A.1 Introduction

#### A.2 The Body of the Paper

##### A.2.1 Type Changes and Special Characters

##### A.2.2 Math Equations

##### *Inline (In-text) Equations*

##### *Display Equations*

##### A.2.3 Citations

##### A.2.4 Tables

##### A.2.5 Figures

##### A.2.6 Theorem-like Constructs

##### *A Caveat for the T<sub>E</sub>X Expert*

#### A.3 Conclusions

#### A.4 Acknowledgments

### A.5 Additional Authors

This section is inserted by L<sup>A</sup>T<sub>E</sub>X; you do not insert it. You just add the names and information in the `\additionalauthors` command at the start of the document.

### A.6 References

Generated by bibtex from your .bib file. Run latex, then bibtex, then latex twice (to resolve references) to create the .bbl file. Insert that .bbl file into the .tex source file and comment out the command `\thebibliography`.

## B. MORE HELP FOR THE HARDY

The acm\_proc\_article-sp document class file itself is chock-full of succinct and helpful comments. If you consider yourself a moderately experienced to expert user of L<sup>A</sup>T<sub>E</sub>X, you may find reading it useful but please remember not to change it.