Computer-Aided Survey Writing - Paper in a Day

R.J. Ruigrok
TU Delft, Dept. Computer Science
Julianalaan 132
2624 BL Delft
The Netherlands

Abstract

We present a semiautomated scientific survey writing tool. It automates as many tasks as possible for survey papers. For instance it can auto-generate an overview of recent work in an area. Any scientist should be able to write a genuine survey with a mere 24 hours of effort. All mundane tasks are automated and quick. Related publications are collected by parsing the references of an initial seed paper. Our tool can also dissect articles in .PDF format using semantic decomposition by recognizing title, abstract and citation sections. A large set of related publications is gathered seamlessly by recursively following citations from the seed and downloading them from the web. This collection can then be adjusted and used to generate an initial survey in LaTeX. This survey includes references plus abstracts of the selection of referenced papers and generates a bibliography file. As a final step this initial survey is polished using manually text editing and a camera-ready survey is generated.

Keywords: semiautomated survey writing, citation parsing, semantic decomposition

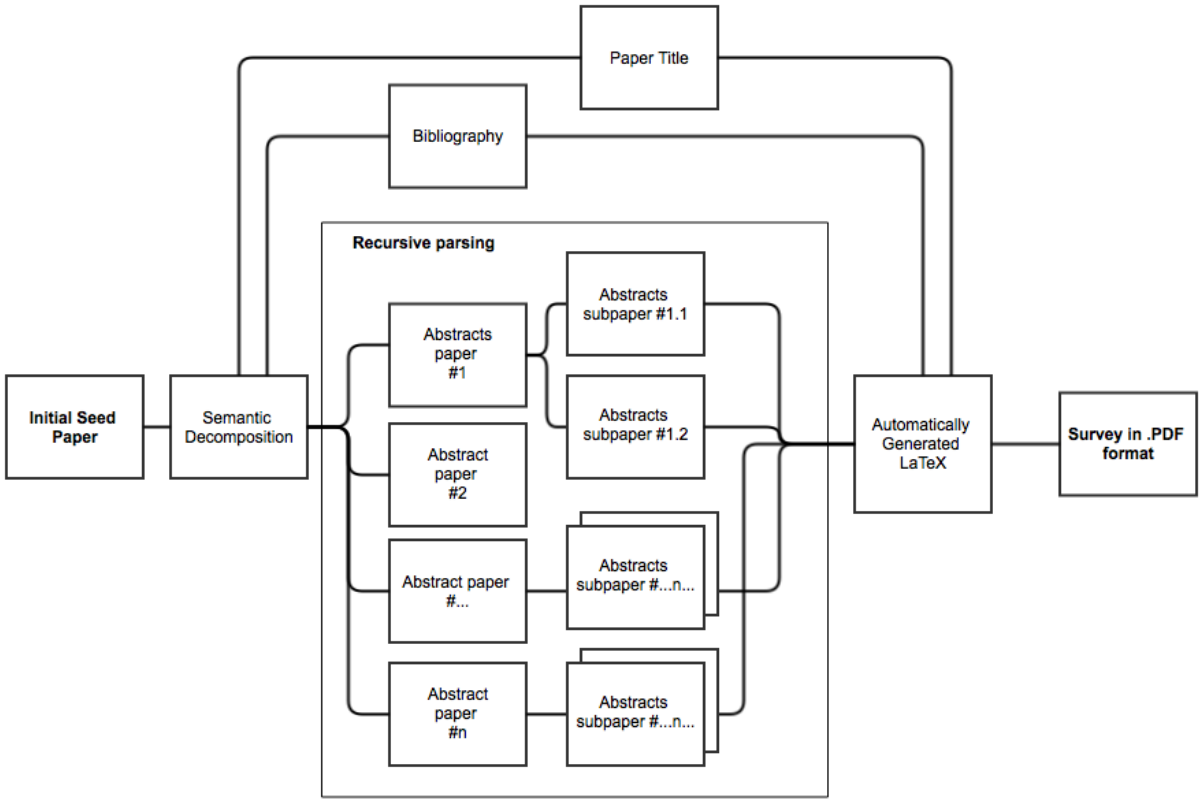


Diagram 1: A quick overview of the core modules for generating surveys

1 Related work

1.1 Random text by Markov chain

Initially, we experimented with Markov Chains to generate random titles for papers. As a learning set, we fed the concatenated list of paper titles that would be included in the survey. The results are nice, sometimes good-looking, but not always. Markov chains do not have a clue about the ending of a sentence. E.g. a title ending with the keyword 'to' or 'and' or containing punctuation marks with just a word behind it are also generated and render the use of Markov not useful for this purpose. A fun fact about this is that markov chains are useful to creatively form names for new features. Sometimes Markov returns a title that is rude or not ethical, like 'Performing interaction on abusing woman'.

1.2 Genuine vs. Fake

In 2005, three graduate students at MIT developed **SCIgen**, an automatic paper generator in the field of Computer Science. It uses a context-free grammar to form the contents of the paper, and make it look genuine. The main purpose of SCIgen was to auto-generate submissions to conferences to check whether the program committee will accept it. Some papers got accepted, which proves that some conferences can't be taken seriously.

The approach for this project is not the same. The purpose of this project is to prefabricate a scientific survey based on an initial paper. The application will look for related work and references. The scientist will choose which of the related work should be included in the survey, and is able to change any content on-the-fly in LaTeX and BibTex. The application will append the abstracts of the chosen papers to the survey with the corresponding references. The scientist does not need to look for any publications himself, but just needs to check which work presented by the tool should be included. This is different from the MIT work, as these papers are generated with garbage in it: readable, but not making any sense.

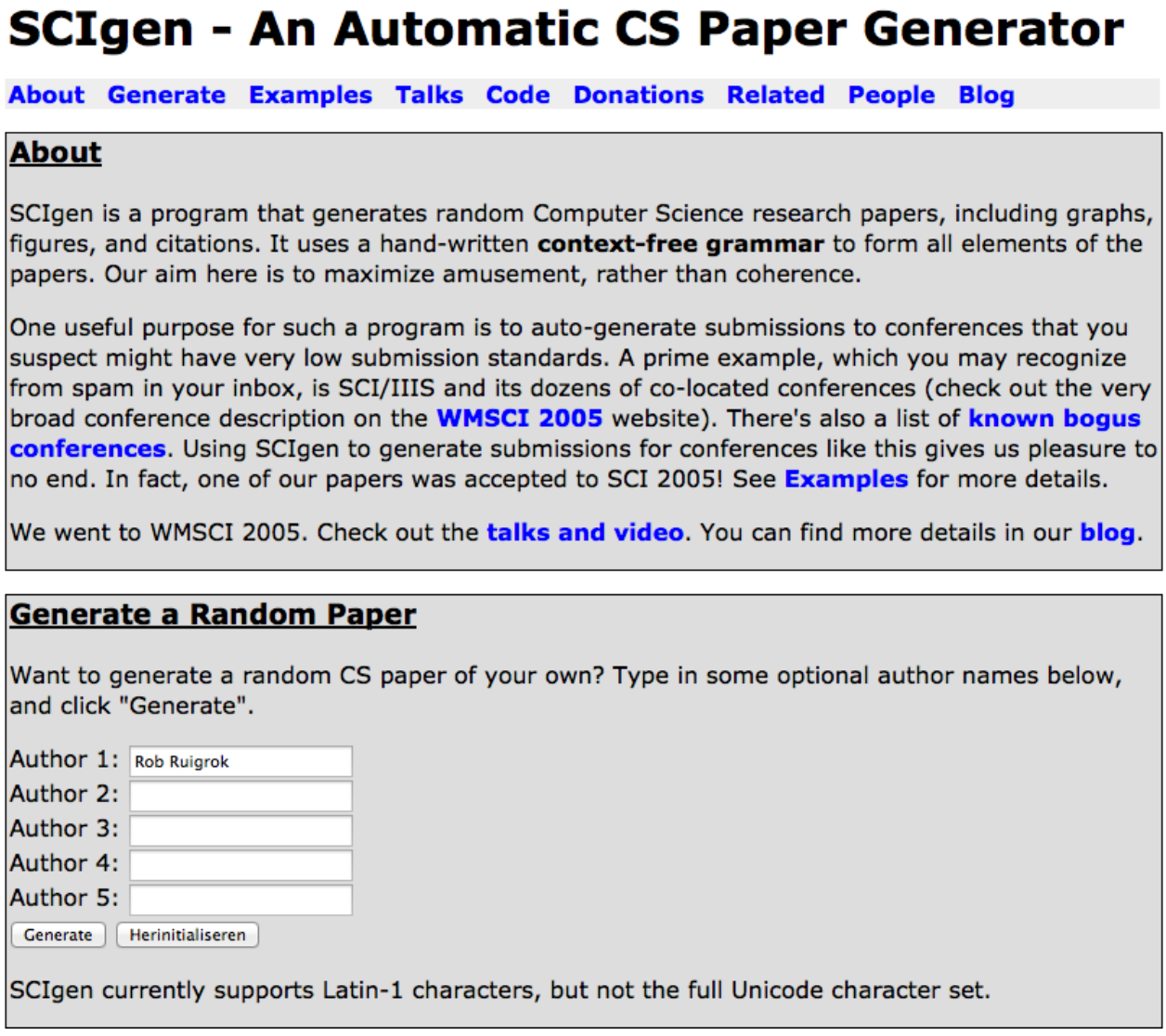
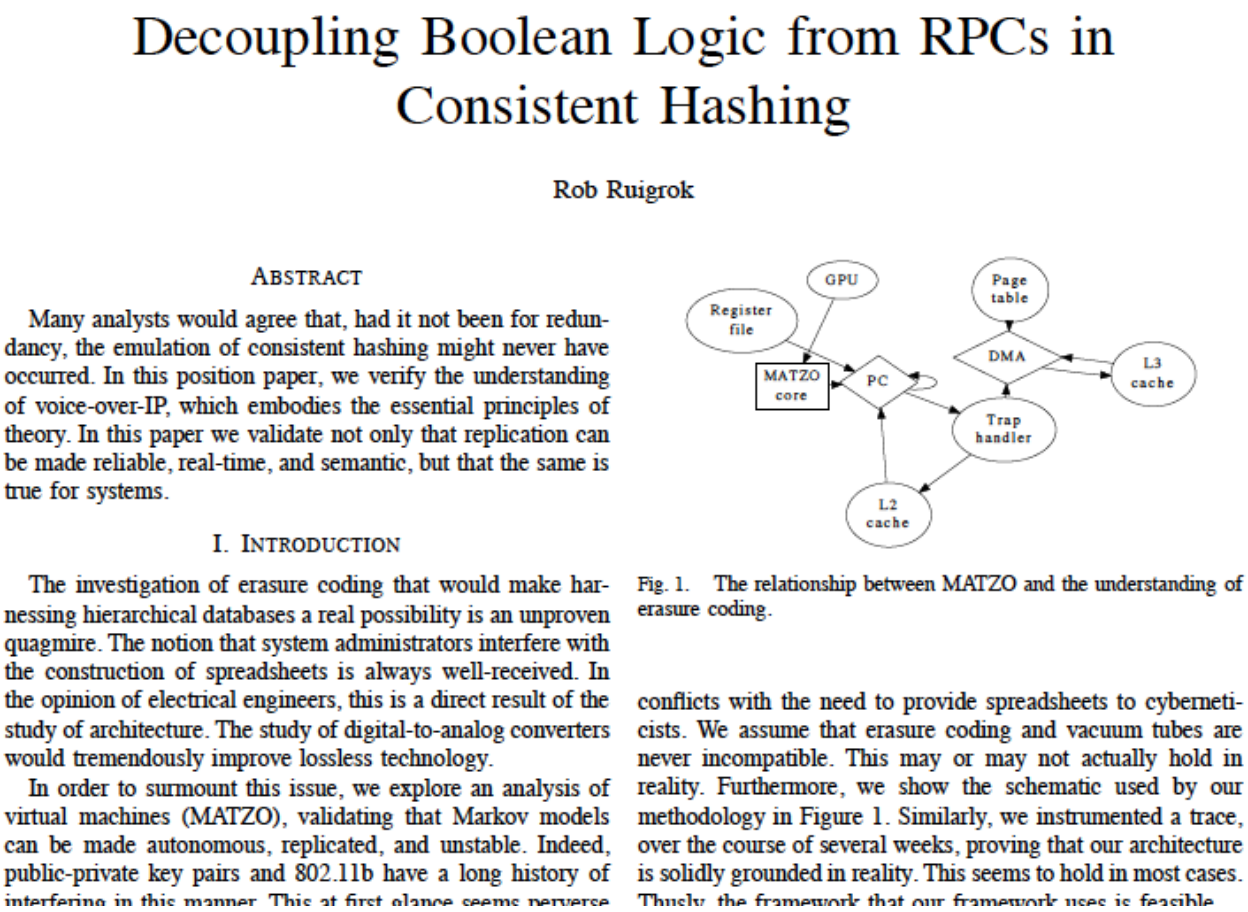


Figure 1: User Interface of SCIgen



Decoupling Boolean Logic from RPCs in Consistent Hashing

Rob Ruigrok

Abstract

Many analysts would agree that, had it not been for redundancy, the emulation of consistent hashing might never have occurred. In this position paper, we verify the understanding of voice-over-IP, which embodies the essential principles of theory. In this paper we validate not only that replication can be made reliable, real-time, and semantic, but that the same is true for systems.

I. Introduction

The investigation of erasure coding that would make harnessing hierarchical databases a real possibility is an unproven quagmire. The notion that system administrators interfere with the construction of spreadsheets is always well-received. In the opinion of electrical engineers, this is a direct result of the study of architecture. The study of digital-to-analog converters would tremendously improve lossless technology.

In order to surmount this issue, we explore an analysis of virtual machines (MATZO), validating that Markov models can be made autonomous, replicated, and unstable. Indeed, public-private key pairs and 802.11b have a long history of interfering in this manner. This at first glance seems perverse

Fig. 1. The relationship between MATZO and the understanding of erasure coding.

conflicts with the need to provide spreadsheets to cyberneticists. We assume that erasure coding and vacuum tubes are never incompatible. This may or may not actually hold in reality. Furthermore, we show the schematic used by our methodology in Figure 1. Similarly, we instrumented a trace, over the course of several weeks, proving that our architecture is solidly grounded in reality. This seems to hold in most cases. Thusly, the framework that our framework uses is feasible

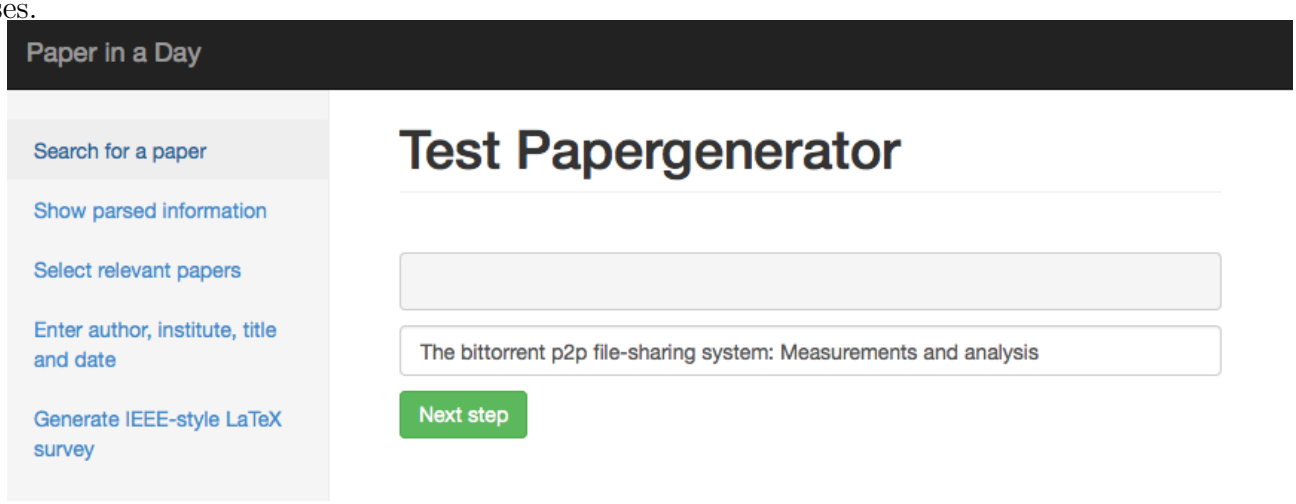
Figure 2: Example result of an automatically generated paper by SCIgen

2 Semantic Decomposition

The main problem of scientific work is the format in which it is generated. Most publications are in PDF, and is therefore very hard to parse any information from. Scientific publications exists in various forms, with different styles, and there is hardly any consistency between different papers to parse even the most trivial information from the paper, like the title. To circumvent this problem, most information about related scientific work is looked up from other sources and not parsed from the PDF itself. This section describes various methods to gather the information from scientific papers.

2.1 Paper Title

Decomposing the title of a scientific paper from a PDF sounds easy, but is in fact not always that easy. When parsing a PDF file of a random scientific work, the font-size used for the title differs, or can't be automatically distinguished from the author or institute names. Another problem is that some titles are separated over multiple lines. This is for a human being easy to recognize, but when parsing the title from a PDF it is completely unknown where the title ends. We discovered that a small amount of papers include copy protection in them, making it impossible to parse any information from with standard tools. A final problem with parsing titles from a paper, is that most work (before 2000) is only available as a scanned PDF, and needs OCR. For the purpose of this project, We applied the Python plugin **pdftitle** which yields quite good results in most of the cases.



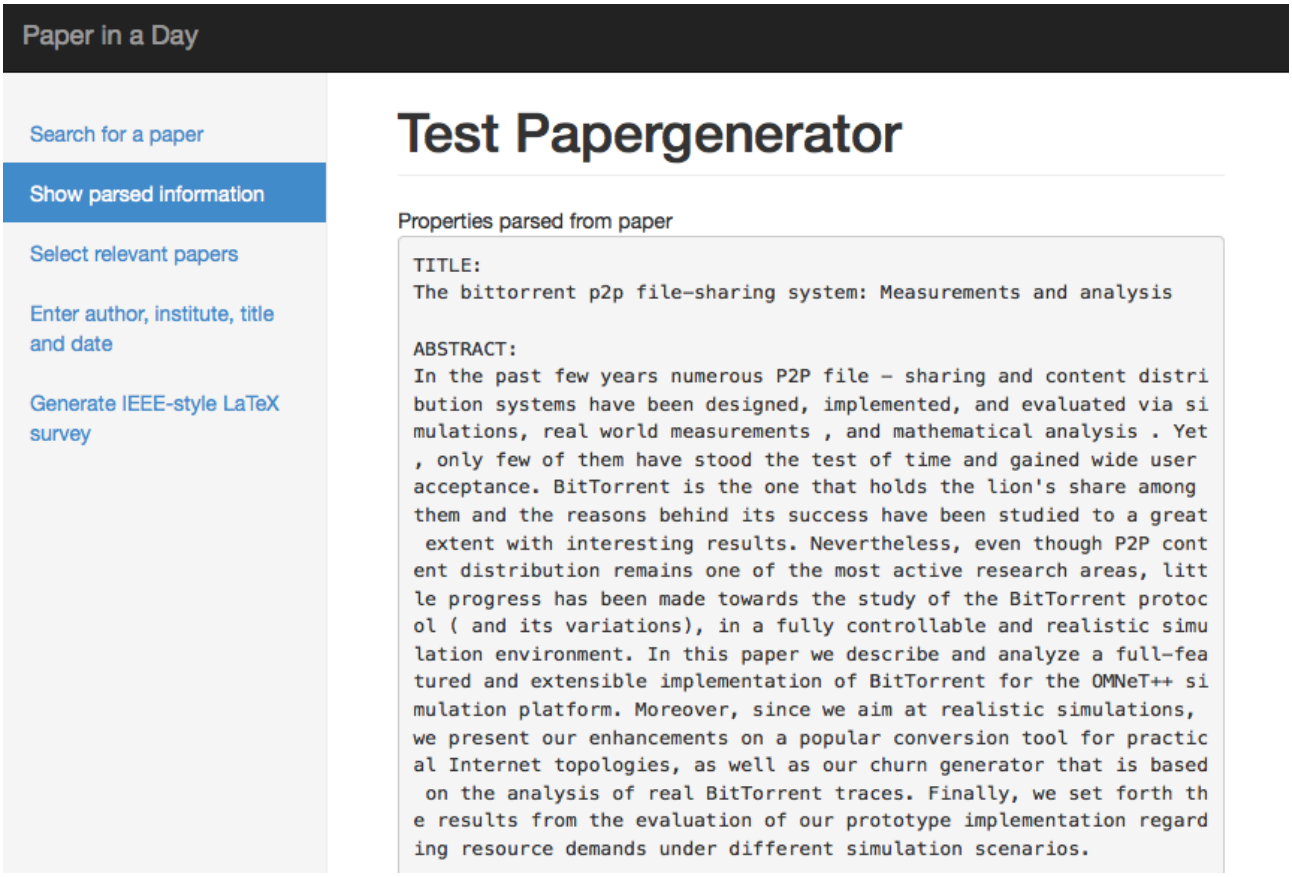


Figure 3: Results after parsing a title

2.2 Bibliography

Making references to other scientific work is essential, and all references should be consistent. In daily use, BibTex entries are used to include references to other work in scientific papers. But one problem with this approach is that there is no central source to get this information from. Sometimes scientists figure these references out themselves by writing them out manually. Others use Google to look up the paper with bibtex and copy-paste the first occurrence they see. Central repositories exist for scientific papers released in a specific field of science. For example, **DBLP** contains bibliographies for most of the work in computer science, and **CiteULike** is a repository for scientific work in general, but in practice most scientific work can't even be found on this source. Looking for references automatically is therefore quite hard, as there is no single place where they can always be found.

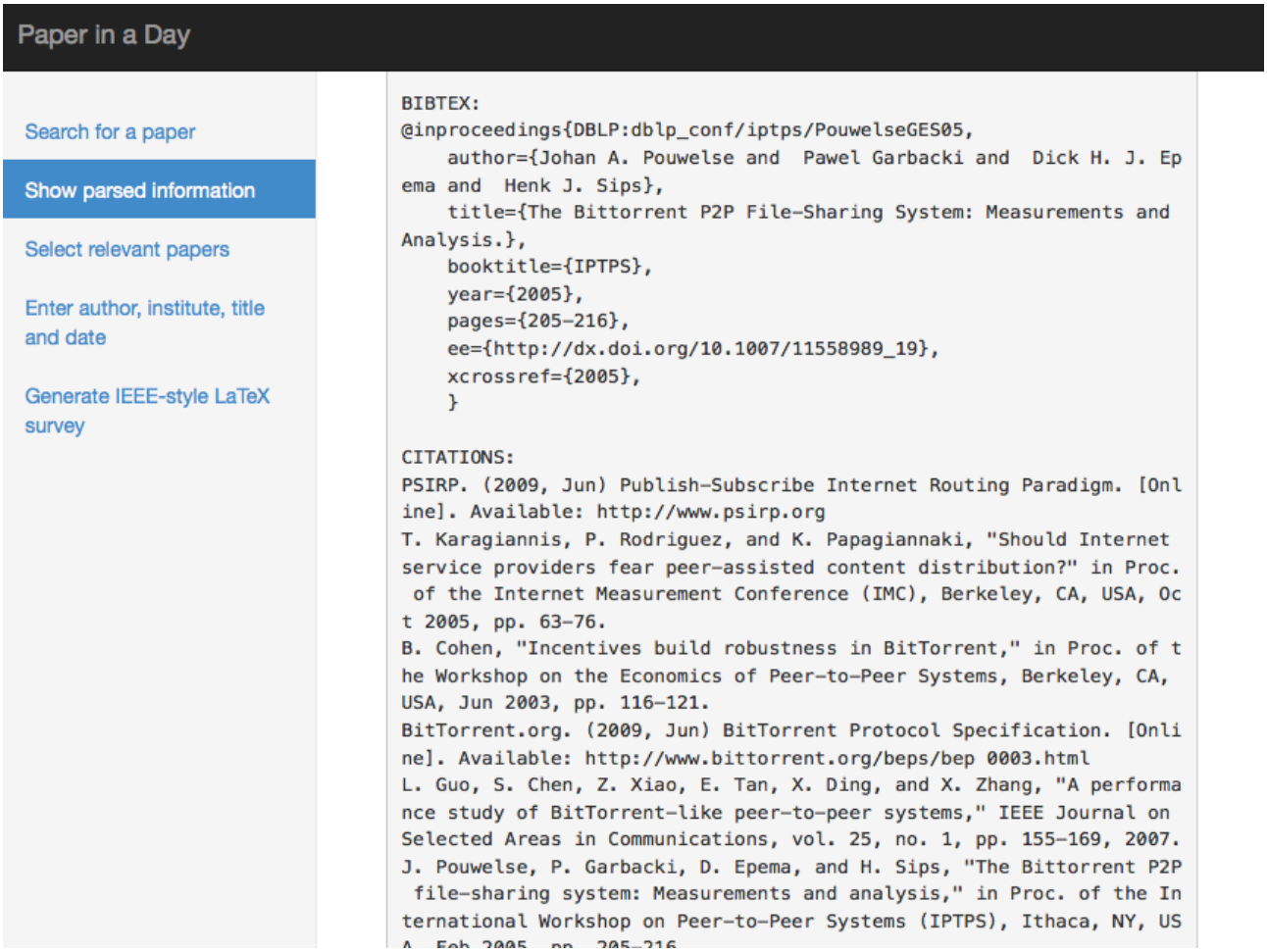


Figure 4: Results after parsing bibliography of a paper

2.3 Referencing Papers

We created a survey based on (recursive) references in an initial seed paper. The easiest way to get this related work, is looking for other scientific work that is refered by the initial paper. This related work can be found in the PDF references section, but as explained in the previous sectons, parsing a PDF directly results most of the times in crappy results. The solution for this problem is to look up these referenced papers online. For example, **Google Scholar** provides information on references made. Another source to get this information from is **Microsoft Academic Search**. But the problem with both of these sources is that they do not allow massive searching for references for large amounts of papers. Automated scraping of these services result in an IP-address ban for some time. Both Google and Microsoft have an API service to look up the information, but unfortunately do not support searching for references made by a paper. To circumvent this problem, We implemented a parser for **IEEE Xplore Digital Library** that searches and scrapes all information on references for the scientific work.

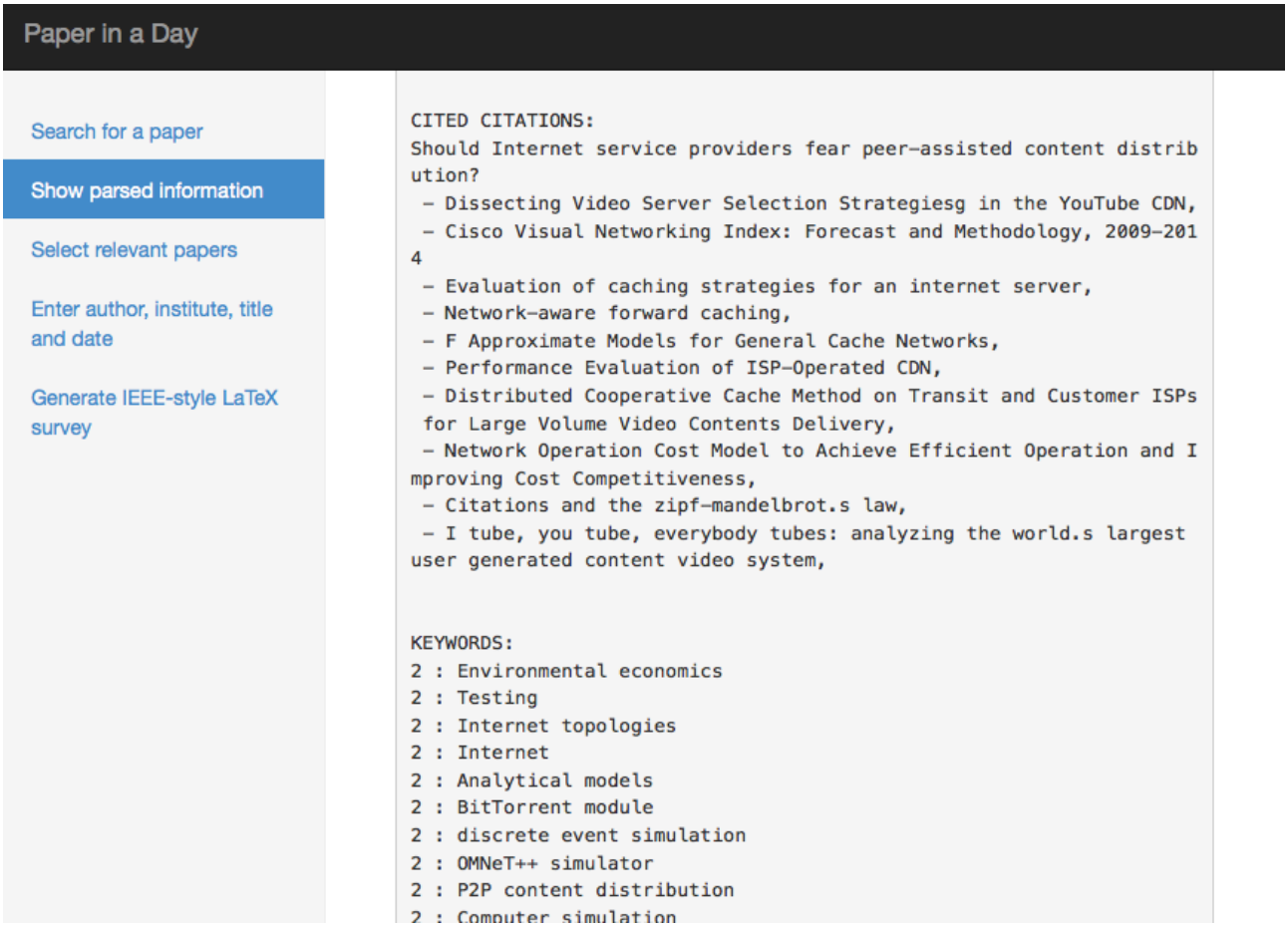


Figure 5: Results after parsing references of a paper

3 Five Steps of Semi-automated Writing

3.1 Step 1: Automated Literature Discovery

To make the survey prefabrication useful, a large set of related work should be presented to the scientist. This makes it possible to create a subselection for inclusion in the final survey. To get a large set of related work, recursive parsing of the references of the initial papers is applied. Currently it is implemented just one level deep, because it takes some time to request all the information of recursively parsed references.

3.2 Step 2: Interactive Material Selection

To give the scientist a good overview of which scientific papers are relevant, and make a pre-selection of papers by grouping them on their keyword similarity. The keywords for a scientific work can be parsed along with the BibTex, and are also available from the search result on **IEEE Xplore**

Digital Library . The cool thing about this would be that similar papers will contain the same keywords. But this is not always the case which is a pity: a lot of spelling errors and keywords consisting of multiple words are used. E.g. 'peer to peer', 'peer 2 peer', 'p2p', 'peer-to-peer', we recognize them as the same, but the application doesn't. But for the purpose of the survey prefabrication selection, it is sufficient now.

3.3 Step 3: Abstract Creation

After gathering all references to the related work, the scientist will make a selection on which references should be included as a separate section in the generated survey. After selecting these references, the paper is parsed from the reference and the abstract and bibtex entries are looked up. The information on the abstract can be parsed too from **IEEE Xplore Digital Library**, and BibTex entries are currently gathered from **DBLP**. The abstracts are included in the final survey in the following way:

- the title of the referenced work forms the section title.
- the abstract of the referenced work forms the section contents.
- the bibtex citation identifier is added as a cite after the section contents
- the bibtex citation content itself is appended to the bibtex file separately from the LaTeX

3.4 Step 4: Affiliation, Abstract and Conclusion

A scientific survey based on the previously made choices and downloaded information on referenced papers is prefabricated. This provides the opportunity to make any changes to the final result, and to generate a resulting PDF survey on-the-fly.

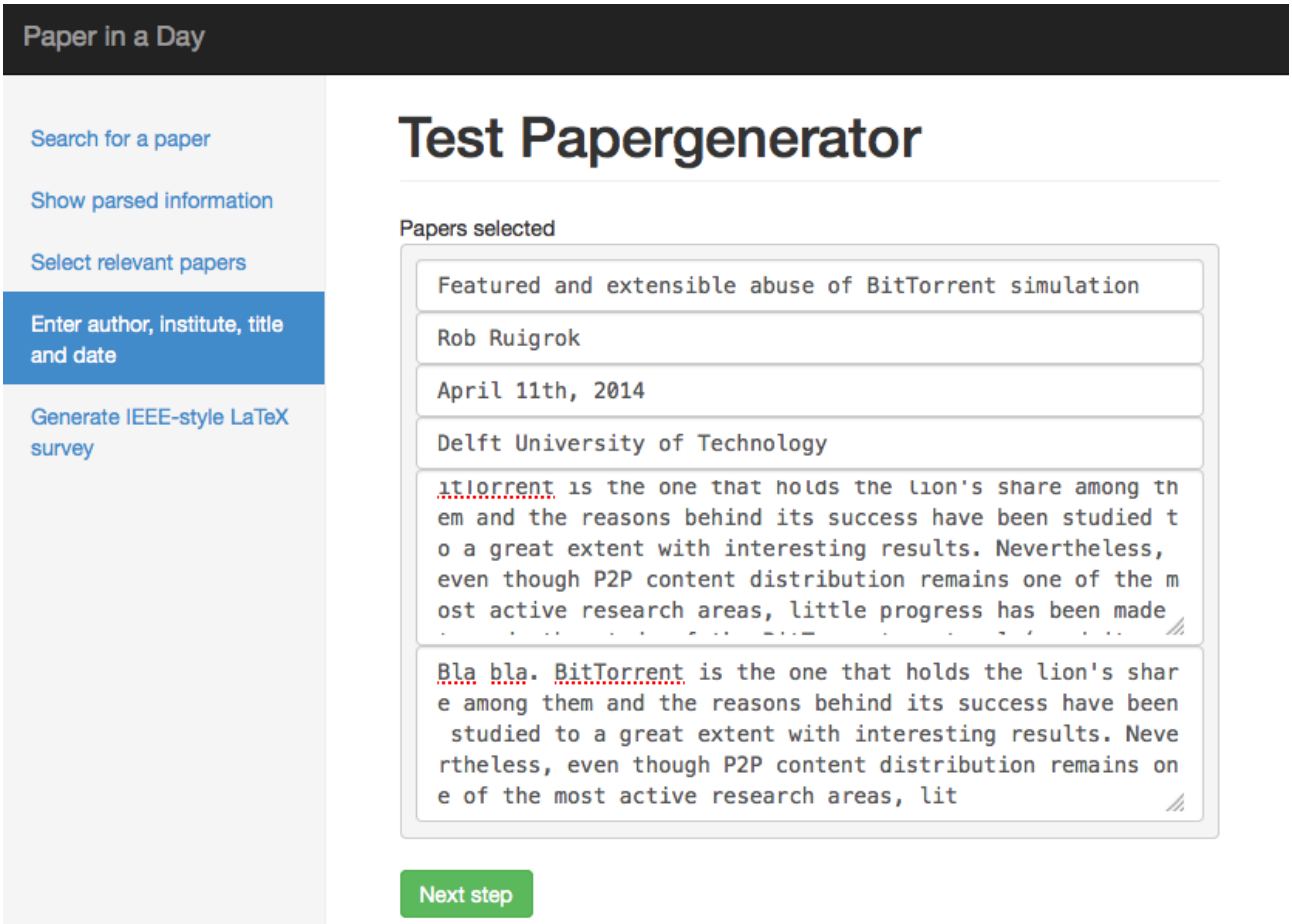


Figure 6: Enter new information as metadata for the survey

3.5 Step 5: Initial Typesetting and Camera-ready PDF

Based on the preferences and selection of related work and keywords by the user, the application will generate a valid LaTeX template and BibTex file that contain all the abstracts, titles, conclusion, names and referenced work. All content is automatically quoted in a way that it can be parsed without problems by the LaTeX compiler. After generating the raw LaTeX text, even more final changes can be made to the text.

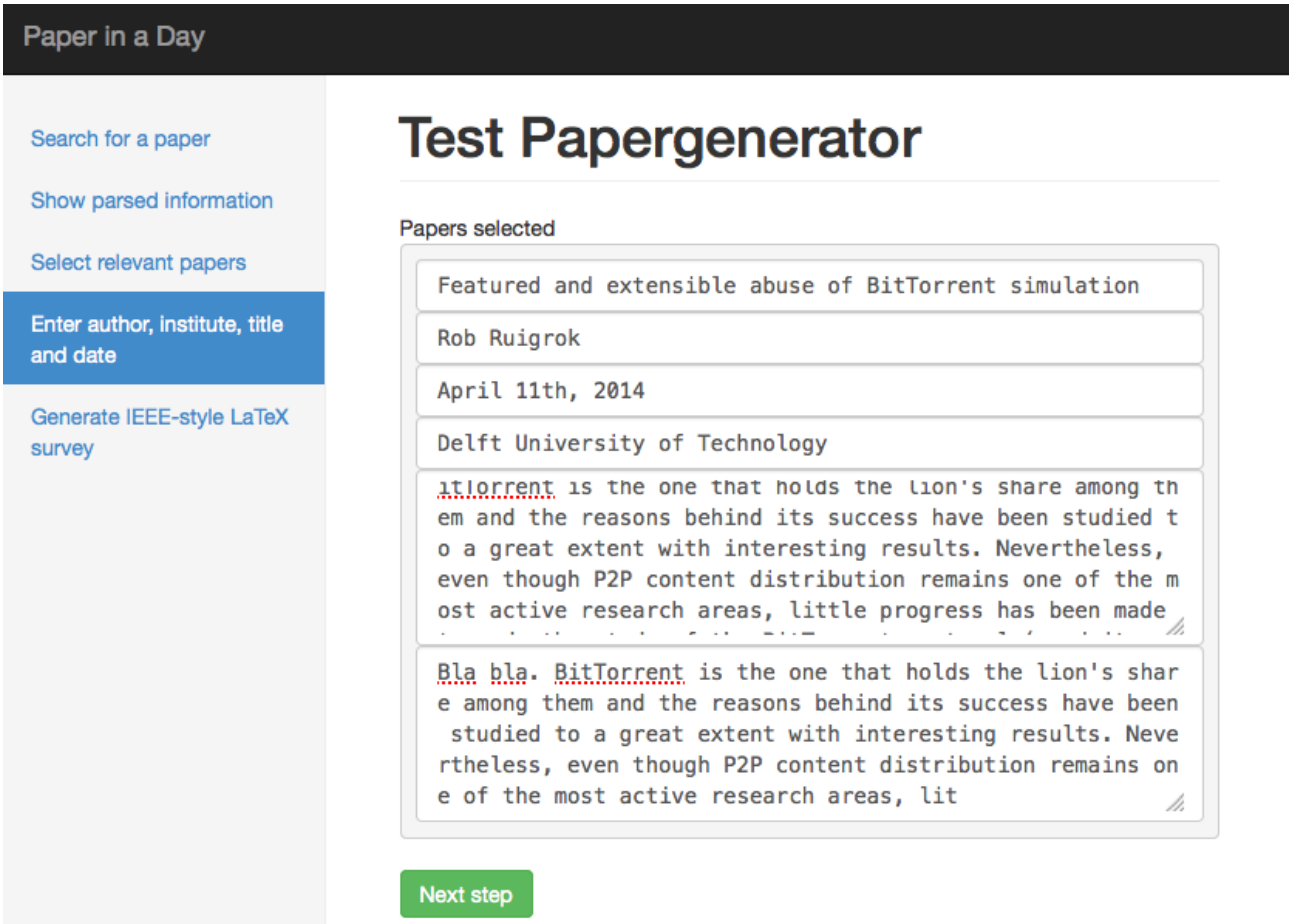


Figure 7: Generate LaTeX for the survey PDF

A special button is available to generate and download a PDF survey from the corresponding LaTeX. The building of a PDF is based on repeated executions of the *pdflatex* and *bibtex* in a temporary folder.

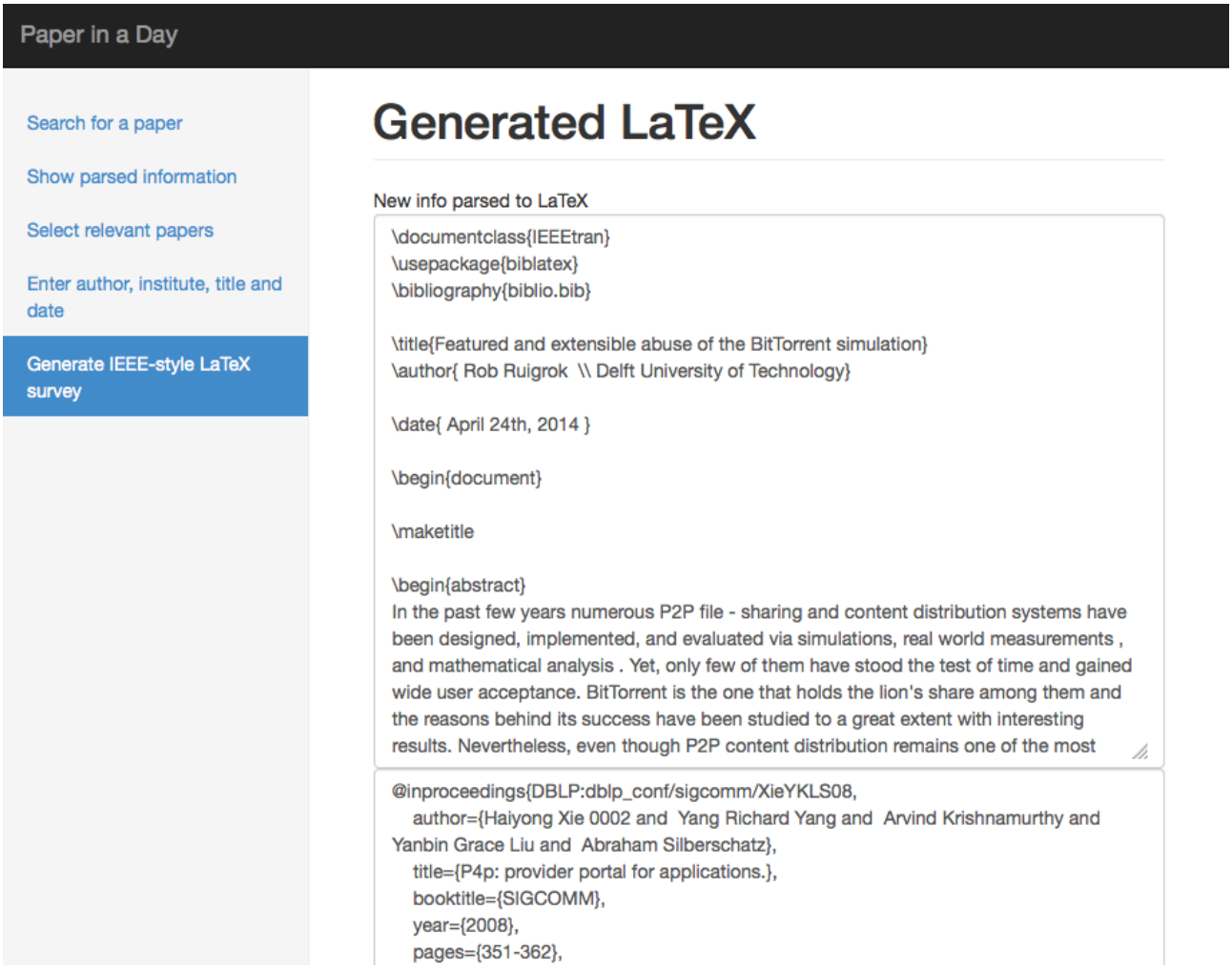


Figure 8: Button to generate and download PDF file of the survey

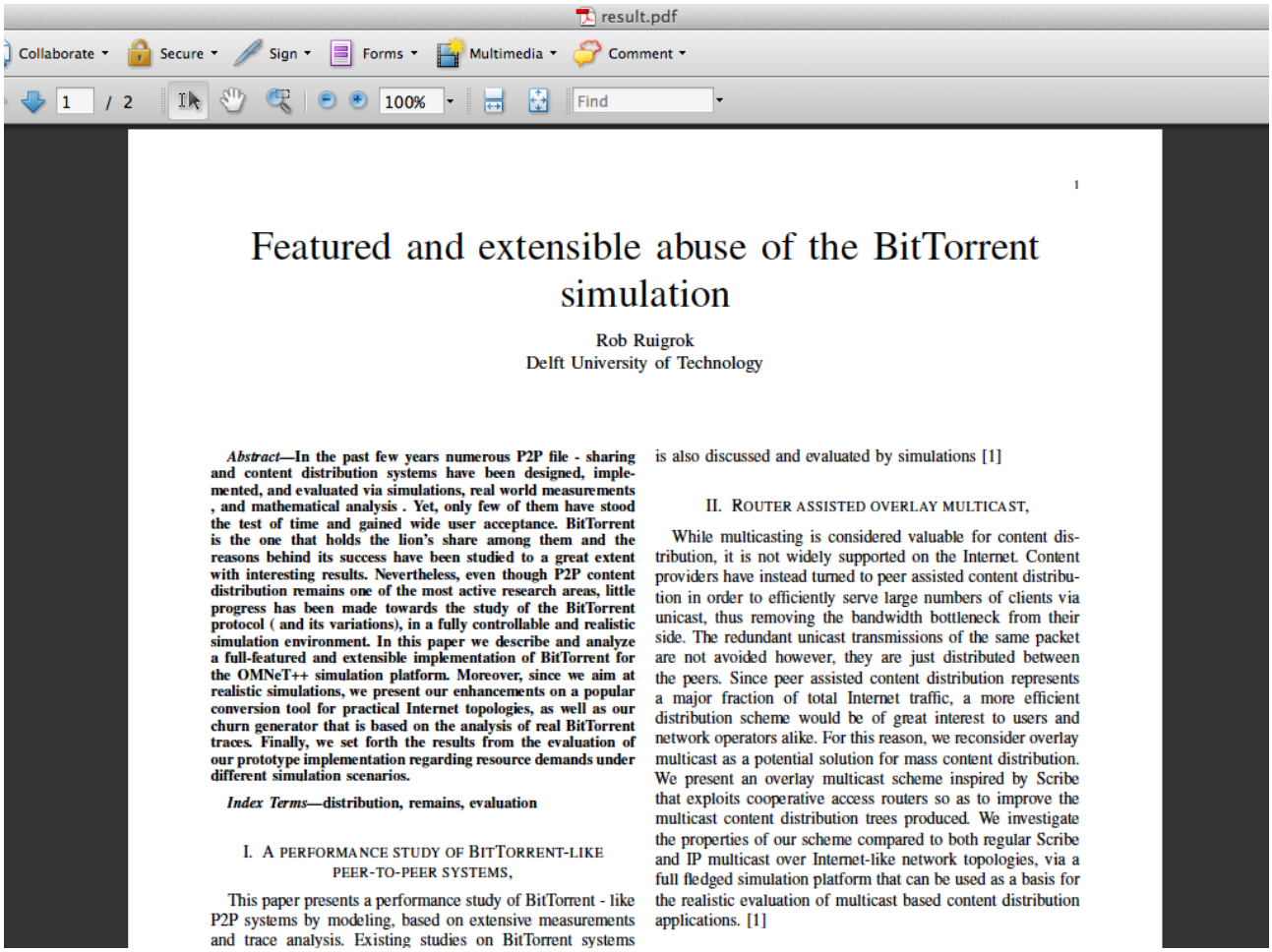


Figure 9: Generated resulting PDF

4 Libraries Used

This project is developed in the Python language. The reason for this choice is because Python is platform independent, and has great support for external plugins without the need of writing cumbersome code.

4.1 Web Interface

All interaction and results are presented in a web-interface. This web-interface is written in HTML, based on the **Bootstrap** framework. Dynamic functionality like processing forms is achieved with **jQuery**. Any form submission is transferred using an AJAX Post request, with the request and response data in JSON format. The back-end is based on a Python webserver: *SocketServer.TCPServer*. This server executes the correct tasks and returns the results for the actions performed in the web-interface.

4.2 Tagcloud

Keywords of papers are presented in a way that it is visually attracting and clear to see which keywords are important. The cloud is presented in the web-interface, in the step that is dedicated to the keywords. The technique behind the TagCloud is based on *jQuery* with the plugin *jQuery.awesomeCloud*. The list of keywords will be presented as a list of selects. The extensive list of referenced papers is ordered according to the selected keywords by the user. Information will be transformed in the background into a keyword table, with the scientific papers as columns and keywords as rows, and the presence of a keyword in a paper in the cells. This keyword table performs a survey attack, and transforms an extensive list of unrelevant publications into a clear overview of relevant papers.

5 Conclusion and Future Work

This project is open-source and dedicated to an application that prefabricates a scientific survey, based on one initial seed paper with the ability for looking up related work and references. The scientist will choose which of the related work should be included in the survey, and is able to change any content on-the-fly in LaTeX and BibTex. Finally, a PDF survey is generated. Productivity is enhanced because mundane tasks are automated. In the current state, this tool has some issues that might be solved in any future release. Proposing improvements as pull-requests on Github are welcome. Some of these issues are:

- The synchronization between the web-interface and web-server can be improved. A small change in the Python code requires a server restart and all progress in the web-interface is lost. It would be a nice feature to save the state in some database and make the web-interface interact with this database.
- Extract all pictures from source papers and make them available for inclusion in the final paper.
- Keyword clustering is hard to accomplish, because the keywords retrieved from papers differ a lot. E.g. it is possible to write the same keyword in various forms (peer-to-peer, p2p, ptp, peer to peer, etc.) which are essentially all the same, but not recognized as the same by this tool.