discuss: Embedding Dialog-Based Discussions into Websites

Christian Meter, Tobias Krauthoff, and Martin Mauve

Department of Computer Science
Heinrich-Heine-University Düsseldorf, Germany
{meter,krauthoff,mauve}@cs.uni-duesseldorf.de
http://cn.hhu.de

Abstract. In this paper we present the web application *discuss*, which provides a novel approach to embed structured discussions into any website. These discussions employ a formal argumentation system in their backend and can be used in addition to or replace existing comment sections. By interacting with the content of the website, we allow to include this content in the discussion. Furthermore, the same discussions can be accessed from multiple websites to bring their audiences together and create a single large discussion. To form a combined audience, it is necessary to use a common backend and we present an exemplary implementation of this scenario.

Keywords: online argumentation, collaborative work, discussions, dialog-based approach, web technologies, computer science

1 Introduction

Many websites and online news media provide their readers with the opportunity to comment and discuss their content. In fact, the ability to participate in such a discussion or to read what others think about an article is a major reason to prefer online content over offline media. While current solutions are quite suitable to provide simple feedback, they do a rather poor job at fostering meaningful discussions among the readers. This is especially true in those cases where this would be most needed: for articles that receive a lot of reader-feedback due to their popularity or controversial nature.

Commonly, comment sections are located below online articles. They provide a vertical-oriented discussion, where one comment follows the other, often combined with the possibility to directly reply to an individual comment. This is the same design used, for example, by Facebook or Twitter or, in fact, in most forum systems. It is well known, that this design has significant flaws when used for discussions and argumentation rather than simple feedback [1,2], for example redundant comments, lack of structure or simply missing scalability when large numbers of users try to express their opinions. Some online editorials, e.g. The Guardian, are really interested in the comments from the users to enrich the journalism, but often they are abusive, violate their community standards and

the journalists are confronted with huge numbers of comments, which they have to moderate [3]. In general, online editorials show keen interest in the discussions in the comments and are interested in the user's opinions.

To solve these problems and allow for meaningful online argumentation regarding issues raised in an online news media article, we propose to integrate *dialog-based online argumentation* in the website hosting the article. In dialog-based online argumentation the user performs a time-shifted dialog with those users who previously participated in the discussion. The new user can then react to statements from those other users and provide her own statements. This dialog is performed in natural language and the user does not need any specific skill other than being able to read and write. This concept has been implemented in the argumentation system *D-BAS* [4], which is a public accessible web-application. The system also provides an application programming interface to use its backend to remotely perform steps in the argumentation.

In this paper, we present *discuss*, which uses the interface of D-BAS to embed structured discussions in arbitrary websites. discuss is a JavaScript-based extension, which can seamless integrate dialog-based discussions into websites. This tool can be used to enhance or replace existing comment sections whenever a discussion is intended to be held with or among the readers. It gives users that participate in the discussion the option to add references to parts of the online article to their statements. Those parts are then marked in the article, so that other readers can jump right into the ongoing discussion. Furthermore, it is possible to browse and search for those arguments in the discussion that reference the current website.

Our main contributions in this paper are: (1) integrating the interface for dialog-based online argumentation into regular web-content, (2) allowing for references between the argumentation and the content of the website, (3) navigating the argumentation by means of links and search requests and (4) providing a way to use the same discussion across multiple websites.

The paper is structured as follows: Section 2 contains the related work to compare our approach to existing established technologies in the Web. Section 3 is about the prototype D-BAS and the concept of dialog-based argumentation. Section 4 describes the functionality of discuss, while Section 5 focuses on our implementation. The last Sections 6 and 7 conclude the paper and give an outlook to future work.

2 Related Work

The most popular tools to provide reader feedback are simple comment sections in form of a linear list of user statements or the use of forum-based systems. Both display all the negative aspects mentioned above. There are three specific systems that we want to discuss in more detail:

The first system is *Disqus*, which enables discussions on arbitrary websites [5]. In fact, Disqus is a JavaScript application, which needs to be installed by webmasters and brings a hosted alternative to self-hosted comment sections. One

unique characteristic is that instances from different websites can discuss about a global topic. Disqus does not introduce new techniques to enhance discussions and, in general, provides the same functionality as normal comment sections, i.e. add, reply to and vote on comments. This tool is popular for its simplicity and is therefore used quite frequently. It does not address the common problems of comment sections, though. Enabling a global discussion, however, is quite interesting and will also be used in our application.

rbutr [6] is a browser extension which gives the users the ability to link several websites sharing a common topic. These links can then be combined with arguments to introduce information from website B, which might support or rebut the article presented on website A. When a user then visits website A, she is presented a small popup showing that B provides arguments against the contents of A. Therefore, rbutr can be used to link contents from different websites to adjust false information presented on another website. The general idea of using contents from the Internet to support one's own statement is also used in discuss.

ArguBlogging from ARG-tech [7] can be installed as a bookmarklet[1], which needs no further configuration and can directly be used by interested users. The main concept of this tool is to select arbitrary text passages from websites and post them with a reference to the original source on one of the supported blogging sites, currently *tumblr* and *Blogger*. ArguBlogging then creates a post on the user's personal blog and gives her the ability to discuss about this text passage. A popup is presented to other users, who use ArguBlogging, when they arrive on a website, where another user already has selected some text and discussed it on her blog. These other users can then react to this statement and join the discussion. The idea behind the text-selection feature from ArguBlogging is also used in discuss, but in our case it will be directly integrated into a dialog-based discussion.

3 Dialog-Based Online Argumention

The goal of dialog-based online argumentation is to enable any user to participate efficiently in a large-scale online argumentation. At the same time it avoids, or at the very least reduces, the problems that occur in unstructured online argumentation such as a high level of redundancy, balkanization, and logical fallacies.

In the following, we briefly describe terms that will be used to explain the main aspects of dialog-based online argumentation. Based on these terms, we then introduce the main concepts of dialog-based online argumentation.

Each discussion is a set of *statements*, which are the most basic primitives used in an online discussion. The negation of a statement is itself a statement. Individual participants might consider a given statement to be true or false. A *position* is a prescriptive statement, i.e., a statement which recommends or

[1] `http://www.bookmarklets.com/about/`

demands that a certain action can be taken. Further on we need to distinguish between first-order and second-order *arguments*. A first-order argument consists out of a premise group — a set of at least one statement — and a conclusion, i.e. a statement. Both are connected by an inference, which is either supporting or attacking, so that the premise group is a reason for or against the conclusion. A second-order argument has the same kind of premise group, but the conclusion is the inference of an argument. With this we can argue about the validity of another reason-relation. Together, the arguments of a debate form a (partially connected) *web of reasons*.

The core idea of dialog-based online argumentation is a loop consisting of three steps: (1) presenting a single argument; (2) gather feedback from the user based on a list of alternatives and (3) the system selecting the next argument that is shown to the user based on the response and, possibly, the data gathered from the responses of other participants [4]. In this way the user and the system perform a *dialog* where the system selects arguments that are likely to be of interest to the user and then the user provides feedback on those arguments.

A first thing that the system needs to do when a new user wants to participate in the online discussion is to choose an initial argument. This is challenging since the system has no information on the user, yet. One fairly straightforward solution is to simply ask the participant for an initial position she is interested in. After she has chosen or provided her position, she is asked to select or provide a statement explaining her choice. This statement is used as the premise, whereas the position forms the conclusion.

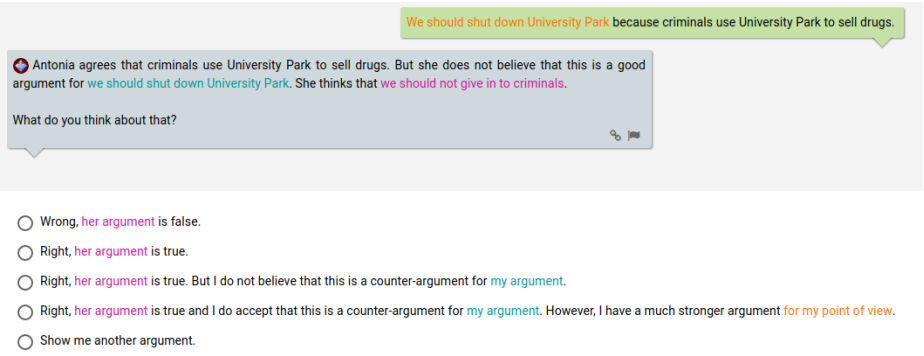

Fig. 1: Gathering feedback during a confrontation in D-BAS.

Once a user is confronted with an argument (see Fig. 1), she can provide feedback on the argument, as shown in Figures 1 and 2. Based on the feedback the system then selects the next argument that is shown to the user. A first

prototype implementing this idea is the dialog-based argumentation system (D-BAS) which is available for testing[2].

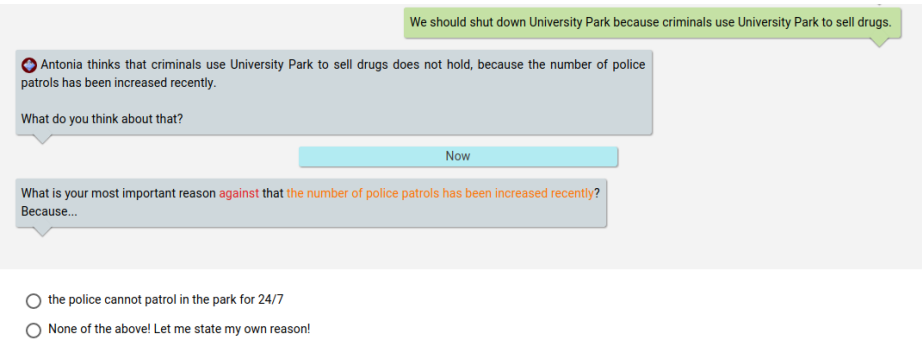

Fig. 2: Justification of the opinion in D-BAS.

4 Functionality of discuss

The idea of discuss is to embed dialog-based online argumentation into regular website content. To describe our implementation in more detail, we use an example where a city wants to reduce its spending and asks the citizen to propose some actions (positions) and to discuss them in detail. A user provided the position "We should shut down University Park" and other users started to discuss this position. This is the current state and we will show through this example how discuss works.

4.1 Embedding discuss into Online Articles

Imagine we have a discuss-powered website and have an article about the situation of the University Park. This article contains facts about the future of the University Park, which other users have proposed to close to cut spending of the city. As an example we assume that the article contains information about an investor, who is going to bear the costs of the park for the next years. We also assume, that our exemplary reader already has knowledge about the ongoing discussion and therefore knows some arguments in it. This is not absolutely necessary, but simplifies the explanation of our contribution.

The user starts reading this article. On her way through, she finds an interesting fact, which she wants to integrate in the discussion about closing University Park. To this end, she selects the appropriate text from the article, e.g. "*But apparently there is an anonymous investor ensuring to pay the running costs for*

[2] `https://dbas.cs.uni-duesseldorf.de`

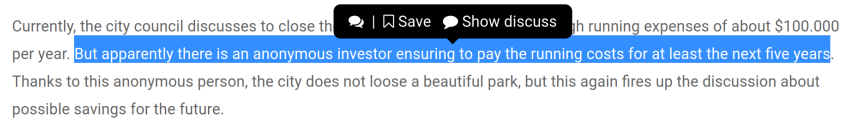

Fig. 3: Tooltip pops up when the user selects a text passage in the article.

at least the next five years". Selecting the text provides her with a *tooltip* (see Fig. 3). Possible options are "*Save*" and "*Show discuss*", where the first option stores the current selection in a *clipboard* for subsequent assembly of an argument for the discussion. The second option toggles the interface to discuss, so that she can directly participate in the discussion. To be flexible and not limited to specific websites, the interface is bound to a *sidebar*, which slides in from the right side, when the second option has been selected. In this sidebar all relevant elements are located which are necessary to participate in the discussion, see Fig. 4.

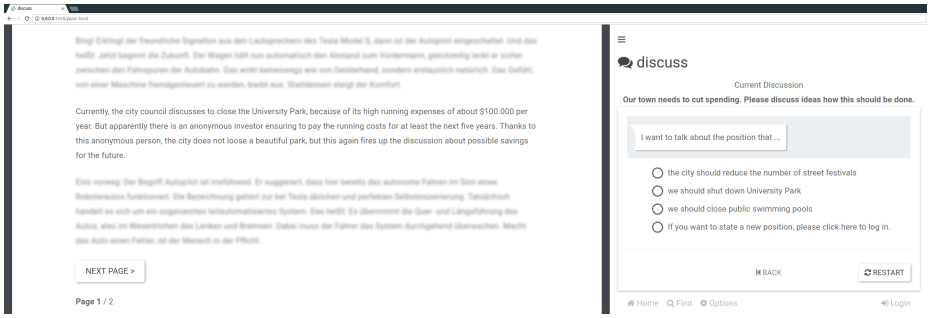

Fig. 4: Side-by-side integration of discuss into an online article.

Now, she can use the selected text and the interface of discuss to participate in the discussion and to create a direct citation of the text passage to her choice. We call these citations *text references* and the user can connect them with any statement in the discussion. With the knowledge the reader gained with this article, she is able to form a counter argument against closing the park and add a suitable reference to her statement. In this case her selection from above, pictured in Fig. 3, seems to be best-fitting, because it describes the future of the University Park in one sentence. These new facts are relevant and can stop the discussion about closing the University Park (if the sources of this article are trustworthy and the contents are true).

As a last step, the reader needs to add her argument to the correct location in the discussion. Since we are assuming, that she already has knowledge about the discussion, she can use the *search engine* for navigation. When the user now wants to add the fact that the investor is going to bear the costs, she needs

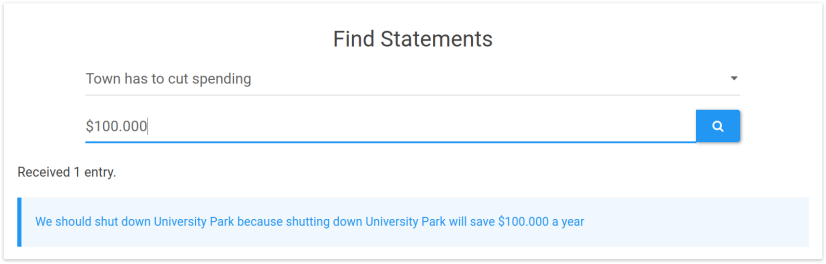

Fig. 5: Find position in the discussion, where the high costs of University Park is discussed.

to find the correct argument from the other user, e.g. "*We should shut down University Park, because shutting down University Park will save $100.000 a year*". Adding the exemplary input "$100.000" in the search engine (see Fig. 5) provides the statement we are looking for and we can now formulate our own argument against it supported by the reference from this article as it can be seen in Fig. 6. This completes the interaction with discuss and the user can close the sidebar to continue reading the article.

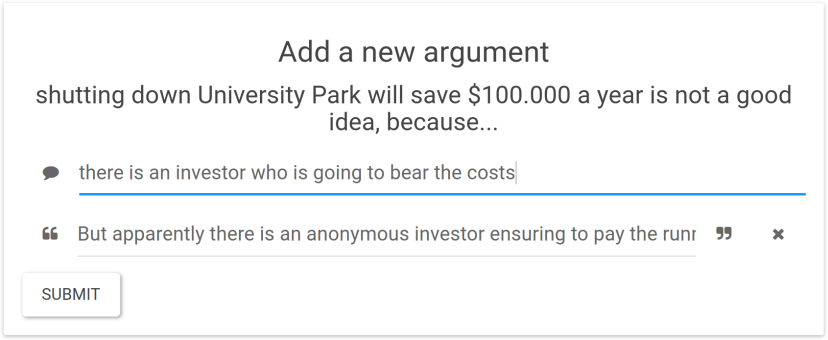

Fig. 6: Constructing a new argument with a text reference.

Arguments, references and their relations are stored in a common backend. All references from this article, which have been used in the discussion, are then highlighted in green color and appear in the text (see Fig. 7). Returning users or new readers of this article can easily see, that these text passages have been used in the discussion, and can interact with them by clicking on a reference. This click again toggles the sidebar and offers a simple interface with all linked locations in the discussion, where this reference has been used (see Fig. 8). Multiple locations are possible, since many users could use the same reference in their arguments.

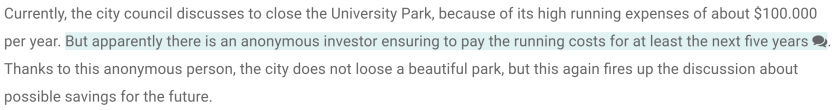

Fig. 7: Highlighted text reference which was previously used by a user.

Text references provide the easiest way to jump to a relevant position in the discussion and to directly start to discuss, because through a reference, our application presents the context of the related argument and asks the user how she wants to react to the argument, see Fig. 9.

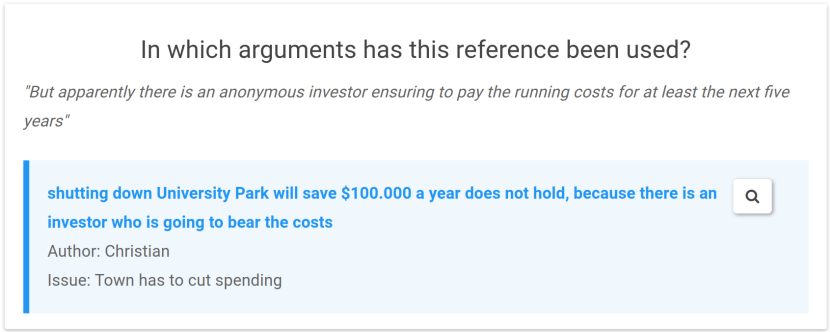

Fig. 8: Jump locations – shows where the references have been used.

4.2 Global Discussion

Common online news media websites, which provide a self-hosted comment section, only allow a local discussion. There is no possibility to leave the borders of this website to interact with users from other news media websites. Disqus [5] provides a feature for inter-website discussions, which we also included in discuss. To realize global discussions, we use one D-BAS instance as a common backend for websites that integrate discuss.

With these global discussions, a more heterogeneous peer group can be reached. Studies showed, that heterogeneous groups have a positive impact on the outcome of a discussion, i.e. solutions emerging from these discussions have a significantly higher quality and those solutions from homogeneous groups were never better compared to the heterogeneous group [8]. Therefore, enabling discussions among users from different online news media, with various levels of education and contrasting opinions, mutually support the discussion. Online news media are often known to have different audiences or specific political orientations and it could be very interesting to analyze discussions between those divergent peer groups, but this leaves the scope of this publication.

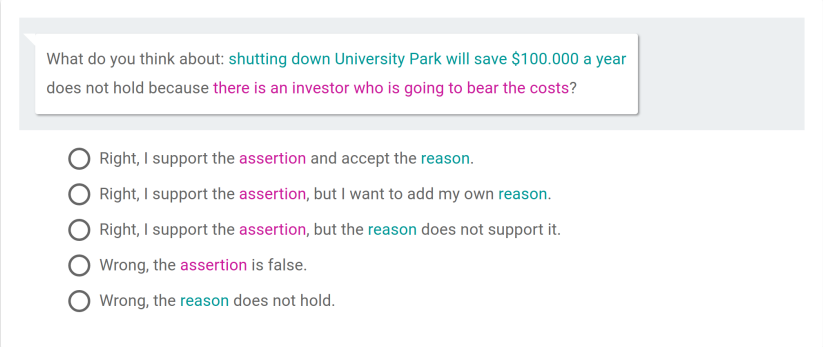

Fig. 9: Jump options – giving the user multiple options how she wants to react to the related argument.

5 Implementation

While implementing discuss we encountered a number of challenges that we outline in the following sections.

5.1 Technical Foundation

To create an application, which does not slow down existing websites and can pick any desired position in the *document object model* (DOM) of the website, we need to have powerful programming techniques and languages fitting our needs. The first prototype was implemented in pure JavaScript, but after few weeks the application became too complex and it was clear that we needed a framework to keep clean code and to reduce complexity. We were also unsatisfied with state-handling and the general language design of JavaScript, which is why we switched to the functional programming language *ClojureScript*[3] and re-implemented the functionality of the first prototype with just a few lines. ClojureScript compiles down to optimized JavaScript code with the *Google Closure Compiler*[4], which results in much faster code than we could manually develop. Using this compiler collection produces also much smaller production files thanks to advanced optimizations and dead code elimination. For dynamic user-interface handling, we chose Facebook's *React.js*[5].

These components allowed us to implement a stable and small web-application without disturbing or conflicting the website it has been embedded into. Since discuss adds many features and DOM manipulations as seen in the previous section, it is very important to choose the best-fitting components, because otherwise it would result in a slow or crowded application.

[3] `https://clojurescript.org/`
[4] `https://developers.google.com/closure/compiler/`
[5] `https://facebook.github.io/react/`

5.2 Including discuss in an Arbitrary Website

Website operators only need to include the to a single compiled and compressed JavaScript file to enable the features described in this paper for their websites. discuss searches in the DOM for a suitable entry point to enable dialog-based discussions. Selecting the text according to Subsection 4.1 is automatically available and the sidebar invisibly includes itself until the toggle in the tooltip is pressed. If an optional `div` is available in the DOM, an additional interface will be displayed on the website.

Enabling the discussion directly when the user reads the text is a difficult problem: the integration should not disturb the user, but should encourage her to participate in the discussion. In our first approach we put the discussion system directly between the lines of the article and split the text when the user toggled discuss with a switch. But this slide effect was very confusing and is possibly not usable in most kinds of websites. We then experimented with including the interface below the article. This also proved to be a bad choice since the reader then has to jump to the bottom of the article to participate in a discussion triggered by a statement in the article. In our final version, we used the sidebar to interact with discuss. Optionally, the webmaster can include a second interface by simply adding a `div` with a specific ID.

Using a tooltip can be seen on several websites, like Medium [9]. We added listeners to the article to activate the tooltip, when a text passage has been selected. This provides an unobtrusive method to interact with our application.

The clipboard temporarily stores the user's text selections for later usage. This has been implemented to provide the possibility to read the text, store interesting passages and keep on reading, see Fig. 10. In the end, the user can pick her favorite selection to add it to her argument via drag and drop.

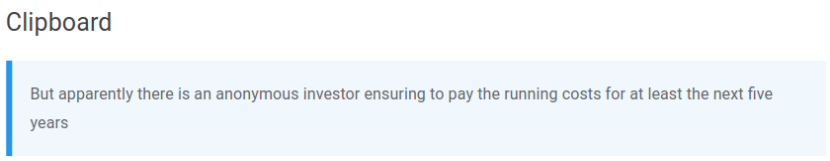

Fig. 10: Using a clipboard to locally store text references.

It is not possible to directly modify the contents of a reference. Our idea is that it should be a direct quotation of the article which is also technically required to find the same text passage in the article. Otherwise, new users will not be able to see the colored reference in the text. We are aware that it is currently still possible to modify the DOM to add a reference of your own desires or to use the browser console for modifications. This would create an untruthful reference, which could lead to false information and false trust in an argument. A server-side verification that the provided string can exactly be found in the article is thinkable, but is currently not implemented.

5.3 Execution Platform

For first testing purposes, we set up a D-BAS instance at our university. As the default configuration uses discuss this backend to directly demonstrate a fully functional application with global discussions enabled. It is possible to use its own backend, which is conform to our application. Therefore, it is not necessarily needed, that the backend is a D-BAS instance – it just needs to provide a suitable interface so that discuss can interact with it.

We are following common best-practices in web development and implemented a RESTful API in D-BAS to expose an interface for external applications, who want to use this dialog-based backend for their applications, whilst discuss is the first project using this interface. This approach for discussion software has already been described in [10] and it presents the general approach how to achieve reusable components in software development, which is why we are also following this structure. Furthermore, [10] proposes the idea to encapsulate the core argumentation logic into an own platform called *Dialog Game Execution Platform* to develop a reusable argumentation core and make it accessible for other applications. In our examples from this paper are we using D-BAS as our default execution platform.

6 Conclusion

Asking the readers to leave a comment below an online news media article is common practice on most websites. But with state-of-the-art comment sections, crowded masses of comments are a typical result. discuss helps to structure discussions and to conduct more productive discourses.

In this paper we used techniques from dialog-based online argumentation to enable our idea of more structured discussions in arbitrary contexts. To achieve this, we implemented discuss as a web application, which follows basic principles of our dialog-based approach and extends discussions by enabling references, global discussions and flexible inclusions into websites.

Feel free to test discuss under `http://cn.hhu.de/discuss` and you are welcome to provide us your feedback.

7 Future Work

We are currently working on more use cases of dialog-based discussions and are evaluating, where our approach could enhance the discourse experience on the Internet. Next, we will extend discuss to support more functions from our backend, e.g. premise groups. In addition, we will evaluate our application in real-world applications and try to cooperate with well-known online news media providers.

Since many people are actively participating in discussions in social networks like Facebook, we will investigate how we can integrate structured discussions into this context. Conceivable are solutions as social bots, which interact with the users based on text messages.

References

[1] Klein, M.: Using metrics to enable large-scale deliberation. In: Collective intelligence in organizations: A workshop of the ACM Group 2010 Conference, pp. 103–233 (2010). DOI 10.1177/004057368303900411

[2] Spada, P., Klein, M., Calabretta, R., Iandoli, L., Quinto, I.: A First Step toward Scaling-up Deliberation: Optimizing Large Group E-Deliberation using Argument Maps (2014). DOI 10.13140/RG.2.1.3863.5688

[3] The Guardian: The dark side of guardian comments (2016). URL `https://www.theguardian.com/technology/2016/apr/12/the-dark-side-of-guardian-comments`. Accessed on: 2016-10-14

[4] Krauthoff, T., Baurmann, M., Betz, G., Mauve, M.: Dialog-Based Online Argumentation. Proceedings of the 2016 conference on Computational Models of Argument (COMMA 2016) (2016). DOI 10.3233/978-1-61499-686-6-33

[5] Disqus, Inc.: Disqus - the no.1 way to build an audience on your website (2016). URL `https://disqus.com`. Accessed on: 2016-10-14

[6] rbutr: rbutls, debunkings and counter arguments to misinformation on the internet - think again (2016). URL `https://rbutr.com`. Accessed on: 2016-10-14

[7] Bex, F., Snaith, M., Lawrence, J., Reed, C.: ArguBlogging: An application for the Argument Web. Web Semantics: Science, Services and Agents on the World Wide Web **25**, 9–15 (2014). DOI 10.1016/j.websem.2014.02.002. URL `http://linkinghub.elsevier.com/retrieve/pii/S1570826814000079`

[8] Hoffman, L.R., Maier, N.R.: Quality and acceptance of problem solutions by members of homogeneous and heterogeneous groups. The Journal of Abnormal and Social Psychology **62**(2), 401 (1961). DOI 10.1037/h0044025

[9] A Medium Corporation: Medium (2016). URL `https://medium.com`. Accessed on: 2016-10-16

[10] Bex, F., Lawrence, J., Reed, C.: Generalising argument dialogue with the dialogue game execution platform. Proceedings of the 2014 conference on Computational Models of Argument (COMMA 2014) pp. 141–152 (2014). DOI 10.3233/978-1-61499-436-7-141