Proceedings of the 3rd International Workshop on Social Influence Analysis (SocInf 2017)
August 19th, 2017 - Melbourne, Australia

A Distributed Approach for the Analysis of Discussions in Twitter

Teresa Alsinet[1], Josep Argelich[1], Ramón Béjar[1], Jordi Planes[1], Joel Cemeli[2], and Cristian Sanahuja[2]

[1] INSPIRES Research Center – University of Lleida
Jaume II, 69 – 25001 Lleida, Spain
{tracy, jargelich, ramon, jplanes}@diei.udl.cat * **
[2] Polytechnic School, University of Lleida
Jaume II, 69 – 25001 Lleida, Spain
jcs13@alumnes.udl.cat, csanahuja10@gmail.com * **

Abstract. In a recent work we have developed an argumentative approach for discovering relevant opinions in Twitter. A Twitter discussion is modeled as a weighted argument graph where each node denotes a tweet, each edge denotes a criticism relationship between a pair of tweets of the discussion and each node is attached with a weight that denotes the social relevance of the corresponding tweet in the discussion. In the social network Twitter, a tweet always refers to previous tweets in the discussion, so the obtained underlying argument graph is acyclic. Based on this structural feature, in this work, we introduce a distributed algorithm for computing the set of globally accepted opinions of a Twitter discussion. The set of accepted opinions is extracted by mapping the weighted argument graph into a valued argumentation framework and it is computed as the biggest set of tweets of the discussion that satisfies that it is consistent.

1 Motivation and Antecedents

In order to understand what are the major accepted and rejected opinions in different domains by Twitter users, in a recent work [1] we have developed a system for analysis of discussions in Twitter.

The system architecture has two main components: a discussion retrieval and a reasoning system. The discussion retrieval component allows us to move from

** This work was partially funded by the Spanish MICINN Projects TIN2014-53234-C2-2-R and TIN2015-71799-C2-2-P.

*** The fifth author and sixth author acknowledge the finantial support of a student grant from the MECD/AGAUR agencies of the Spanish Government and the finantial support of a student grant from the Student Vice-Rectorate of the University of Lleida, respectively.

a discussion in Twitter (a set of tweets) in natural language to a weighted graph which is computed taking into account criticism relationships between tweets and three different attributes of a tweet: the number of followers of the author, the number of retweets and the number of favorites. The reasoning system component maps the weighted graph into a weighted argumentation framework and the set of socially accepted tweets in the discussion is evaluated from the weight assigned to each tweet and the criticism relationships between the tweets of the discussion, and it is computed as the ideal semantics [8] of a valued abstract argumentation framework [2].

In abstract argumentation [7], a graph is used to represent a set of arguments and counterarguments. Each node is an argument and each edge denotes an attack between arguments. Several different kinds of semantics for abstract argumentation frameworks have been proposed that highlight different aspects of argumentation (for reviews see [3,4,15]). Usually, semantics are given to abstract argumentation frameworks in terms of extensions. For a specific extension an argument is either accepted, rejected, or undecided and, usually, there is a set of extensions that is consistent with the semantic context.

The system developed in [1] builds a weighted argument graph for a Twitter discussion, where each node denotes a tweet, each edge denotes a criticism relationship between a pair of tweets of the discussion and each node is attached with a weight that denotes the social relevance of the corresponding tweet in the discussion and it is computed from some tweet's attributes. In the social network Twitter, a tweet always answers or refers to previous tweets in the discussion, so the obtained underlying argument graph is acyclic.

Based on the fact that the graphs we obtain are acyclic, in this work, we introduce and investigate a distributed implementation of the skeptical approach based on the ideal semantics of a valued abstract argumentation framework. A tweet defeats a second one if it criticizes the second one and it has at least the same social relevance as the second one. The ideal semantics for valued abstract argumentation guarantees that the set of tweets in the solution is the maximal set of tweets that satisfies that it is consistent, in the sense that there are no defeaters among them. But the solution also satisfies that it defends against defeaters from the tweets outside the solution, in the sense we will see in the formal definition of ideal semantics.

The social relevance is measured with a social valuation function that for each tweet considers different information sources from the social network, such as the number of followers of the author, the number of retweets and the number of favorites. The distributed approach can be of special relevance for assessing Twitter discussions that involve a large number of tweets and the system can be applied in fields where identifying groups of tweets globally compatible or consistent, but at the same time that are widely accepted, is of particular interest, such as for instance for the assistance and guidance of marketing and policy makers.

After this introduction, in the next section, we formalize the structure to model Twitter discussions and, in Section 3, we define the reasoning model for

computing their solutions. Then, in Section 4, we present a novel distributed strategy for the implementation of the reasoning model based on the ideal semantics for a valued abstract argumentation framework. We end the paper with some conclusions and a discussion of future work. As far as we know, this is the first distributed algorithm presented to solve such problem.

2 A weighted graph for Twitter discussions

Following the approach proposed in [1], in this section, we introduce a computation structure (a weighted graph) to represent a Twitter discussion considering only criticism relationships between pairs of tweets.

Definition 1. *(Twitter Discussion) A Twitter discussion Γ is a non-empty set of tweets. A tweet $t \in \Gamma$ is a tuple $t = (m, a, fl, r, fv)$, where m is the up to 140 characters long message of the tweet, a is the author's identifier of the tweet, $r \in \mathbb{N}$ is the number of retweets and $fv \in \mathbb{N}$ is the number of favorites. Let $t_1 = (m_1, a_1, fl_1, r_1, fv_1)$ and $t_2 = (m_2, a_2, fl_2, r_2, fv_2)$ be a pair of tweets of a Twitter discussion Γ. We say that t_1 answers t_2 iff t_1 is a reply to the tweet t_2 or t_1 mentions (refers to) tweet t_2.*

Definition 2. *(Discussion Graph) The Discussion Graph* (DisG) *for a Twitter discussion Γ is the directed graph (T, E) such that*

- *for every tweet in Γ there is a node in T and*
- *if tweet $t_1 = (m_1, a_1, fl_1, r_1, fv_1)$ answers tweet $t_2 = (m_2, a_2, fl_2, r_2, fv_2)$, with $a_1 \neq a_2$, and m_1 criticizes the claim expressed in m_2, there is a directed edge (t_1, t_2) in E.*

Only the nodes and edges obtained by applying this process belong to T and E, respectively.

Although our system allows us to analyze any discussion in Twitter (set of tweets), in this work we deal with discussions where a tweet answers previous tweets in the discussion. Moreover, in our approach, $(t_1, t_2) \in E$ iff $t1$ answers t_2 and the message of tweet t_1 does not agree with the claim expressed in the message of tweet t_2. So, the answers between tweets whose messages are not classified as criticisms, do not give rise to edges and, therefore, some nodes (tweets) of a discussion graph can be disconnected. Thus, we can find nodes for which the input or the output degree, or both, is zero.

Since the social network we are considering in this work is Twitter, every tweet of a discussion can reply at most one tweet, but can mention many tweets, and all of them are prior in the discussion. So, every tweet can answer and, in turn, can criticize many prior tweets of the discussion, and thus, every tweet can criticize many prior tweets from a same author and from different authors.

From an implementation point of view, in order to check if a tweet criticizes another tweet, we can use some of the components we have developed in [1]. On the one hand, to check if a tweet t_1 replies a tweet t_2, the system can use the

data structure in the JSON format provided by the Twitter API. In particular, this fact can be easily checked from the attribute `in_reply_to_status_id` of the object of t_1, which provides the tweet identifier to which t_1 replies. To check the set of mentions of t_1, the system searches for all authors mentions in the message of t_1. Every mention of an author is stored in the message with a label of the form: `@<author_identifier>`. So, t_1 mentions t_2 whenever the author's identifier of t_2 is in the set of mentions of t_1.

On the other hand, to check if a tweet does not agree with the claim expressed in a different tweet, the system uses an automatic labeling system based on Support Vector Machines (SVM). Our SVM model for labeling relations between tweets considers different attributes obtained from the tweets of an answer (t_1, t_2). On the one hand, we have attributes that count the number of occurrences of relevant words in the tweets t_1 and t_2. We have considered two kinds of words: regular words and stopwords. We have considered the inclusion of stopwords as attributes because the typical tweet is very short and the fraction of stopwords that can be giving information about the kind of answer could be relevant. For example, in the next tweet

```
@ponpimpampum @LL_Sosa Jajajaja...!!!
```

the stopwords `...` and `!!!` give information about the sentiment associated to the tweet.

On the other hand, we also consider attributes that have to be computed from the text and from the additional information that comes with the tweets. In particular, for each tweet these attributes are the number of images and the number of URLs mentioned in the tweet, the number of positive and negative emoticons and the sentiment expressed by the tweet. Our labeling system incorporates a sentiment analysis computation module [12,14] that given the set of words in a tweet it provides a sentiment value in the range $[-5, +5]$, where -5 is the most negative sentiment and +5 is the most positive sentiment. Finally, the sentiment value is incremented (or decremented) considering every positive (negative) emoticon.

Since SVM follows a supervised learning approach, we first have to train a model from an already labeled data set of answers. To this aim, we have collected a set of several Twitter discussions, on the Spanish language, and we have manually labeled the answers in the discussions to be able to train a SVM labeling model for Spanish discussions.

The training collection contains 12 discussions and a total of 582 pairs of tweets (answers). We have considered the creation of SVM models with different number of regular words (w) and different number of stopwords (s). The words in the collection are sorted by number of occurrences, and for an SVM model with w regular words, we select the first w most frequent regular words. The stopwords have been obtained from the natural language toolkit (NLTK) [6] and have been also ordered by number of occurrences, so we also select the s most frequent stopwords.

Using this training collection, we have trained four models with different values for w and s, in order to get a good labeling model. In order to compute the sentiment for the tweets in this collection, we have taken the AFINN data[3] used in [12,14] and we have translated the words to Spanish. See [1] for a detailed description of the SVM training model that we have implemented for checking criticism relationships between tweets from Spanish Twitter discussions.

Definition 3. *(Weighted Discussion Graph) A weighted discussion graph* (WDisG) *for a Twitter discussion Γ is a tuple $\langle T, E, R, W \rangle$, where*

- *(T, E) is the* DisG *graph for Γ,*
- *R is a nonempty set of ordered values and*
- *W is a weighting scheme $W : T \to R$ that assigns a weight value in R to each tweet in T, representing the social relevance of the tweet.*

Regarding the implementation, we instantiate the set of ordered values R to the natural numbers $\mathbb{N}$ and, for each node with tweet $t = (m, a, fl, r, fv)$, we consider the following weighting scheme:

$$W(t) = \lfloor \log_2(fl + 20 * r + 40 * fv + 1) \rfloor,$$

which takes into account not only the number of followers of the author, but also the number of retweets and favorites. This function allows us to quantify the orders of magnitude of the social relevance of tweets following the statistics about tweets and retweets defined in [5], trying to give each attribute a weight proportional to its relevance. From the statistics shown in [5], we observe that on weighting with twenty times the value of retweets and forty times the value of favorites, the magnitudes of the three attributes are comparable and one attribute does not dominate the others, since the number of followers is usually much bigger than the number of retweets and favorites. We finally compute the $\log_2$ function of the combined value, since we want to consider that one tweet is more relevant than another only if such combined weight is at least two times bigger for the first tweet. We will refer to this weighting scheme as `fl1r20fv40`.

Figure 1 shows a WDisG graph instance for a Twitter discussion obtained from the political domain using the `fl1r20fv40` weighting scheme. Each tweet is represented as a node and each criticism answer as an edge. The root tweet of the discussion is labeled with 0 (the tweet that starts the discussion) and the other nodes are labeled with consecutive identifiers according to the temporal generation order of the tweets in the social network. The nodes that appear disconnected in the graph correspond to tweets that have participated in the discussion in response to other tweets, but have not been classified as criticisms answers (nodes 3 and 5). The discussion has a simple structure, possibly one of the most frequent in Twitter. A root tweet starts the discussion and some answers criticize it, and there are not many criticisms between non-root tweets. The discussion contains 13 tweets and 14 criticisms answers. Nodes are colored

[3] `http://www2.compute.dtu.dk/~faan/data/AFINN.zip`

in *blue scale*, where the darkness of the color is directly proportional to its weight with respect to the maximum value in the discussion.

Since our reasoning model defines the set of socially accepted tweets of a discussion as a set of tweets without effective conflicts, the solution is mainly defined by tweets that generate or receive criticism of other tweets and it is presented in next section.

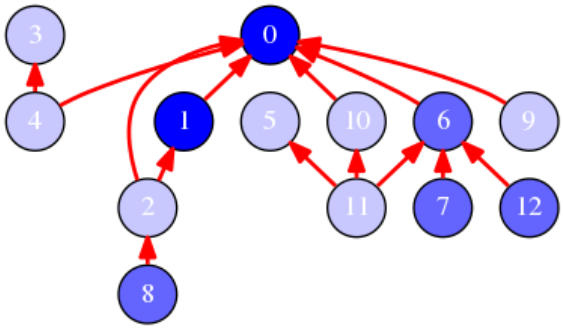

Fig. 1. WDisG graph instance.

3 An argumentation-based reasoning model

Once we have introduced our formal representation model of Twitter discussions with criticism relationships between pairs of tweets, the next key component is the definition of the reasoning model used to obtain the set of socially accepted tweets. To this end, we use a valued abstract argumentation framework [2] for modeling the weighted argumentation problem associated with a WDisG graph and ideal semantics [8] for defining the solution (the set of socially accepted tweets).

A valued argumentation framework (VAF) is a tuple $\langle A, attacks, R, Val, Valpref \rangle$ where A is a set of arguments, *attacks* is an irreflexive binary relation on A, R is a nonempty set of values, *Val* is a valuation function $Val : A \rightarrow R$ that assigns to each argument in A a weight value in R, and $Valpref \subseteq R \times R$ is a preference relation on R (transitive, irreflexive and asymmetric), reflecting the value preferences of arguments.

Given a Twitter discussion Γ and its WDisG graph $G = \langle T, E, \mathbb{N}, \texttt{fl1r20fv40} \rangle$ based on the weighting scheme $\texttt{fl1r20fv40} : T \rightarrow \mathbb{N}$, the VAF for G is the tuple $\mathcal{F} = \langle T, E, \mathbb{N}, \texttt{fl1r20fv40}, \text{¿} \rangle$, where each tweet in T results in an argument of $\mathcal{F}$, each criticism answer in E results in an attack between the arguments of $\mathcal{F}$, $\mathbb{N}$ is the set of valuations or weights of the arguments, the weighting scheme `fl1r20fv40` is the weighting valuation function of $\mathcal{F}$, and the order relation $>$ of $\mathbb{N}$ is the preference relation between the valuations or weights

of the arguments; i.e. a tweet t_2 is more valued than a tweet t_1 whenever $\texttt{fl1r20fv40}(t_2) > \texttt{fl1r20fv40}(t_1)$.

Then, a *defeat* relation (or effective attack relation) between tweets (arguments) is defined as follows: $\mathit{defeats} = \{(t_1, t_2) \in E \mid \texttt{fl1r20fv40}(t_2) \not> \texttt{fl1r20fv40}(t_1)\}$.

Moreover, a set of tweets (arguments) $S \subseteq T$ is *conflict-free* if for all $t_1, t_2 \in S, (t_1, t_2) \notin \mathit{defeats}$, and a conflict-free set of tweets $S \subseteq T$ is *maximally admissible* if for all $t_1 \notin S$, $S \cup \{t_1\}$ is not conflict-free and, for all $t_2 \in S$, if $(t_1, t_2) \in \mathit{defeats}$ then there exists $t_3 \in S$ such that $(t_3, t_1) \in \mathit{defeats}$. Finally, the *set of socially accepted tweets* of Γ, referred as the *solution* of Γ, is computed as the largest admissible conflict-free set of tweets $S \subseteq T$ in the intersection of all maximally admissible conflict-free sets. Remark that the tweets of a discussion that do not generate nor receive criticism, are always part of the solution.

Figure 2 shows the VAF solution for the WDisG graph instance of Figure 1. The nodes colored in blue are the tweets in the solution and the nodes colored in gray are the rejected tweets, where the darkness of the color is directly proportional to its weight. According to the `fl1r20fv40` valuation function, the tweets of the discussion are stratified in three levels denoting their relevance in the discussion. For each level, we find the following sets of tweets: level 0: $\{3, 2, 4, 5, 9, 10, 11\}$, level 1: $\{6, 7, 8, 12\}$, level 2: $\{0, 1\}$, being level 0 the lowest level and level 2 the highest one. The solution contains 7 tweets (Tweets 1, 4, 7, 8, 9, 11 and 12) of the 13 tweets of the discussion, and 6 tweets are rejected (Tweets 0, 2, 3, 5, 6 and 10). On the one hand, Tweet 1 defeats the root tweet, since Tweet 1 attacks the root tweet and both have the same weight. The same happens with Tweets 3, 5, 6 and 10, since Tweet 4 attacks Tweet 3, Tweet 11 attacks Tweets 5 and 10, and Tweets 7 and 12 attack Tweet 6. On the other hand, Tweet 8 defeats Tweet 2, since Tweet 8 attacks Tweet 2 and Tweet 8 is heavier than Tweet 2. Finally, Tweets 1, 4, 7, 8, 9, 11 and 12 are accepted since they do not have any defeater in the solution. Notice that in this discussion with high controversy around Tweets 0 and 6 (with a high number of attacks), we end up rejecting both tweets due to the weight they get during the discussion.

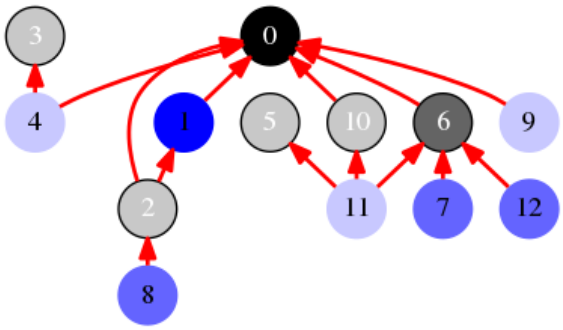

Fig. 2. WDisG graph solution.

In [1] we implemented a reasoning system for computing the VAF solution of a WDisG graph based on the algorithm for computing the ideal extension for an argumentation framework presented in [9], but adapting it to work with valued arguments. Regarding the implementation we used an approach based on Answer Set Programming (ASP) available in the argumentation system ASPARTIX [10], but we extended it to work with VAFs, as the current implementation in ASPARTIX only works with non-valued arguments. To develop such extension we modified the manifold ASP program explained in [11] incorporating:

- the *valuation function* for arguments,
- the *preference relation* between argument valuations and
- the *defeat relation* relation between arguments (effective attack).

Now in this work, we define a distributed strategy for implementing the underlying reasoning algorithm for computing the ideal extension for a VAF. The algorithm takes a WDisG graph of a Twitter discussion and outputs the set of accepted tweets based on the valuation function, the preference relation and the computation of the largest admissible conflict-free set of tweets according to the defeat relation.

4 Distributed solution computation

We design a distributed strategy for computing the solution of a WDisG graph using the distributed model of computation of Pregel [13]. This model is appropriate for our problem, because the input for a Pregel algorithm is a directed graph, where the nodes can be in different states, and the goal of a distributed algorithm in Pregel is to compute the state of each node based on the state of the nodes' neighbors. Any Pregel algorithm starts initializing each node to some initial state. Then, the distributed computation follows a sequence of *supersteps* separated by global synchronization points until the algorithm finishes a point where every node is happy with its current state. This computation model is actually inspired by Valiant's Bulk Synchronous Parallel model [16].

Within each superstep the nodes compute their state in parallel, executing a specific function that computes the new state of the node taking into account the possible messages sent by the nodes' neighbors in the *previous* superstep. Then, as a byproduct of the computation of the node state, some messages may be sent to some of the nodes' neighbors, that they will be processed in the next superstep. The idea is that the messages are used by nodes to indicate some change in their state, that could have some influence on the state of their neighbors in the next superstep. The superstep finishes when every node has computed its state and has sent the necessary messages to its neighbors.

In the computation model of Pregel, the input can be any directed graph. However, the algorithm we present here works only with discussion graphs that are acyclic, such that it allows us to solve discussions of big size with an efficient polynomial time distributed algorithm, where the state of each node depends only on the state of its attacking nodes. Observe that in the case of Twitter,

where a tweet only answers previous tweets, the discussion graphs are always acyclic, so restricting the algorithm to consider only acyclic discussion graphs is not a real restriction for Twitter.

For the distributed computation of the solution (ideal extension) for a discussion acyclic graph, we propose a Pregel algorithm where each node can be in two states: accepted or not accepted (rejected), but a node also stores a defeaters count, to keep track of the number of *accepted defeaters* the node has, to actually compute its acceptance state. Initially, every node starts in the rejected state and with its defeaters counter equal to zero.

Algorithm 1 Compute the acceptance state for a node a

```
1: procedure a.CompAcceptance(i) ▷ Update acceptance state of a at superstep i
2:     a.storeCurrentAcceptanceState()
3:     for all msg ∈ a.received(i-1) do                ▷ Check defeaters count update
4:         if (msg.type == attacker) then a.updateDefeaters( msg.value )
5:     a.updateAcceptanceState()
6:     if ( a.StateChanged() ) then        ▷ a changed to accepted or to not accepted
7:         sendToAllNeighbors( msg( attacker, (a.weight(), a.isAccepted() ) ) )
8:     else
9:         a.VoteToHalt()                  ▷ Vote to finish distributed computation
```

Algorithm 1 shows the pseudocode of the function used by each node a to compute its state in a superstep i. [4] It works as follows. In the superstep i, a node a checks if its state has to be changed after updating its defeaters counter. The node a updates its defeaters counter based on the received messages (from the previous superstep $i-1$) from any nodes v connected with a with an incoming edge (v, a). These messages are processed in the for loop of lines 3 and 4. There are two possible changes that every message can produce on the defeaters counter of a. On the one hand, if in the previous superstep a node v, and such that (v, a) is an edge of the graph, changed its state to accepted then v sent a message to a of type *attacker* and with value $(v.weight(), +1)$, indicating to a that its defeaters counter should be increased by one but only if v is a defeater of a (if $v.weight() \geq a.weight()$). On the other hand, such node v could instead have changed its state to not accepted, so in that case the message sent to a will be also of type *attacker* but with value $(v.weight(), -1)$, indicating that its defeaters counter should be decreased by one if v is a defeater of a. The possible addition or subtraction to its defeaters counter, caused by a message msg sent by a node v is checked (and performed when needed) by calling the function $a.updateDefeaters(msg.value)$, where the value has the format indicated previously. Then, after updating the defeaters counter the state

[4] The pseudocode is written using object oriented notation, as the Pregel API is written in C++. However, our actual implementation is based on the Pregel implementation found on the Spark distributed programming framework, graphX, that is written in Scala.

of a is updated calling the function $a.updateAcceptanceState()$, that checks if the updated defeaters counter is greater than zero. Finally, we call the function $a.StateChanged()$ (in line 6) to check whether the state of a changed (from not accepted to accepted or viceversa). If it changed, a sends an *attacker* message to all its outgoing neighbors with value $(a.weight(), a.isAccepted())$, where the function $a.isAccepted()$ will return either +1 or -1 depending on whether the state of a is accepted or not accepted. These sent messages will be processed by the outgoing neighbors of a when the next superstep begins. If the state of a did not change, then a votes towards finishing the distributed computation of the solution (line 9), but only when all the nodes agree to finish, in a same superstep, will the algorithm finish.

Consider the execution of the algorithm when solving the example discussion we have presented in Section 2, and whose solution is shown on Figure 2:

1. Before the first superstep, all the nodes start in the not accepted state and with defeaters counter equal to zero, so in the first superstep, as there are no incoming messages for any node, all of them will change their state to accepted. Then, any node with outgoing edges (all except 3 and 0), sends a message to its attacked nodes with value $(node.weight(), +1)$. Observe that nodes $4, 7, 8, 9, 11$ and 12 will remain accepted for the rest of the execution of the algorithm, as they will never receive any messages.
2. In the second superstep, the received messages produce that nodes $0, 1, 2, 3, 5, 6$ and 10 change to not accepted, as they receive messages from attacking accepted nodes that defeat them, so each one of these nodes (except 0 and 3) will send a message with value $(node.weight(), -1)$. Nodes $2, 3, 5, 6$ and 10 will remain not accepted for the rest of the execution of the algorithm.
3. In the third superstep, the messages sent in the previous superstep produce that nodes 0 and 1 change to accepted, and so node 1 will send a message to node 0 with value $(1.weight(), +1)$. Node 1 will remain accepted for the rest of the execution.
4. Finally, in the fourth superstep the message received by node 0 from 1, as 1 is a defeater for 0, will change the state of 0 to rejected, being this its final state and the end of the execution of the algorithm as no more messages are sent by any nodes.

Observe that the number of supersteps is equal to the maximum path length of the discussion graph, and although one can think about variants of this algorithm where less messages are sent, it seems that in the worst case it is not possible the have less supersteps than the maximum path length of the discussion graph. So, for typical Twitter discussions, where one has many branches in the discussion graph but not too deep, this algorithm may solve big discussions.

We have started to evaluate the performance of this algorithm on real Twitter discussions of different sizes, working with a small Spark cluster with five computers working with Linux. A table with preliminary results for a test set of Twitter discussions can be found at the URL: `https://www.dropbox.com/s/u4yrz5an78d6dfw/lasttable.pdf?dl=0`.

5 Conclusions and future work

In this paper we introduce a distributed system for mining the set of globally accepted tweets of a Twitter discussion. We model discussions with a weighted argumentation graph, where each node denotes a tweet, each edge denotes a criticism relationship between a pair of tweets of the discussion and each node is attached with a weight, that denotes the social relevance of the corresponding tweet in the discussion and it is computed from some tweet's attributes, such as the number of followers of the author, the number of retweets and the number of favorites.

The set of accepted tweets is defined following an skeptical approach based on the ideal semantics of a valued argumentation framework. The ideal semantics for valued argumentation guarantees that the set of tweets in the solution is the maximal set of tweets that satisfies that it is consistent, in the sense that there are no defeaters among them, and that all of the tweets outside the solution are defeated by a tweet within the solution.

Our distributed strategy is based on the distributed computation model of Pregel which is appropriate for our problem, because the input for a Pregel algorithm is a directed graph, where the nodes can be in different states, and the goal is to compute the state of each node based on the state of the nodes' neighbors.

In our system each node (tweet) can be in two states, accepted or not accepted (rejected), and the algorithm works only with discussion graphs that are acyclic, such that it allows us to solve discussions of big size with an efficient polynomial time distributed algorithm, where the state of each node depends only on the state of its attacking nodes. As far as we know, the system is the first distributed implementation of an argumentative reasoning algorithm for social network analysis.

As future work, we plan to extend the representation model to consider support relationships between tweets and also to explore an efficient implementation of a distributed algorithm for bipartite graphs.

References

1. Alsinet, T., Argelich, J., Béjar, R., Fernández, C., Mateu, C., Planes, J.: Weighted argumentation for analysis of discussions in Twitter. Int. J. Approx. Reasoning 85, 21–35 (2017)
2. Bench-Capon, T.J.M.: Persuasion in practical argument using value-based argumentation frameworks. Journal of Logic and Computation 13(3), 429–448 (2003)
3. Bench-Capon, T.J.M., Dunne, P.E.: Argumentation in artificial intelligence. Artif. Intell. 171(10-15), 619–641 (2007)
4. Besnard, P., Hunter, A.: A logic-based theory of deductive arguments. Artif. Intell. 128(1-2), 203–235 (2001)
5. Bild, D.R., Liu, Y., Dick, R.P., Mao, Z.M., Wallach, D.S.: Aggregate characterization of user behavior in Twitter and analysis of the retweet graph. ACM Trans. Internet Techn. 15(1), 4:1–4:24 (2015)

6. Bird, S.: NLTK: the natural language toolkit. In: Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, ACL 2006, Sydney, Australia. pp. 17–21 (2006)
7. Dung, P.M.: On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. Artificial Intelligence 77(2), 321 – 357 (1995)
8. Dung, P.M., Mancarella, P., Toni, F.: Computing ideal sceptical argumentation. Artif. Intell. 171(10-15), 642–674 (2007)
9. Dunne, P.E.: The computational complexity of ideal semantics I: abstract argumentation frameworks. In: Proceedings of Computational Models of Argument, COMMA 2008, Toulouse, France. pp. 147–158 (2008)
10. Egly, U., Gaggl, S.A., Woltran, S.: Aspartix: Implementing argumentation frameworks using answer-set programming. In: Proceedings of the 24th International Conference on Logic Programming, ICLP 2008. pp. 734–738 (2008)
11. Faber, W., Woltran, S.: Manifold answer-set programs for meta-reasoning. In: Proceedings of Logic Programming and Nonmonotonic Reasoning, LPNMR 2009. pp. 115–128 (2009)
12. Hansen, L.K., Arvidsson, A., Nielsen, F.A., Colleoni, E., Etter, M.: Good friends, bad news - affect and virality in Twitter. In: International Workshop on Social Computing, Network, and Services (SocialComNet 2011) (2011)
13. Malewicz, G., Austern, M.H., Bik, A.J.C., Dehnert, J.C., Horn, I., Leiser, N., Czajkowski, G.: Pregel: a system for large-scale graph processing. In: Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2010. pp. 135–146 (2010)
14. Nielsen, F.A.: A new anew: Evaluation of a word list for sentiment analysis in microblogs. In: Proceedings of the ESWC2011 Workshop on 'Making Sense of Microposts'. pp. 93–98 (2011)
15. Rahwan, I., Simari, G.R.: Argumentation in Artificial Intelligence. Springer Publishing Company, 1st edn. (2009)
16. Valiant, L.G.: A bridging model for multi-core computing. J. Comput. Syst. Sci. 77(1), 154–166 (2011)