AI Communications 00 (20xx) 1–15
DOI 10.3233/AIC-140627
IOS Press

Integrating argumentation and sentiment analysis for mining opinions from Twitter

Kathrin Grosse [a], María P. González [b,c], Carlos I. Chesñevar [b,c,*] and Ana G. Maguitman [b,c]

[a] *Institut für Kognitionswissenschaft, Universität Osnabrück, Osnabrück, Germany*

[b] *Artificial Intelligence Research and Development Laboratory, Department of Computer Science and Engineering, Universidad Nacional del Sur, Av. Alem 1253, (8000) Bahía Blanca, Argentina*

[c] *Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET),* **???**, *Argentina*

Abstract. Social networks have grown exponentially in use and impact on the society as a whole. In particular, microblogging platforms such as Twitter have become important tools to assess public opinion on different issues. Recently, some approaches for assessing Twitter messages have been developed, identifying sentiments associated with relevant keywords or hashtags. However, such approaches have an important limitation, as they do not take into account contradictory and potentially inconsistent information which might emerge from relevant messages. We contend that the information made available in Twitter can be useful to extract a particular version of arguments (called "opinions" in our formalization) which emerge bottom-up from the social interaction associated with such messages. In this paper we present a novel framework which allows to mine opinions from Twitter based on incrementally generated queries. As a result, we will be able to obtain an "opinion tree", rooted in the first original query. Distinguished, conflicting elements in an opinion tree lead to so-called "conflict trees", which resemble dialectical trees as those used traditionally in defeasible argumentation.[1]

Keywords: Argumentation, opinion mining, social media

1. Introduction and motivations

Social networks have grown exponentially in use and impact on the society as a whole, aiming at different communities and providing differentiated services. In particular, microblogging has become a very popular communication tool among Internet users, being Twitter[2] by far the most widespread microblogging platform. Twitter, created in 2006, enables its users to send and read text-based posts of up to 140 characters, known as "tweets". It has grown into a technology which allows to assess public opinion on different issues. Thus, for example, nowadays it is common to read newspaper articles referring to the impact of political decisions measured by their associated positive or negative comments in Twitter. Symmetrically, policy makers make public many of their claims and opinions, having an influence on the citizenry,[3] prompting their "tweeting back" with further comments and opinions. As the audience of microblogging platforms and services grows everyday, data from these sources can be used in opinion mining and sentiment analysis tasks. Indeed, the scientific study of emotions in opinions associated with a given topic has become relevant, consolidating a new area known as *sentiment analysis* [7,14], with application in several real-world problems such as e-government [4,5], stock market analysis [16] and tracking breaking events [6], among others.

As pointed out in recent research (e.g., [19,24]) microblogging platforms (in particular Twitter) offer a number of advantages for opinion mining. On the one hand, Twitter is used by different people to express their opinion about different topics, and thus they are a valuable source of people's opinions. Given the enormous number of text posts, the collected corpus can be arbitrarily large. On the other hand, Twitter's audience

[*] Corresponding author: carlos.ivan.chesnevar@gmail.com.

[1] This paper extends preliminary work [11] presented at the First Intl. Conf. on Agreement Technologies (AT 2012), held in Dubrovnik, Croatia, in Oct. 2012.

[2] www.twitter.com.

[3] For example, the current UK Prime Minster David Cameron, the current US President Barack Obama and the Pope Francis I can be followed on Twitter at @Number10gov, @BarackObama and @pontifex, respectively.

varies from regular users to celebrities, company representatives, politicians, and even country presidents. Therefore, it is possible to collect text posts of users from different social and interests groups. According to Merriam Webster online dictionary,[4] an *opinion* can be seen as: (a) a view, judgment, or appraisal formed in the mind about a particular matter; (b) belief stronger than impression and less strong than positive knowledge; a generally held view; (c) a formal expression of judgment or advice by an expert. Clearly, there is a natural link between opinion and argument. In many cases, opinions by themselves do not provide arguments, as they do not necessarily imply giving reasons or evidence for accepting a particular conclusion. However, from a meta-level perspective, policy makers devote much effort in analyzing the reasons underlying complex collections of opinions from the citizenry, as they indicate the willingness of the people to accept or reject some particular issue. A well-known example in this setting is the analysis of public opinion (e.g., through the quantitative measurement of opinion distributions through polls and the investigation of the internal relationships among the individual opinions that make up public opinion on an issue).

A fundamental need for policy makers is to back their decisions and agreements on reasons or opinions provided by citizens. They might even argue with other policy makers about why making a particular decision is advisable (e.g., "according to the last poll, 80% of the people are against the health system reform; therefore, the reform should not be carried out"). From this perspective, social networks like Twitter provide a fabulous knowledge base from which information could be collected and analyzed in order to enhance and partially automatize decision making processes. In particular, tweets have a rich structure, providing a number of record fields which allow to detect provenance of the tweet (author), number of re-tweets, followers, etc. We contend that the information made available from such tweets can be useful for modeling opinions which emerge bottom-up from the social interaction existing in Twitter.

In this article we present a novel framework which allows to mine opinions from Twitter based on incrementally generated queries. Given a query Q (corresponding to one or more keywords or hashtags), our approach allows to collect those distinguished tweets referring to Q, according to an aggregation criterion (also provided as an input). This collection of tweets will be called a *Twitter-based argument* A for Q, associated with a *prevailing sentiment* (computed on the basis of the tweets involved).[5] By expanding Q in different ways, we can obtain other, more specific arguments, which might be in conflict with A. These counter-arguments might be in turn in conflict with other more specific arguments. This will result in the characterization of an "opinion tree", rooted in the first original query. By considering distinguished nodes in an opinion tree we can define so-called "conflict trees", which resemble dialectical trees as those used traditionally in defeasible argumentation. We also provide theoretical results which account for a lattice-based characterization of our proposal, using equivalence classes to minimize the representation space to be analyzed when contrasting arguments.

The rest of the article is structured as follows. In Section 2 we present our proposal for characterizing Twitter-based arguments and their interrelationships. We will formalize the notion of *opinion tree*, which can be constructed from user queries, allowing to assess alternative opinions associated with incrementally generated queries. A high-level algorithm for computing opinion trees is presented, along with a case study to illustrate our proposal. Section 3 discusses the relationship emerging from opinions in conflict, modeled as a conflict tree. Section 4 generalizes previous results using superior lattices, for both opinion and conflict trees. Then, in Section 5, we present an example showing the practical use of our approach. The benefits of applying this mathematical approach are discussed in Section 6. Section 7 reviews related work, and finally Section 8 summarizes the conclusions.

2. Twitter-based argumentation framework: viewing aggregated tweets as arguments

In this section we will describe how different elements in Twitter can be captured under an argumentative perspective [3,20]. First we will characterize distinguished collections of tweets (obtained on the basis of a given query) as arguments with an associated prevailing sentiment. Such arguments will be called TB-arguments (Twitter-based arguments). Then, we will formalize interrelationships between TB-arguments, which lead to the notion of *opinion tree*.

[4] http://www.merriam-webster.com.

[5] Several software tools have been recently developed for such an association, such as www.sentiment140.com or tweetsentiments.com.

2.1. Formalizing aggregation of Twitter messages

Twitter messages (Tweets) are 140 character long, with a number of additional fields which help identify relevant information within a message (sender, number of retweets associated with the message, etc.). In particular, we will focus on the presence of *descriptors* which are either *hashtags* (words or phrases prefixed with the symbol #, a form of metadata tag) or terms that tend to occur often in the context of a given topic. Hashtags are used within IRC networks to identify groups and topics and in short messages on microblogging social networking services such as Twitter, identi.ca or Google+ (which may be tagged by including one or more hashtags with multiple words concatenated). Other good descriptors can be dynamically found by looking for terms that are frequently used in tweets related to the topic at hand. In the sequel we will assume that the term "descriptor" refers to either actual hashtags in Twitter or to relevant keywords found in tweets.

Definition 2.1 (Tweet. Twitter query). We define a tweet T as a bag (or multiset) of terms $\{t_1, t_2, \ldots, t_k\}$, where every $t_i \in T$ is a string. A *Twitter query* (or just query) is a non-empty set $Q = \{d_1, d_2, \ldots, d_k\}$ of descriptors, where every $d_i \in Q$ is a string.

In the analysis that follows, we will assume that a tweet is just a bag of words, not taking into account the actual order of terms in the tweet. Additionally, we assume that the set of all currently existing tweets corresponds to a snapshot of Twitter messages at a given fixed time, as the Twitter database is highly dynamic. In our approach, a query Q is any set of descriptors used for *filtering* some relevant tweets from the set of existing tweets $\mathfrak{Tweets}$ based on a given criterion C. In order to abstract away how such selection is performed, we will define an aggregation operator $\mathsf{Agg}_{\mathfrak{Tweets}}(Q, C)$. Formally:

Definition 2.2 (Tweet set. Aggregation operator). Let $\mathfrak{Tweets}$ be the set of all currently existing tweets. We will write $2^{\mathfrak{Tweets}}$ to denote the set of all possible subsets of $\mathfrak{Tweets}$. Any element in $2^{\mathfrak{Tweets}}$ will be called a *tweet set*. Given a query Q, and a criterion C, we will define an aggregation operator $\mathsf{Agg}_{\mathfrak{Tweets}}(Q, C)$ which returns an element (tweet set) in $2^{\mathfrak{Tweets}}$ based on Q and C.

The aggregation operator could be defined in several ways. For instance, suppose that C_1 is a criterion that indicates that only tweets posted between time *timestamp*$_1$ and *timestamp*$_2$ are to be selected. Then $\mathsf{Agg}_{\mathfrak{Tweets}}(Q, C_1) =_{\text{def}} \{T \in \mathfrak{Tweets}$ such that $Q \subseteq T$ and T satisfies $C_1\}$ will be the set of tweets that contain all the terms of query Q and have been posted in the time period [*timestamp*$_1$, *timestamp*$_2$]. Other examples of criteria that can be naturally applied are, for instance, requiring that those tweets T were retweeted more than n times, requiring that every user that posted tweets T has at least m followers, etc.

Note that for the same query Q, different alternative criteria $(C_1, C_2, \ldots, C_k)$ can lead to different distinguished elements in $2^{\mathfrak{Tweets}}$. As explained before, tweet sets can be associated with different feelings or sentiments. Even if in real life there may be a lot of emotions in tweets (such as anger, happiness, and so on), we will assume here that there is a distinguished set $\mathbb{S}$ of possible sentiments. Thus, given a query Q and a criterion C, we assume that the tweet set $\mathsf{Agg}_{\mathfrak{Tweets}}(Q, C)$ is associated with a prevailing sentiment in $\mathbb{S}$.[6] We will consider that some sentiments might convey different, possibly *conflicting* feelings or emotions (e.g., anger and happiness; boredom and excitement, etc.). As before, we will abstract away which are potentially conflicting sentiments as follows.

Definition 2.3 (*sent* and *conflict* mappings). Let $\mathbf{T} \in 2^{\mathfrak{Tweets}}$ be a tweet set, and let *sent* : $2^{\mathfrak{Tweets}} \to \mathbb{S}$ and *conflict* : $\mathbb{S} \to 2^{\mathbb{S}}$ be mappings. The sentiment *sent*($\mathbf{T}$) will be called the prevailing sentiment (or just sentiment) for $\mathbf{T}$. For any sentiment $s \in \mathbb{S}$, we will define *conflict*(s) as a subset of $\mathbb{S}$, such that: (a) $s \notin$ *conflict*(s) (a sentiment is not in conflict with itself); (b) for any $s' \in$ *conflict*(s), then $s \in$ *conflict*(s') (the notion of conflict is symmetrical). Given two sentiments s_1 and s_2, we will say that they are *in conflict* whenever $s_2 \in$ *conflict*(s_1). For simplicity, given a sentiment $s \in \mathbb{S}$, we will write $\overline{s}$ to denote any $s' \in$ *conflict*(s).

The previous elements will allow us to characterize the notion of TB-framework and TB-argument as follows:

[6] A possible range for $\mathbb{S}$ could be positive, negative and neutral (as done for example in platform sentiment140.com). In this platform, prevailing sentiments associated with a tweet set are expressed by percentages.

Definition 2.4 (TB-framework). A *Twitter-based argumentation framework* (or *TB-framework*) is a 5-tuple $(\mathfrak{Tweets}, C, \mathbb{S}, \mathit{sent}, \mathit{conflict})$, where $\mathfrak{Tweets}$ is the set of available tweets, C is a selection criterion, $\mathbb{S}$ is a non-empty set of possible sentiments and *sent* and *conflict* are sentiment prevailing and conflict mappings.

Definition 2.5 (TB-argument). Given a TB-framework $(\mathfrak{Tweets}, C, \mathbb{S}, \mathit{sent}, \mathit{conflict})$, a *Twitter-based argument* (or *TB-argument*) for a query Q is a 3-tuple $\langle \mathit{Arg}, Q, \mathit{Sent}\rangle$, where Arg is $\mathsf{Agg}_{\mathfrak{Tweets}}(Q, C)$ and Sent is $\mathit{sent}(\mathsf{Agg}_{\mathfrak{Tweets}}(Q, C))$.

Example 2.1. Consider a TB-framework $(\mathfrak{Tweets}, C, \mathbb{S}, \mathit{sent}, \mathit{conflict})$, where $Q = \{\text{"abortion"}, \text{"murder"}\}$, C is defined as "all $T \in \mathfrak{Tweets} \mid \mathit{timestamp}(T) \geqslant$ 2012-01-01T00:00:00", and $\mathbb{S} = \{\mathit{pos}, \mathit{neg}, \mathit{neutral}\}$, such that:

- $\mathit{conflict}(\mathit{pos}) =_{\text{def}} \{\mathit{neg}, \mathit{neutral}\}$,
- $\mathit{conflict}(\mathit{neg}) =_{\text{def}} \{\mathit{pos}, \mathit{neutral}\}$ and
- $\mathit{conflict}(\mathit{neutral}) =_{\text{def}} \{\mathit{pos}, \mathit{neg}\}$.

Then $\mathit{Arg} = \mathsf{Agg}_{\mathfrak{Tweets}}(Q, C)$ is the set of all possible tweets containing $\{\text{"abortion"}, \text{"murder"}\}$ that have been published since January 1, 2012. Suppose that $\mathit{sent}(\mathsf{Agg}_{\mathfrak{Tweets}}(Q, C)) = \mathit{negative}$. Then $\langle \mathit{Arg}, \{\text{"abortion"}, \text{"murder"}\}, \mathit{negative}\rangle$ is a TB-argument.

2.2. *Specificity in a TB-framework. Opinion trees*

In the previous section we have shown how to express arguments for queries associated with a given prevailing sentiment. Such arguments might be attacked by other arguments, which on their turn might be attacked, too. In argumentation theory, this leads to the notion of *dialectical analysis* [20], which can be associated with a tree-like structure in which arguments, counter-arguments, counter-counter-arguments, and so on, are taken into account. Our approach will be more generic, in the sense that for a given argument, the children nodes will correspond to more specific arguments that are not necessarily in conflict with the parent argument. Next we will formalize these notions.

A natural relation that arises between TB-arguments is derived from the inclusion relation between their associated queries. This is formalized by the following definition.

Definition 2.6 (Argument selectivity). Consider a TB-framework $(\mathfrak{Tweets}, C, \mathbb{S}, \mathit{sent}, \mathit{conflict})$ and let $\langle \mathit{Arg}_1, Q_1, \mathit{Sent}_1\rangle$ and $\langle \mathit{Arg}_2, Q_2, \mathit{Sent}_2\rangle$ be two TB-arguments. We say that $\langle \mathit{Arg}_2, Q_2, \mathit{Sent}_2\rangle$ is *more selective* than $\langle \mathit{Arg}_1, Q_1, \mathit{Sent}_1\rangle$, and we denote it $\langle \mathit{Arg}_2, Q_2, \mathit{Sent}_2\rangle \preceq_Q \langle \mathit{Arg}_1, Q_1, \mathit{Sent}_1\rangle$, if $Q_1 \subseteq Q_2$.

If two distinct queries Q_1 and Q_2 result in the same set of retrieved tweets, it is useful to identify Q_1 and Q_2 as equivalent queries. This gives rise to the following definition.

Definition 2.7 (Query equivalence). Let $(\mathfrak{Tweets}, C, \mathbb{S}, \mathit{sent}, \mathit{conflict})$ be a TB-framework. Given two queries Q_1 and Q_2, we will say that Q_1 *is equivalent* to Q_2 whenever $\mathsf{Agg}_{\mathfrak{Tweets}}(Q_2, C) = \mathsf{Agg}_{\mathfrak{Tweets}}(Q_1, C)$.

While it is clear that whenever $Q_1 \subseteq Q_2$, it will hold that $\mathsf{Agg}_{\mathfrak{Tweets}}(Q_2, C) \subseteq \mathsf{Agg}_{\mathfrak{Tweets}}(Q_1, C)$, it may be the case that for certain queries Q_1 and Q_2, $\mathsf{Agg}_{\mathfrak{Tweets}}(Q_2, C) \subseteq \mathsf{Agg}_{\mathfrak{Tweets}}(Q_1, C)$ but $Q_1 \nsubseteq Q_2$. In order to define a broader notion than query inclusion, we provide the following definition of query subsumption.

Definition 2.8 (Query subsumption). Given a TB-framework $(\mathfrak{Tweets}, C, \mathbb{S}, \mathit{sent}, \mathit{conflict})$ and two queries Q_1 and Q_2, we will say that Q_1 *subsumes* Q_2 whenever it holds that $\mathsf{Agg}_{\mathfrak{Tweets}}(Q_2, C) \subset \mathsf{Agg}_{\mathfrak{Tweets}}(Q_1, C)$.

Example 2.2. A query Q_1 formed by $\{\text{"abortion"}\}$ subsumes the query Q_2 formed by $\{\text{"abortion"}, \text{"murder"}\}$, as all the tweets that are returned by Q_2 will be part of the tweets returned by Q_1, but not the other way around.

Note that the *subsumption* relation is more general than the *inclusion* relation, since Q_1 *subsumes* Q_2 whenever $Q_1 \subset Q_2$ (as $\mathsf{Agg}_{\mathfrak{Tweets}}(Q_2, C) \subset \mathsf{Agg}_{\mathfrak{Tweets}}(Q_1, C)$). However, it is possible that Q_1 *subsumes* Q_2 even when $Q_1 \not\subset Q_2$.

Definition 2.9 (Argument specificity). Consider a TB-framework $(\mathfrak{Tweets}, C, \mathbb{S}, \mathit{sent}, \mathit{conflict})$ and let $\langle \mathit{Arg}_1, Q_1, \mathit{Sent}_1\rangle$ and $\langle \mathit{Arg}_2, Q_2, \mathit{Sent}_2\rangle$ be two TB-arguments. We say that $\langle \mathit{Arg}_2, Q_2, \mathit{Sent}_2\rangle$ is *strictly more specific* than $\langle \mathit{Arg}_1, Q_1, \mathit{Sent}_1\rangle$, and we denote it $\langle \mathit{Arg}_2, Q_2, \mathit{Sent}_2\rangle \prec \langle \mathit{Arg}_1, Q_1, \mathit{Sent}_1\rangle$, if Q_1 subsumes Q_2. We will write $\langle \mathit{Arg}_2, Q_2, \mathit{Sent}_2\rangle \preceq \langle \mathit{Arg}_1, Q_1, \mathit{Sent}_1\rangle$ when Q_1 subsumes Q_2 or Q_1 is equivalent to Q_2.

Suppose that a TB-argument supporting the query “abortion” is obtained, with a prevailing sentiment *negative*. If the original query Q is extended in some way into a new query Q' that is more specific than Q (i.e., $Q' = Q \cup \{d\}$), it could be the case that a TB-argument supporting Q' has a different (possibly conflicting) prevailing sentiment. For example, more specific opinions about abortion are related to other topics, like for example ethics, social problems or programs, religious issues, etc. To explore all possible relationships associated with TB-arguments returned for a specified query Q and criteria C, we can define an algorithm to construct an “opinion tree” recursively as follows:

(1) We start with a TB-argument A obtained from the original query Q (i.e., $\langle Arg, Q, Sent\rangle$), which will be the root of the tree.
(2) Next, we compute within A all relevant descriptors that might be used to “extend” Q, by adding a new element (*NewTerm*) to the query, obtaining $Q' = Q \cup \{NewTerm\}$.
(3) Then, a new argument for Q' is obtained, which will be associated with a subtree rooted in the original argument A.

The high-level algorithm can be seen in Fig. 1. As stated before, note that our approach to opinion trees is more generic than the one used for dialectical trees in argumentation (as done, e.g., in [10]), in the sense that for a given argument, the children nodes will correspond to more specific arguments that are not necessarily in conflict with the parent argument.

It is also easy to see that for any query Q, the algorithm *BuildOT* finishes in finite time: given that a tweet may not contain more than 140 characters, the number of contained descriptors is finite, and therefore the algorithm will eventually stop, providing an opinion tree as an output.

2.3. Case study

As discussed before, the algorithm shown in Fig. 1 allows to obtain an opinion tree from a given query Q, a criterion C, and the set $\mathfrak{Tweets}$ of all possible tweets. An additional parameter **R** allows us to specify the set of tweets to be considered when searching for a new descriptor. Initially, $\mathbf{R} = \mathfrak{Tweets}$. The cardinality of **R** determines the threshold associated with the depth of the tree.

Consider the query $Q =$ “abortion”, and a criterion $C = \{T \in \mathfrak{Tweets} \mid T$ was posted less than 48 hours ago$\}$. A root TB-argument is computed for Q, C and $\mathfrak{Tweets}$, obtaining an associated prevailing sentiment (*negative*). If $|\mathbf{R}|$ is above a given threshold value, the algorithm computes the most frequent word d in **R** whenever d is not already present in $Q \cup Stopwords$. The underlying idea is that any new descriptor used to extend the query associated with a node of the opinion tree should not appear in previous nodes at the same level (from left to right) or in ancestors of previous nodes at the same level. The set *Stopwords* will usually include terms such as *the*, *as*, *which*, etc. In our example, $d =$ “michigan”.[7] A new TB-argument can now be built for query $Q_{\text{new}} = \{$“abortion”$\} \cup \{$“michigan”$\}$, criterion C and the preserving sentiment calculated for the new subset of tweets $Tweets_{Q_{\text{new}}}$. In the recursive call, the most frequent word is calculated for this subset (obtaining the result “senate”), so that a new TB-argument for the query $\{$“abortion”, “michigan”, “senate”$\}$ is obtained, with a new associated prevailing sentiment. Note that within a particular instance of the recursive call, the REPEAT loop takes care of alternative ways of “extending” Q. This is accomplished by selecting a particular descriptor d from the set of tweets in **R**. The process is repeated until the threshold has been reached. At the end, the resulting opinion tree OT_Q is returned.

Figure 2 illustrates how the construction of an opinion tree for the query $Q = \{$“abortion”$\}$ looks like. Distinguished symbols (“+”, “−”, “=”) are used to denote positive, negative and neutral sentiments, respectively. Note that the original query Q has cardinality 1, and further levels in the opinion tree refer to incrementally extended queries (e.g., $\{$“abortion”, “michigan”$\}$, or $\{$“abortion”, “murder”$\}$). Leaves correspond to arguments associated with a query Q which cannot be further expanded, as the associated number of tweets is too small for any possible query $Q \cup W$. Furthermore, we can identify some subtrees in $\mathsf{OT}_{\{\text{“abortion”}\}}$ which consist of nodes which have *all the same sentiment*. In other words, further expanding a query into more complex queries does not change the prevailing sentiment associated with the root node. In other cases, expanding some queries results in a sentiment change (e.g., from $\{$“abortion”$\}$ into $\{$“abortion”, “option”$\}$ or $\{$“abortion”, “wish”$\}$). This situation will allow us to characterize *conflict trees*, in which we take into account opinions that attack each other, as discussed in the next section.

[7] This example was obtained from Twitter in December 2012, when Michigan legislature was debating several regulations on abortion practices.

```
ALGORITHM BuildOT
INPUT: $\mathfrak{Tweets}$, $\mathbf{R}$, $Q$, $C$
          {Initially, $\mathbf{R} = \mathfrak{Tweets}$. Each $T_i \in \mathfrak{Tweets}$ is represented as a multiset}
OUTPUT: Opinion Tree $\mathsf{OT}_Q$
          {opinion tree rooted in $Root_{\mathsf{OT}_Q}$}
$Root_{\mathsf{OT}_Q} := \langle Arg, Q, Sent \rangle$,
where $Arg$ is $\mathsf{Agg}_{\mathfrak{Tweets}}(Q, C)$ and $Sent$ is $sent(\mathsf{Agg}_{\mathfrak{Tweets}}(Q, C))$.
IF $|\mathbf{R}| >$ threshold {Cardinality of $\mathbf{R}$ determines maximum depth level}
THEN
  REPEAT
    $W := \{d \mid d$ is the most frequent word in $\biguplus_{T_i \in \mathbf{R}} T_i$
          such that $d \notin Q \cup Stopwords\}$[a]
    IF $W \neq \emptyset$
    THEN
      $Q_{\text{new}} := Q \cup W$
      $Tweets_{Q_{\text{new}}} := \mathsf{Agg}_{\mathfrak{Tweets}}(Q_{\text{new}}, C)$
      $\mathsf{OT}_{Q_{\text{new}}} := BuildOT(Tweets_{Q_{\text{new}}}, \mathbf{R}, Q_{\text{new}}, C)$
      $\mathsf{OT}_Q = PutSubtree(Root_{\mathsf{OT}_Q}, \mathsf{OT}_{Q_{\text{new}}})$
      $\mathbf{R} := \mathbf{R} - Tweets_{Q_{\text{new}}}$
  UNTIL $W = \emptyset$
RETURN $\mathsf{OT}_Q$
```

[a] For simplicity, we assume W is a singleton. In a more general case, a distinguished element from W could be selected according to some criterion (e.g., overall frequency in Twitter, etc.). The operation $\uplus$ is the union operation on multisets.

Fig. 1. High-level algorithm for computing an opinion trees OT_Q from Twitter.

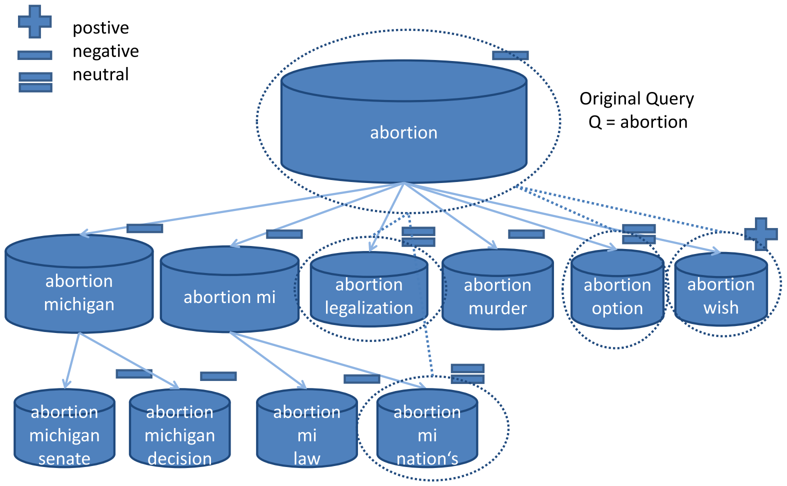

Fig. 2. Opinion tree based on query “abortion”. The associated conflict tree for the same query is shown in dotted lines. (Colors are visible in the online version of the article; http://dx.doi.org/10.3233/AIC-140627.)

3. Conflict trees

Next we will provide a formal definition of conflict between TB-arguments. Intuitively, a conflict will arise whenever two arguments for similar queries lead to conflicting sentiments assuming that the involved queries are related to each other by the subsumption relationship.

Definition 3.1 (Argument attack). Consider a TB-framework $(\mathfrak{Tweets}, C, \mathbb{S}, sent, conflict)$ and let $\langle Arg_1, Q_1, Sent_1 \rangle$ and $\langle Arg_2, Q_2, Sent_2 \rangle$ be two TB-arguments such that Q_1 subsumes Q_2, we say that $\langle Arg_2, Q_2, Sent_2\rangle$ *attacks* $\langle Arg_1, Q_1, Sent_1\rangle$ whenever $Sent_1$ and $Sent_2$ are in conflict.

Example 3.1. Consider query $Q_1 = \{\text{“abortion”}\}$ and query $Q_2 = \{\text{“abortion”}, \text{“option”}\}$ with associated TB-arguments $\langle Arg_1, Q_1, negative\rangle$ and $\langle Arg_2, Q_2, neutral\rangle$. Then $\langle Arg_2, Q_2, neutral\rangle$ attacks $\langle Arg_1, Q_1, negative\rangle$, and vice versa.

Note that in the previous situation, adding the descriptor “option” to the original query “abortion” involves a sentiment change. We will formalize this situation as follows:

Definition 3.2 (Sentiment-preserving descriptor. Sentiment-shifting descriptor). Let $\langle A_1, Q, Sent_1\rangle$ be a TB-argument. We say that a keyword or hashtag d is a *sentiment-preserving* (resp. *sentiment-shifting*) descriptor w.r.t. Q whenever there exists a TB-argument $\langle A_2, Q \cup \{d\}, Sent_2\rangle$ such that $Sent_1$ and $Sent_2$ are non-conflicting (resp. conflicting). TB-argument $\langle A_2, Q \cup \{d\}, Sent_2\rangle$ will be called *sentiment-preserving* (resp. *sentiment-shifting* argument).

Given a particular query Q, note that several alternative expansions (supersets of Q) can be identified. We are interested in identifying which is the *smallest* superset of Q which is associated with a sentiment-shifting argument. This gives rise to the following definition:

Definition 3.3 (Minimal-shift descriptor. Minimal-shifting relation). Let $(\mathfrak{Tweets}, C, \mathbb{S}, sent, conflict)$ be a TB-framework. Given two conflicting arguments $\langle Arg_1, Q_1, Sent\rangle$ and $\langle Arg_2, Q_2, \overline{Sent}\rangle$, we will say that Q_2 is a *minimal shift descriptor* w.r.t. Q_1 iff $\langle Arg_2, Q_2, \overline{Sent}\rangle$ is a sentiment-shifting argument w.r.t. Q_1 and $\nexists Q' \subset Q_2$ such that $\langle Arg', Q', \overline{Sent}\rangle$ is a sentiment-shifting argument w.r.t. Q_1.

We define a *minimal-shifting relation* “$\preceq_Q^{\min}$” as follows: $\langle Arg_1, Q_1, Sent_1\rangle \preceq_Q^{\min} \langle Arg_2, Q_2, Sent_2\rangle$ iff $\langle Arg_2, Q_2, Sent_2\rangle$ attacks $\langle Arg_1, Q_1, Sent_1\rangle$ and Q_2 is a minimal-shifting descriptor w.r.t. Q_1.

Definition 3.4 (Conflict tree). Let $(\mathfrak{Tweets}, C, \mathbb{S}, sent, conflict)$ be a TB-framework. Given a query Q, and its associated argument $\langle Arg, Q, Sent\rangle$ we will define a *conflict tree* for Q (denoted CT_Q) recursively as follows:

(1) If there is no $\langle Arg_i, Q_i, Sent_i\rangle$ such that $\langle Arg, Q, Sent\rangle \preceq_Q^{\min} \langle Arg_i, Q_i, Sent_i\rangle$, then CT_Q is a conflict tree consisting of a single node $\langle Arg, Q, Sent\rangle$.

(2) Let $\langle Arg_1, Q_1, Sent_1\rangle, \langle Arg_2, Q_2, Sent_2\rangle, \ldots, \langle Arg_k, Q_k, Sent_k\rangle$ be those arguments in $(\mathfrak{Args}, \mathfrak{Tweets}, C, \mathbb{S}, s)$ such that $\langle Arg, Q, Sent\rangle \preceq_Q^{\min} \langle Arg_i, Q_i, Sent_i\rangle$ (for $i = 1, \ldots, k$). Then CT_Q is a conflict tree consisting of $\langle Arg, Q, Sent\rangle$ as the root node and $\mathsf{CT}_{Q_1}, \ldots, \mathsf{CT}_{Q_k}$ are its immediate subtrees.

Intuitively, a conflict tree depicts all possible ways of extending the original query Q such that every extension (child node in the tree) corresponds to a sentiment change. Figure 2 illustrates how a conflict tree for the query $Q = \{\text{“abortion”}\}$ looks like, depicting nodes and arcs with dotted lines. Every node in the tree (except the root) is associated with a TB-argument which is a sentiment-shifting argument w.r.t. its parent. Leaves correspond to nodes for which no further sentiment shift can be found.

4. Generalizing opinion and conflict trees as superior lattices

Next we will show a formal lattice-based characterization of our approach and propose an effective procedure to compute conflict superior lattices, which can be regarded as a generalization of conflict trees. Superior lattices will account for a more generic view of opinion and conflict trees, identifying relevant sublattices based on an equivalence relation between TB-arguments. First we will review some background definitions to make our presentation self-contained.

Definition 4.1 (Partial order. Partially ordered set). A *partial order* is a binary relation “$\preceq$” over a set A which is reflexive, antisymmetric, and transitive, i.e., for all a, b and c in A, we have that (1) $a \preceq a$ (reflexivity); if $a \preceq b$ and $b \preceq a$ then $a = b$ (antisymmetry); if $a \preceq b$ and $b \preceq c$ then $a \preceq c$ (transitivity). A set with a partial order is called *partially ordered set* (or just ordered set).

Definition 4.2 (Cover relation). Given an ordered set $(A, \preceq)$, for two elements $a, b \in A$ we use $a \prec b$ to specify that $a \preceq b$ and $a \neq b$. Let $(A, \preceq)$ be an ordered set. Then for any $a, b \in A$ we say that a *covers* b if $b \prec a$ and there is no $c \in A$ such that $b \prec c \prec a$.

Definition 4.3 (Tree order). An ordered set $(A, \preceq)$ is a *tree* if (1) there is a unique $a \in A$ such that $b \preceq a$ for all $b \in A$, and (2) for all $a, b, c \in A$, if b covers a and c covers a, then $b = c$.

Definition 4.4 (Superior lattice. Inferior lattice. Lattice). Let $(A, \preceq)$ be an ordered set. Then for any $a, b \in A$, we will say that $c \in A$ is the *least upper bound* of a and b (also called the *join* of a and b), denoted $c = a \vee b$, whenever (i) $a \preceq c$ and $b \preceq c$; (ii) if for $x \in A$, it holds that $a \preceq x$ and $b \preceq x$, then $c \preceq x$. An ordered set $(A, \preceq)$ is a *superior lattice* whenever for any pair of elements $a, b \in A$ there is a least upper bound element in A. The notions of *greatest lower bound* (or *meet*), denoted $c = a \wedge b$, and *inferior lattice* are defined analogously as the duals of the notions of *least upper bound* and *superior lattice*. An ordered set $(A, \preceq)$ that is both a *superior lattice* and an *inferior lattice* is called a *lattice*.

Definition 4.5 (Join-homomorphism. Meet-homomorphism. Lattice homomorphism). The mapping h from $(X, \preceq)$ to $(Y, \preceq)$ is a *join-homomorphism* provided that for any $a, b \in x$, $h(a \vee b) = h(a) \vee h(b)$. It is also said that "$h$ preserves joins". The notion of *meet-homomorphism* is defined analogously as the dual of the notion of *join-homomorphism*. The mapping h is a *lattice homomorphism* if it is both a join-homomorphism and a meet-homomorphism.

In the rest of this section we will show that the above definitions provide a solid mathematical foundation for the study of TB-arguments. Note, in the first place, that the ordered set $(2^{\mathfrak{Tweets}}, \subseteq)$ is a lattice, as is the case for any power set of a given set, ordered by inclusion. The join is given by the union and the meet by the intersection of the subsets. More interestingly, it can be shown that for any query Q, the resulting opinion tree OT_Q associated with a query Q defines a tree order (see Definition 4.3).

Lemma 4.1. *Let Q be a query and let OT_Q be an opinion tree for Q in a TB-framework $(\mathfrak{Tweets}, C, \mathbb{S}, \mathit{sent}, \mathit{conflict})$. Then $(\mathsf{OT}_Q, \preceq_Q)$ defines a tree order.*

Proof. In order to prove that $(\mathsf{OT}_Q, \preceq_Q)$ is a tree order, we first need to prove that $(\mathsf{OT}_Q, \preceq_Q)$ is an ordered set. This is straightforward since the "$\preceq_Q$" relation is defined in terms of the "$\supseteq$" relation as follows. $\langle \mathit{Arg}_1, Q_1, \mathit{Sent}_1\rangle \preceq_Q \langle \mathit{Arg}_2, Q_2, \mathit{Sent}_2\rangle$, if and only if $Q_1 \supseteq Q_2$. Given that the "$\supseteq$" relation defines a partial order on the set of queries, it is clear that $(\mathsf{OT}_Q, \preceq_Q)$ is an ordered set.

To complete the proof we need to show that (1) there is a unique TB-argument $\langle \mathit{Arg}, Q, \mathit{Sent}\rangle$ such that $\langle \mathit{Arg}', Q', \mathit{Sent}'\rangle \preceq_Q \langle \mathit{Arg}, Q, \mathit{Sent}\rangle$ for all $\langle \mathit{Arg}', Q', \mathit{Sent}'\rangle$ in OT_Q, and (2) for all TB-arguments $\langle \mathit{Arg}_1, Q_1, \mathit{Sent}_1\rangle$, $\langle \mathit{Arg}_2, Q_2, \mathit{Sent}_2\rangle$, $\langle \mathit{Arg}_3, Q_3, \mathit{Sent}_3\rangle$ in OT_Q, if $\langle \mathit{Arg}_2, Q_2, \mathit{Sent}_2\rangle$ covers $\langle \mathit{Arg}_1, Q_1, \mathit{Sent}_1\rangle$ and $\langle \mathit{Arg}_3, Q_3, \mathit{Sent}_3\rangle$ covers $\langle \mathit{Arg}_1, Q_1, \mathit{Sent}_1\rangle$, then $\langle \mathit{Arg}_2, Q_2, \mathit{Sent}_2\rangle = \langle \mathit{Arg}_3, Q_3, \mathit{Sent}_3\rangle$. Part (1) follows from the construction of the opinion tree in algorithm *BuildOT* by taking $\langle \mathit{Arg}, Q, \mathit{Sent}\rangle$ as $\mathit{Root}_{\mathsf{OT}_Q}$. Then it is clear that $\mathit{Root}_{\mathsf{OT}_Q}$ is the only TB-argument such that $\langle \mathit{Arg}', Q', \mathit{Sent}'\rangle \preceq_Q \mathit{Root}_{\mathsf{OT}_Q}$ for all $\langle \mathit{Arg}', Q', \mathit{Sent}'\rangle$ in OT_Q. In order to prove (2), assume $\langle \mathit{Arg}_2, Q_2, \mathit{Sent}_2\rangle$ covers $\langle \mathit{Arg}_1, Q_1, \mathit{Sent}_1\rangle$ and $\langle \mathit{Arg}_3, Q_3, \mathit{Sent}_3\rangle$ covers $\langle \mathit{Arg}_1, Q_1, \mathit{Sent}_1\rangle$. According to the *BuildOT* algorithm, this means that $Q_1 = Q_2 \cup \{d\}$ and $Q_1 = Q_3 \cup \{d'\}$, with $d \notin Q_2$ and $d' \notin Q_3$. The descriptor selection mechanism for extending queries implemented in the algorithm guarantees that any query is extended by selecting a descriptor not appearing already as part of a query in a previous node at the same level (from left to right) or in ancestors of previous nodes at the same level. Assume, without loss of generality, that Q_2 is a predecessor of Q_3. Then, according to the restriction on the descriptor selection mechanism mentioned above, we can conclude that $d' \notin Q_2$. As a result it must be the case (from $Q_1 = Q_2 \cup \{d\}$, $Q_1 = Q_3 \cup \{d'\}$ and $d' \notin Q_2$) that $Q_2 = Q_3$ and therefore $\langle \mathit{Arg}_2, Q_2, \mathit{Sent}_2\rangle = \langle \mathit{Arg}_3, Q_3, \mathit{Sent}_3\rangle$. This concludes the proof. □

Once the opinion tree OT_Q is built, it is possible to identify and merge equivalent TB-arguments. The equivalence relation between TB-arguments is induced by the equivalence of the corresponding queries, i.e., $\langle \mathit{Arg}_i, Q_i, \mathit{Sent}_i\rangle \sim_{\text{query}} \langle \mathit{Arg}_j, Q_j, \mathit{Sent}_j\rangle$ if and only if Q_i is equivalent to Q_j (see Definition 2.7). The algorithm for merging equivalent TB-arguments is presented in Fig. 3. As we will see later, this algorithm returns an opinion superior lattice as a result. Figure 4 illustrates the application of this algorithm on an opinion tree. For the sake of simplicity, we use labels $Q_1, \ldots, Q_{27}$ to represent $\langle \mathit{Arg}_1, Q_1, \mathit{Sent}_1\rangle, \ldots, \langle \mathit{Arg}_{27}, Q_{27}, \mathit{Sent}_{27}\rangle$. On the left-hand side of this figure we can see an opinion tree as a tree order $(\mathsf{OT}_Q, \preceq_Q)$. Note that each element in OT_Q is of the form $\langle \mathit{Arg}_i, Q_i, \mathit{Sent}_i\rangle$, while the order relation "$\preceq_Q$" is defined as $\langle \mathit{Arg}_1, Q_1, \mathit{Sent}_1\rangle \preceq \langle \mathit{Arg}_2, Q_2, \mathit{Sent}_2\rangle$ if and only if $Q_2 \subseteq Q_1$ (see Definition 2.6). In this figure we have indicated that some queries are equivalent.

```
ALGORITHM BuildOL
INPUT:  Opinion Tree $\mathsf{OT}_Q$
OUTPUT:  Opinion (Superior) Lattice $\mathsf{OL}_Q$
             {opinion (superior) lattice derived from $\mathsf{OT}_Q$}
For each pair of TB-arguments $\langle Arg_i, Q_i, Sent_i\rangle$ and $\langle Arg_j, Q_j, Sent_j\rangle$ in
  the Opinion Tree $\mathsf{OT}_Q$, we define the $\sim_{\text{query}}$ equivalence relation as follows:
  $\langle Arg_i, Q_i, Sent_i\rangle \sim_{\text{query}} \langle Arg_j, Q_j, Sent_j\rangle$ if and only if $Q_i$ is equivalent to $Q_j$
Define $\mathsf{OL}_Q$ as the quotient (superior) lattice of $\mathsf{OT}_Q$ modulo $\sim_{\text{query}}$
RETURN $\mathsf{OL}_Q$
```

Fig. 3. High-level algorithm for computing opinion (superior) lattices from opinion trees.

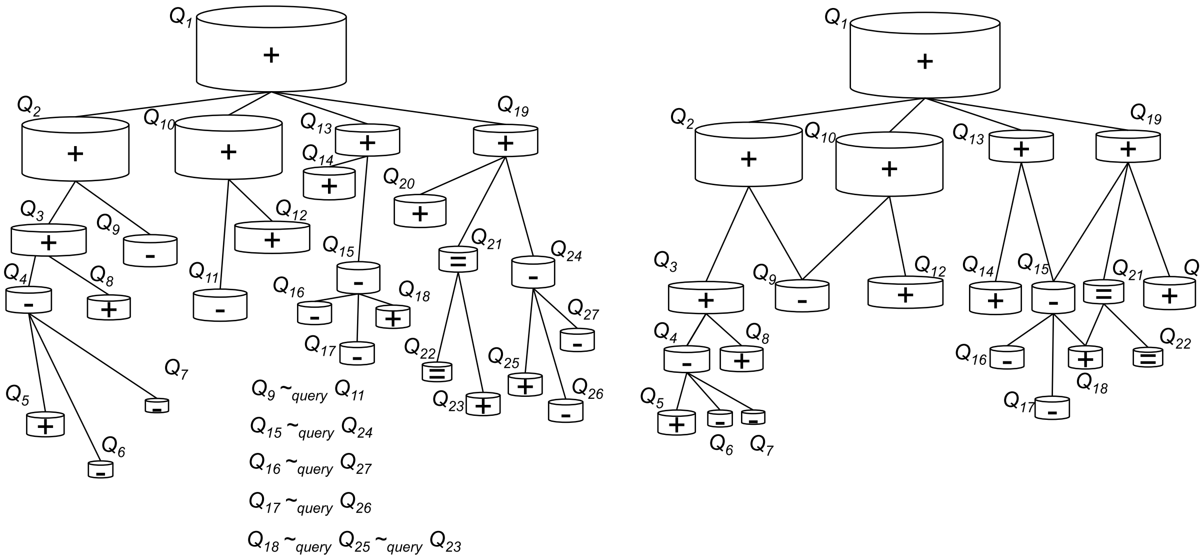
$Q_9 \sim_{query} Q_{11}$
$Q_{15} \sim_{query} Q_{24}$
$Q_{16} \sim_{query} Q_{27}$
$Q_{17} \sim_{query} Q_{26}$
$Q_{18} \sim_{query} Q_{25} \sim_{query} Q_{23}$
Q_1 Q_2 Q_3 Q_4 Q_5 Q_6 Q_7 Q_8 Q_9 Q_{10} Q_{11} Q_{12} Q_{13} Q_{14} Q_{15} Q_{16} Q_{17} Q_{18} Q_{19} Q_{20} Q_{21} Q_{22} Q_{23} Q_{24} Q_{25} Q_{26} Q_{27}
+
-
=

Fig. 4. Opinion tree (*left*) and its corresponding opinion (superior) lattice (*right*).

As a consequence, based on the given algorithm, we can identify a quotient set, where each member is an equivalence class $[\langle Arg_i, Q_i, Sent_i\rangle]$ defined as the set $\{\langle Arg_j, Q_j, Sent_j\rangle \mid Q_j \text{ is equivalent to } Q_i\}$.

We show on the right-hand side of Fig. 4 the quotient set resulting from the given opinion tree. Note that this new set is a superior lattice (see Definition 4.4). In general, any opinion tree induces a superior lattice OL_Q, which we will refer to as opinion (superior) lattice. This is formally stated in the following lemma:

Lemma 4.2. *Let Q be a query and let OL_Q be the quotient set of* OT_Q *by the* query equivalence *relation. Then* $(\mathsf{OL}_Q, \preceq)$ *is a superior lattice.*

Proof. In order to prove that $(\mathsf{OL}_Q, \preceq)$ is a superior lattice we need to prove that for any pair of TB-argument classes $[\langle Arg_1, Q_1, Sent_1\rangle]$ and $[\langle Arg_2, Q_2, Sent_2\rangle]$ in OL_Q, $[\langle Arg_1, Q_1, Sent_1\rangle] \vee [\langle Arg_2, Q_2, Sent_2\rangle]$ is a TB-argument in OL_Q. Since OL_Q is the quotient set of OT_Q by the query equivalence relation, and by Lemma 4.1 OT_Q is a tree order, we have that $\langle Arg_1, Q_1, Sent_1\rangle \vee \langle Arg_2, Q_2, Sent_2\rangle$ is the most specific common ancestor of $\langle Arg_1, Q_1, Sent_1\rangle$ and $\langle Arg_2, Q_2, Sent_2\rangle$. Then, for the given pair of TB-argument classes, we take $[\langle Arg_1, Q_1, Sent_1\rangle] \vee [\langle Arg_2, Q_2, Sent_2\rangle] = [\langle Arg_1, Q_1, Sent_1\rangle \vee \langle Arg_2, Q_2, Sent_2\rangle]$. This concludes the proof. □

Although an opinion (superior) lattice is typically more compact than an opinion tree, we might be interested in finding in a computationally effective way the minimal structure that reflects all existing conflicts between opinions for a given query Q. In other words, we want to build a minimal superior lattice $(\mathsf{CL}_Q, \preceq)$ such that it is possible to define a join-homomorphism h (see Definition 4.5) from $(\mathsf{OL}_Q, \preceq)$ to $(\mathsf{CL}_Q, \preceq)$. In addition, we will require that if

$h(\langle \mathit{Arg}_i, Q_i, \mathit{Sent}_i\rangle) = \langle \mathit{Arg}_j, Q_j, \mathit{Sent}_j\rangle$ then Sent_i and Sent_j are non-conflicting. We will call CL_Q the conflict (superior) lattice for Q. By applying a partitioning algorithm it is possible to obtain a conflict (superior) lattice from any opinion (superior) lattice. The algorithm for computing conflict (superior) lattices for a given opinion (superior) lattice is presented in Fig. 5.

Figure 6 illustrates the transformation of an opinion (superior) lattice into a conflict (superior) $\langle \mathit{Arg}_1, Q_1, \mathit{Sent}_1\rangle, \ldots, \langle \mathit{Arg}_{27}, Q_{27}, \mathit{Sent}_{27}\rangle$. Initially, the 0-equivalent classes are computed based on the polarity of the sentiment associated with each TB-argument. Therefore, $\langle \mathit{Arg}_i, Q_i, \mathit{Sent}_i\rangle$ and $\langle \mathit{Arg}_j, Q_j, \mathit{Sent}_j\rangle$ are in the same 0-equivalent class if and only if $\mathit{Sent}_i = \mathit{Sent}_j$.

```
ALGORITHM BuildCL
INPUT:   Opinion (Superior) Lattice $\mathsf{OL}_Q$
OUTPUT:  Conflict (Superior) Lattice $\mathsf{CL}_Q$
             {conflict (superior) lattice derived from $\mathsf{OL}_Q$}
For each pair of TB-arguments $\langle Arg_i, Q_i, Sent_i\rangle$ and $\langle Arg_j, Q_j, Sent_j\rangle$ in
  the Opinion (Superior) Lattice $\mathsf{OL}_Q$, we define the $\sim_0$ equivalence relation as follows:
  $\langle Arg_i, Q_i, Sent_i\rangle \sim_0 \langle Arg_j, Q_j, Sent_j\rangle$ if and only if $Sent_i = Sent_j$
Compute the 0-equivalent classes based on the $\sim_0$ equivalence relation
$n = 0$
REPEAT
  $n = n + 1$
  Compute the $n$-equivalent classes as a refinement of the $(n-1)$-equivalent classes:
   $\langle Arg_i, Q_i, Sent_i\rangle \sim_n \langle Arg_j, Q_j, Sent_j\rangle$ if and only if
     (1) $\langle Arg_i, Q_i, Sent_i\rangle \sim_{n-1} \langle Arg_j, Q_j, Sent_j\rangle$, and
     (2) For all $\langle Arg_k, Q_k, Sent_k\rangle$ in $\mathsf{OL}_Q$
         $\langle Arg_i, Q_i, Sent_i\rangle \vee \langle Arg_k, Q_k, Sent_k\rangle \sim_{n-1} \langle Arg_j, Q_j, Sent_j\rangle \vee \langle Arg_k, Q_k, Sent_k\rangle$
UNTIL the $n$-equivalent classes are equal to the $(n-1)$-equivalent classes
Define $\mathsf{CL}_Q$ as the quotient (superior) lattice of $\mathsf{OL}_Q$ modulo $\sim_n$
RETURN $\mathsf{CL}_Q$
```

Fig. 5. High-level algorithm for computing conflict (superior) lattices from opinion (superior) lattices.

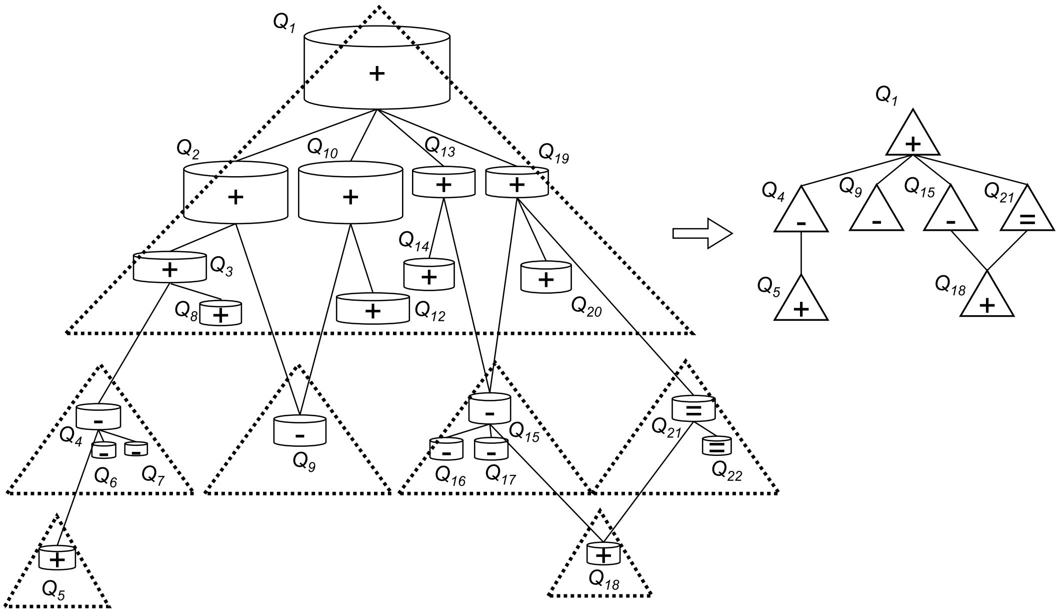

Fig. 6. From an opinion superior lattice (*left*) to a conflict superior lattice (*right*).

This results in the following classes:

0-equivalent classes:

$$\{Q_1, Q_2, Q_3, Q_5, Q_8,$$
$$Q_{10}, Q_{12}, Q_{13}, Q_{14}, Q_{18}, Q_{19}, Q_{20}\},$$
$$\{Q_4, Q_6, Q_7, Q_9, Q_{15}, Q_{16}, Q_{17}\},$$
$$\{Q_{21}, Q_{22}\}.$$

As specified in Algorithm *BuildCL*, the n-equivalent classes are computed as a refinement of the $(n-1)$-equivalent classes. This will be characterized as follows: we will say that two TB-arguments $\langle \mathit{Arg}_i, Q_i, \mathit{Sent}_i\rangle$ and $\langle \mathit{Arg}_j, Q_j, \mathit{Sent}_j\rangle$ are n-equivalent if and only if (1) $\langle \mathit{Arg}_i, Q_i, \mathit{Sent}_i\rangle$ and $\langle \mathit{Arg}_j, Q_j, \mathit{Sent}_j\rangle$ are $(n-1)$-equivalent, and (2) for every $\langle \mathit{Arg}_k, Q_k, \mathit{Sent}_k\rangle$ in OL_Q it holds that $\langle \mathit{Arg}_i, Q_i, \mathit{Sent}_i\rangle \vee \langle \mathit{Arg}_k, Q_k, \mathit{Sent}_k\rangle$ and $\langle \mathit{Arg}_j, Q_j, \mathit{Sent}_j\rangle \vee \langle \mathit{Arg}_k, Q_k, \mathit{Sent}_k\rangle$ are $(n-1)$-equivalent.

In order to compute the 1-equivalent classes, we take each pair of 0-equivalent TB-arguments Q_i and Q_j and verify whether $Q_i \vee Q_k$ is 0-equivalent to $Q_j \vee Q_k$ for all Q_k. If this is the case, then Q_i and Q_j remain in the same 1-equivalent class. Otherwise, these two TB-arguments are distinguishable and each one is included in a different 1-equivalent class.

To illustrate this, take for instance Q_3 and Q_{20}. Since for all Q_k, $Q_3 \vee Q_k$ is 0-equivalent to $Q_{20} \vee Q_k$, we can assert that Q_3 and Q_{20} are 1-equivalent (i.e., they are not distinguishable at this stage). On the other hand, take the pair Q_1 and Q_5. We have that $Q_1 \vee Q_6 = Q_1$ and $Q_5 \vee Q_6 = Q_4$. Furthermore, we know that Q_1 is not 0-equivalent to Q_4. Since there is at least a Q_k such that $Q_1 \vee Q_k$ is not 0-equivalent to $Q_5 \vee Q_k$, Q_1 and Q_5 are included in separate 1-equivalent classes (i.e., they are distinguishable at this stage). After performing a similar analysis with the remaining pairs of TB-arguments, we obtain the following classes:

1-equivalent classes:

$$\{Q_1, Q_2, Q_3, Q_8, Q_{10}, Q_{12}, Q_{13}, Q_{14}, Q_{19}, Q_{20}\},$$
$$\{Q_5\},$$
$$\{Q_{18}\},$$
$$\{Q_4, Q_6, Q_7\},$$
$$\{Q_9\},$$
$$\{Q_{15}, Q_{16}, Q_{17}\},$$
$$\{Q_{21}, Q_{22}\}.$$

If we attempt to compute the 2-equivalent classes, we found out that they are identical to the 1-equivalent classes. Therefore the process terminates and the conflict (superior) lattice shown on the right-hand side of Fig. 6 is returned. Note that in addition, it is possible to verify that the canonical mapping, that maps each element in OL_Q to its equivalence class in CL_Q, defines a join homomorphism.

5. Application

In this section, we show how our framework can be applied in a real-world situation by presenting a possible user scenario. The described scenario shows ways in which a policy maker could recognize opinions from mass deliberations of citizens expressing their views on “taxes”. The topic of taxes is typically a trendy one in Twitter, in particular among United States citizens commenting on current tax legislation or on tax changes proposals advocated by their government.

The topic of taxes can be analyzed form various perspectives. A possible perspective would be by looking at opinions that address the issue of taxes on property. A second perspective could focus on the topic of the IRS scandal. A third perspective is provided by analyzing how the health care reform affects taxes. Yet another perspective emerges from analyzing the new tax law, which imposes higher taxes on families earning over $250,000 a year, without changing the situation for the middle class.

The proposed tool facilitates the exploration of these various perspective by imposing a rich structure on a large set of unstructured tweets. In addition, it allows to easily recognize the polarity of each group of opinions (TB-arguments) as well as conflict relations between them. Figure 7 presents a conflict superior lattice for the query “taxes”. In this scenario, certain emerging TB-arguments could shed light on the general desires of citizens. For instance, the fact that the sentiment polarity of “taxes pay companies” is positive may be indicating that the general public expect companies to pay higher corporate taxes. In addition, the use of the tool to identify current topics, such as those associated with the queries “taxes irs scandal” or “taxes health” could greatly help decision and policy makers define priorities and better address citizens’ present-day concerns.

We have developed a Java prototype[8] as a beta version of a software tool for mining opinions from Twit-

[8] Available to download from http://cs.uns.edu.ar/~cic/decide2.0/twitter.zip.

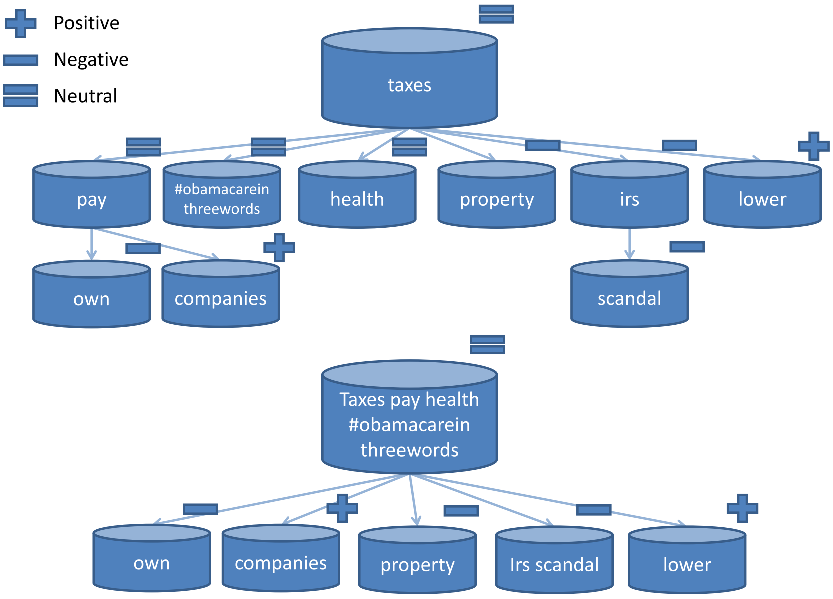

Fig. 7. Opinion tree and conflict lattice for the query "taxes". Results are simplified (4 nodes were left out). (Colors are visible in the online version of the article; http://dx.doi.org/10.3233/AIC-140627.)

ter. This prototype was used for the analysis of the abortion case (Section 2.3) and the previous tax example.

6. Discussion

Our mathematical characterization of opinion and conflict trees as superior lattices provides a natural foundation for the analysis of important concepts prevailing in argumentation theory. In particular, the use of conflict (superior) lattices to represent diverging arguments leads to the identification of the minimal structure that reflects the existing collective positions with respect to a topic of interest.

From the user viewpoint, conflict lattices are intended to provide the theoretical basis for developing an explorative tool in a decision making platform. Consider for example the opinion tree based on the query "abortion". By having the conflict tree at hand, an analyst[9] would be able to easily identify which are the terms or keywords that induce a sentiment shift when considering different tweets. For the case in Fig. 2, it can be noted that we get a conflict tree (which can be considered as particular case of conflict lattice). In a more general situation, conflict superior lattices provide a suitable mathematical structure for avoiding redundancies when considering attacks in conflict trees. For example, in Fig. 6(right), the analyst will be able to identify a single argument (Q18) which simultaneously attacks two other arguments (Q15 and Q21).

Argument specificity is a key notion in argumentation theory, as it is the first purely syntactic preference criterion proposed to compare arguments. In our framework, specificity can be associated with the "$\preceq$" relation identified in the resulting superior lattices. The use of minimal structures to represent conflicting views facilitates the identification of specificity relations as well as the recognition of relevant (or irrelevant) elements in the argumentation space, as it is formalized by the notions of sentiment-shifting descriptors (or sentiment-preserving descriptors). Similarly, the minimal-shift relation "$\preceq_Q^{\min}$" can be intuitively studied in the light of the resulting mathematical structures. It must be remarked that our dialectical analysis of TB-arguments aims at modeling the possible space of alternatives associated with different (incrementally more specific) queries. In contrast, the dialectical analysis in standard argumentation frameworks [3,20] aims at determining the ultimate status of a given argument at issue (in terms of some acceptability semantics).

It is important to mention that our analysis was done for the English language only. This is due to the fact that English is the *lingua franca* worldwide, being widely used in Twitter. In addition, most existing sen-

[9] Possible users could be, e.g., a journalist, a deputy analyzing a law proposal, etc.

timent analysis tools assume English as the underlying language. We are currently developing a sentiment analysis tool for the Spanish language, which will allow us to extend the capabilities of the system. We will also investigate the benefits of using a stemming algorithm for Spanish.

7. Related work

Our approach is inspired by recent research in integrating argumentation and social networks (e.g., [13, 23]). In the last years, there has been growing interest in assessing meaning to streams of data from microblogging services such as Twitter, as well as some recent research on using argumentation for social networks.

To the best of our knowledge, Torroni and Toni [23] were the first to combine social networks and argumentation in a unified approach, coining the term *bottom-up argumentation* for the grass-root approach to the problem of deploying computational argumentation in online systems. In this novel view, argumentation frameworks are obtained bottom-up starting from the users' comments, opinions and suggested links, with no top-down intervention of or interpretation by "argumentation engineers". As the authors point out "topics emerge, bottom-up, during the underlying process, possibly serendipitously". In contrast with that proposal, in this paper we generalize this view by identifying arguments automatically from Twitter messages, establishing as well conflict relationships in terms of sentiment analysis (and not specified at the meta-level using rules, as it is the case in [23]). This proposal was recently extended (see [8]), leading towards so-called "microdebates" to help organizing and confronting users' opinions in an automated way. A microdebate is a stream of tweets where users annotate their messages by using some special tags. In contrast with this approach, in our proposal we are not explicitly searching for debates containing arguments and counterarguments. Rather, different opinions emerge automatically based on collecting tweets associated with a particular topic (TB-arguments), and interrelationships among opinions are obtained on the basis of sentiment shifting/preserving descriptors.

In [1], Abbas and Sawamura formalize argument mining from the perspective of intelligent tutoring systems. In contrast with our approach, they rely on a relational database, and their aim is not related with identifying arguments underlying social networks as done in this paper. In [13], Leite and Martins introduce a novel extension to Dung's abstract argumentation model, called Social Abstract Argumentation. Their proposal aims at providing a formal framework for social networks and argumentation, incorporating social voting and defining a new class of semantics for the resulting frameworks. In contrast with our approach, the automatic extraction of arguments from social networks data is not considered (as done in this paper), nor the modeling of conflicts between arguments in terms of sentiment analysis. In [2], Amgoud and Serrurier propose a formal argumentation-based model for classification, which generalizes the well-known concept learning model based on version spaces [15]. The framework shares some structural similarities with our approach (as a lattice-based characterization is also involved when contrasting hypotheses). However, the aims of the two approaches are different, as our proposal is not focused on solving classification tasks in a machine learning sense.

A related research area is formal concept analysis [9], which is a method for deriving conceptual structures out of data. As done in our approach, the theory of partial orders is used to formally characterize these structures. However, it differs from our proposal in dealing with concepts rather than opinions and in not attempting to associate sentiments with the elements of the partial order. In addition, it does not deal with notions such as arguments, conflict and attack.

It must be remarked that the rise of social media such as blogs and social networks has fueled interest in sentiment analysis. With the proliferation of reviews, ratings, recommendations and other forms of online expression, online opinion has turned into a kind of virtual currency for businesses looking to market their products, identify new opportunities and manage their reputations. Several research teams in universities around the world currently focus on understanding the dynamics of sentiment in e-communities through sentiment analysis. The EU funded Cyberemotions consortium[10] was created in 2009 to better understand collective emotional phenomena in cyberspace, with the help of knowledge and methods from natural, social, and engineering sciences. Within this project, Thelwall et al. [21,22] carried out a number of experiments to assess the feasibility of sentiment analysis within social networks, with a particular focus on Twitter. In contrast with our approach, no opinion mining was considered in this context, nor the analysis of alternative opinions (as modeled by conflict trees in our proposal).

[10] http://www.cyberemotions.eu/.

8. Conclusions and future work

In this paper we have presented a novel approach which integrates argumentation theory and microblogging technologies, with a particular focus on Twitter. To the best of our knowledge, no other approach has been developed in a similar direction. We have also presented a definition of a Twitter-based argument for a query Q that considers as a support the bunch of tweets which are associated with Q according to a given criterion. For such an argument, we also define a prevailing sentiment, obtained in terms of sentiment analysis tools. This allowed us to characterize the notion of opinion tree, which can be recursively built by considering arguments associated with incrementally extended queries. We have implemented a prototype of our proposal as a proof of concept, which was used to compute the opinion tree for the case study presented in the paper.

We have also presented a theoretical setting for analyzing Twitter-based arguments, associating a superior lattice rooted in the initial argument for the first given query. Based on the notion of attack between arguments, we have established as well a refined order relationship between conflicting arguments. As a result, from every superior lattice associated with a given query Q, a *conflict tree* rooted in Q can be built, in which alternating opinions can be better contrasted. Given a node A (argument) associated with query Q' with a prevailing sentiment s, every children node for A in a conflict tree corresponds to an argument for a more specific query Q', which is in conflict with A as it is associated with a sentiment shift. Conflict trees allow us to explore the space of possible confronting opinions associated with a given opinion, using the specificity principle as traditionally used in argumentation for preferring arguments.

The prototype that we have implemented so far is intended to be used as a proof of concept. The development of a full-fledged software tool will require tackling several issues, mostly related with user-interface and usability aspects. We believe that before embarking on that stage, it is crucial to investigate and provide a full account of the functional capabilities of the proposed system, both based on a theoretical study and by validating its behavior through simulations with realistic data. Part of our future work will focus on improving the existing prototype, aiming at the deployment of a software tool for real-world users. As a basis for such deployment, visual tools for displaying and analyzing dialectical trees have been already developed for Defeasible Logic Programming [17]. We expect to use the underlying algorithms from this tool in our framework. Additionally, we expect to perform different experiments with hashtags associated with relevant topics, assessing the applicability of our approach in a real-world context.

Another future research avenue would be to take advantage of existing semantic information sources, such as dictionaries, topic directories or ontologies, to better explore query space, either by using synonyms of existing terms or other important terms for the domain under analysis. In addition, we anticipate that the proposed framework could be integrated with mechanisms that allow weighting TB-arguments based on different aspects, such as provenance of the tweets, number of associated tweets, opinion strength, etc. In particular, we are analysing the possibility of extending the current framework in order to consider ontologies for sentiment analysis of Twitter posts (as proposed in [12]). Finally, we are currently working on extending the current Twitter-based model to a more generic setting, in which opinions are collected from other social networks (such as Facebook).[11] Research in this direction is currently being pursued.

Acknowledgements

The authors would like to thank anonymous reviewers which helped improve the original version of this article. This research work was funded by Projects LACCIR R1211LAC004 (Microsoft Research, CONACyT and IDB), PIP 112-200801-02798, PIP 112-200901-00863 (CONICET, Argentina), PGI 24/ZN10, PGI 24/N006, PGI 24/N029 (SGCyT, UNS, Argentina) and Universidad Nacional del Sur.

References

[1] S. Abbas and H. Sawamura, Argument mining based on a structured database and its usage in an intelligent tutoring environment, *Knowl. Inf. Syst.* **30**(1) (2012), 213–246.

[2] L. Amgoud and M. Serrurier, Agents that argue and explain classifications, *Autonomous Agents and Multi-Agent Systems* **16**(2) (2008), 187–209.

[3] P. Besnard and A. Hunter, *The Elements of Argumentation*, MIT Press, London, 2008.

[4] Q. Cao, M.A. Thompson and Y. Yu, Sentiment analysis in decision sciences research: an illustration to it governance, *Decision Support Systems* **54**(2) (2013), 1010–1015.

[11] http://www.facebook.com.

[5] C.I. Chesñevar, A.G. Maguitman, E. Estevez and R.F. Brena, Integrating argumentation technologies and context-based search for intelligent processing of citizens' opinion in social media, in: *ICEGOV*, D. Ferriero, T.A. Pardo and H. Qian, eds, ACM, New York, 2012, pp. 166–170.

[6] D. Choi and P. Kim, Sentiment analysis for tracking breaking events: a case study on Twitter, in: *ACIIDS (2)*, A. Selamat, N.T. Nguyen and H. Haron, eds, Lecture Notes in Computer Science, Vol. 7803, Springer, Berlin, 2013, pp. 285–294.

[7] R. Feldman, Techniques and applications for sentiment analysis, *Commun. ACM* **56**(4) (2013), 82–89.

[8] S. Gabbriellini and P. Torroni, Large scale agreements via microdebates, in: *Proceedings of the First International Conference on Agreement Technologies, AT 2012, Dubrovnik, Croatia, October 15–16, 2012*, S. Ossowski, F. Toni and G.A. Vouros, eds, CEUR Workshop Proceedings, Vol. 918, 2012, pp. 366–377, available at: ceur-ws.org.

[9] B. Ganter and R. Wille, *Formal Concept Analysis – Mathematical Foundations*, Springer, 1999.

[10] A.J. García and G.R. Simari, Defeasible logic programming: an argumentative approach, *Theory and Practice of Logic Programming* **4**(1,2) (2004), 95–138.

[11] K. Grosse, C.I. Chesñevar and A.G. Maguitman, An argument-based approach to mining opinions from Twitter, in: *Proceedings of the First International Conference on Agreement Technologies, AT 2012, Dubrovnik, Croatia, October 15–16, 2012*, S. Ossowski, F. Toni and G.A. Vouros, eds, CEUR Workshop Proceedings, Vol. 918, 2012, pp. 408–422, available at: ceur-ws.org.

[12] E. Kontopoulos, C. Berberidis, T. Dergiades and N. Bassiliades, Ontology-based sentiment analysis of Twitter posts, *Expert Syst. Appl.* **40**(10) (2013), 4065–4074.

[13] J. Leite and J. Martins, Social abstract argumentation, in: *IJCAI*, T. Walsh, ed., IJCAI/AAAI, 2011, pp. 2287–2292.

[14] J. Martineau, Identifying and isolating text classification signals from domain and genre noise for sentiment analysis, PhD thesis, Univ. Maryland, Baltimore County, USA, 2011.

[15] T.M. Mitchell, Generalization as search, *Artif. Intell.* **18**(2) (1982), 203–226.

[16] K. Mizumoto, H. Yanagimoto and M. Yoshioka, Sentiment analysis of stock market news with semi-supervised learning, in: *ACIS-ICIS*, H. Miao, R.Y. Lee, H. Zeng and J. Baik, eds, IEEE, 2012, pp. 325–328.

[17] S. Modgil, F. Toni, F. Bex, I. Bratko, C. Chesñevar, W. Dvořák, M.A. Falappa, S.A. Gaggl, A.J. García, M.P. Gonzalez, T.F. Gordon, J. Leite, M. Mozina, C. Reed, G.R. Simari, S. Szeider, P. Torroni and S. Woltran, The added value of argumentation: examples and challenges, in: *Handbook of Agreement Technologies*, Springer, Berlin, 2013, pp. 357–404.

[18] S. Ossowski, F. Toni and G.A. Vouros, eds, *Proceedings of the First International Conference on Agreement Technologies, AT 2012, Dubrovnik, Croatia, October 15–16, 2012*, CEUR Workshop Proceedings, Vol. 918, 2012, available at: ceur-ws.org. <uncited>

[19] A. Pak and P. Paroubek, Twitter as a corpus for sentiment analysis and opinion mining, in: *LREC*, N. Calzolari, K. Choukri, B. Maegaard, J. Mariani, J. Odijk, S. Piperidis, M. Rosner and D. Tapias, eds, European Language Resources Association, 2010.

[20] I. Rahwan and G. Simari, *Argumentation in Artificial Intelligence*, Springer, Berlin, 2009.

[21] M. Thelwall, K. Buckley and G. Paltoglou, Sentiment in Twitter events, *Journal of the Association for Information Science and Technology* **62**(2) (2011), 406–418.

[22] M. Thelwall, K. Buckley and G. Paltoglou, Sentiment strength detection for the social web, *Journal of the Association for Information Science and Technology* **63**(1) (2012), 163–173.

[23] F. Toni and P. Torroni, Bottom-up argumentation, in: *TAFA*, S. Modgil, N. Oren and F. Toni, eds, Lecture Notes in Computer Science, Vol. 7132, Springer, Berlin, 2011, pp. 249–262.

[24] C. Wagner, P. Singer, L. Posch and M. Strohmaier, The wisdom of the audience: an empirical study of social semantics in Twitter streams, in: *ESWC*, P. Cimiano, Ó. Corcho, V. Presutti, L. Hollink and S. Rudolph, eds, Lecture Notes in Computer Science, Vol. 7882, Springer, Berlin, 2013, pp. 502–516.