ArgP2P: An argumentative approach for intelligent query routing in P2P networks

Ana L. Nicolini, Ana G. Maguitman, and Carlos I. Chesñevar

Institute for Computer Science and Engineering (ICIC)
Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET)
Universidad Nacional del Sur – Av. Alem 1253, (8000) Bahía Blanca – Argentina
Email: {aln,agm,cic}@cs.uns.edu.ar

Abstract. The Internet is a cooperative and decentralized network built out of millions of participants that share large amounts of information. Peer-to-peer (P2P) systems go hand-in-hand with this huge decentralized network, where each individual node can serve content as well as request it. In this scenario, thematic search algorithms should lead to and benefit from the emergence of semantic communities that are the result of the interaction among participants. As a consequence, intelligent algorithms for neighbor selection should give rise to a logical network topology reflecting efficient communication patterns. When routing queries within a P2P network different conflicting issues may arise in individual nodes, such as deciding whether to propagate a query or to reject its processing. Such issues emerge in the context of incomplete and potentially inconsistent information in a distributed setting. To the best of our knowledge, current algorithmic approaches to P2P query processing are mostly based on a "reactive" approach, endowing the individual nodes with little or no intelligence.This paper presents a novel approach to use argumentation as part of the decision making machinery within individual nodes in a P2P network for thematic search. Our approach will rely on assumption-based argumentation (ABA). We provide a formalization for P2P networks for thematic search, on top of which intelligent algorithms based on ABA are specified. A case study is used to illustrate the proposed approach, providing insights into the performance of the new framework.

1 Introduction and motivations

The Internet is a cooperative and decentralized network built out of millions of participants that share large amounts of information with other users. Peer-to-peer (P2P) systems go hand-in-hand with this huge decentralized network, where each individual node can serve content as well as request it. Thematic search is the process of seeking information related to a topic of interest. In the P2P scenario, thematic search algorithms should lead to and benefit from the emergence of semantic communities that are the result of the interaction among participants. As a result, intelligent algorithms for neighbor selection should give rise to a logical network topology reflecting efficient communication patterns [1].

In this paper we propose a novel model for thematic search in P2P systems, where every node in the network has the ability to perform intelligent query routing by combining both *reactive* and *argumentative* behavior. Both nodes and queries are associated with topics. Therefore, each node needs to analyze the topic of an incoming query to determine if it is capable of answering the query or if the query needs to be forwarded to other nodes. Given a particular query Q, reactive behavior will correspond to hard-wired code used by an individual node in order to make decisions concerning Q. Such code follows a black-box model, and is typically based on specific features (e.g. whether the node is available for answering Q, whether the query Q is relevant to the node, etc.) without relying on knowledge representation and reasoning mechanisms. Argumentative behavior, on the other hand, is based on a dynamic knowledge base that captures the information acquired by the node during its lifetime, resulting from previous interactions with other nodes (e.g. whether there are reasons to believe that certain nodes are reliable to provide an answer for Q). The resulting knowledge base associated with each node will be usually incomplete and potentially inconsistent. Therefore, argumentative reasoning (formalized using assumption-based argumentation [10]) will provide every node with the ability for autonomous intelligent decision making, without interfering with the operation of the other peers. This will lead to a novel framework, called ArgP2P, that extends the traditional model for decision making in P2P thematic search, as each node will act as an autonomous agent with an inference engine which combines reactive and rational behaviors. A configuration parameter allows to set the frequency used by the node to behave reactively (just performing a black-box analysis for decision making) or rationally (i.e., applying argumentative inference to decide how to route a query). Since argumentation is a computationally expensive process, the use of this configuration parameter is crucial to ensure that the nodes' response time is reasonable in the context of a P2P search system, as will be described later.

ArgP2P applies argumentation to alleviate a common problem in P2P search frameworks, to which we refer as *Closed Communities Problem*. In this setting, as the interaction among peers increases the learning degree also increases leading to the formation of semantic communities. In this scenario one or more nodes can be disconnected from their community or can form another community with the same topic without being connected to each other. In figure 1 the logical network resulting from applying a thematic search algorithm is shown, in particular a logical network that presents a graphical visualization of the Closed Communities Problem. Each color represents a topic (similar colors represent similar topics) and a link joining two nodes represents that the nodes know each other. In this figure we can appreciate communities of nodes with related topics but we can also see related communities that are disconnected. .

The goal of argumentation in the ArgP2P framework is to provide the nodes with the ability to decide, in an autonomous way, if it is the right time to explore the network beyond their specific communities. This action makes it possible to

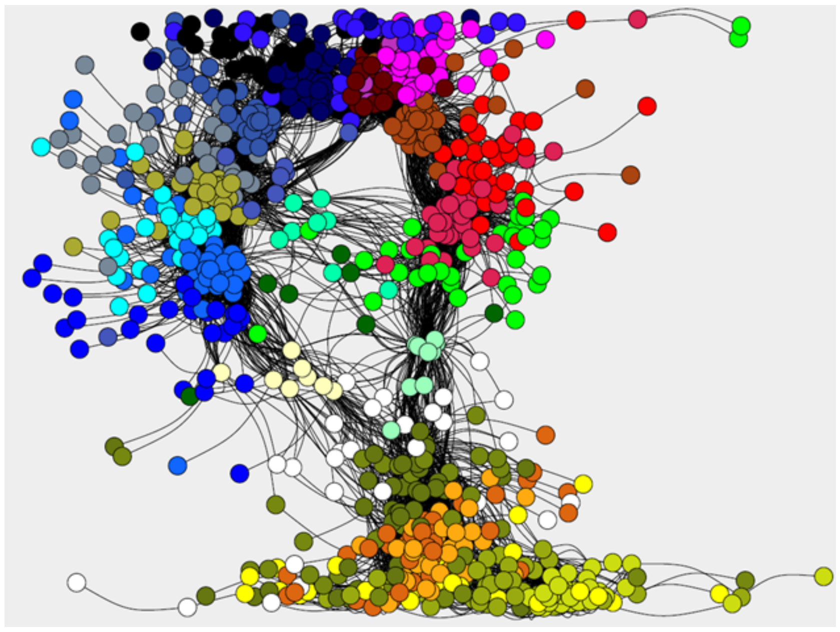

Fig. 1. Logical network resulting from applying a thematic search algorithm

discover other potentially useful peers and helps connecting related communities that otherwise would remain disconnected.

The rest of this paper is structured as follows. Section 2 presents an overview of the main elements involved in P2P networks for thematic search, providing a suitable formalization and characterizing the main elements of the algorithm for an individual node. Section 3 summarizes the main elements of assumption-based argumentation (ABA). Section 4 presents a framework which integrates ABA within P2P thematic search. Section 5 presents a case study to illustrate how the proposed framework can be deployed in a P2P network. Section 6 discusses the main characteristics of our approach as well as some comparisons with related work. Finally, Section 7 summarizes the conclusions and discusses some future work.

2 P2P networks for thematic search: overview

A peer-to-peer [21] network is a distributed system in which every peer (host) communicates with other peers without the intervention of centralized hosts. Every peer in a P2P system has the potential to act as a client or as a server, sharing resources to work in conjunction. Robustness is an expected property of a P2P system, which accounts for maintaining the functionality and integrity of the system even if a peer fails. The absence of a central server distinguishes P2P networks from a client-server model. The most common applications of

P2P networks are file sharing, distributed file storage, resource sharing and distributed computing. A good survey and discussion of P2P architectures and applications can be found in [3].

P2P network topologies are typically classified as *structured* [15] or *unstructured* [13], while some networks combine some properties of both [20]. A structured network is organized into a specific topology with a protocol aimed at ensuring a reasonable search performance. On the other hand, an unstructured network does not follow a specific pattern for the organization of its nodes and has a random topology. Because of this lack of structure, these networks have a relatively low search efficiency compared with the structured ones. As a consequence, the study of algorithms that improve search efficiency in unstructured P2P systems has been an active research area in the last years (e.g., [22, 19, 7, 27]).

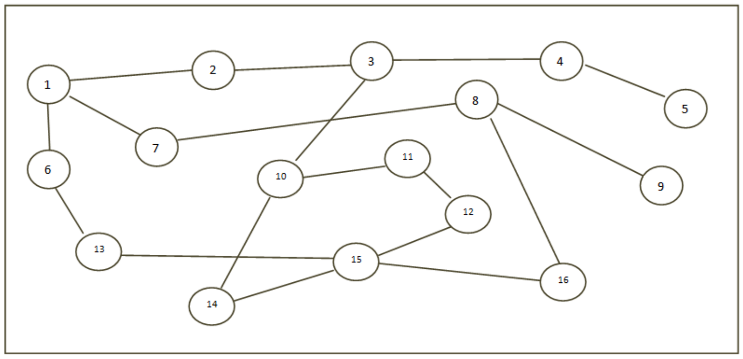

Fig. 2. Example of an unstructured P2P network

Figure 2 shows an example of an unstructured P2P network. In this network all participants (nodes) stand for devices such as personal computers, smartphones or large servers that exchange information. The typical way of routing queries in this kind of networks is by *flooding*. For example, node 1 generates a query message so that this query is sent through its adjacent nodes (nodes 2, 6 and 7). If these nodes do not provide an answer for the query, then they forward on their turn the query to their adjacent nodes, and so on, until an answer is found or the pre-established time to live (TTL) for the query message is reached.

The major limitations of flooding algorithms are the resulting network congestion and the possibility of not finding the answer to the query. Some results confirm that in order to solve this problem it is desirable that the participants send their queries to other participants that are specialized in the query topic. Thus, a query can be efficiently propagated in the network, suggesting that collaborative and distributed search can benefit from the user context and the existence of semantic communities [26].

Endowing the nodes with the ability to make decisions in an autonomous way gives them the possibility of routing queries to potentially relevant nodes based on some acquired knowledge about other peers' interests. Sometimes this knowledge is inconsistent and making the right decision is not a trivial issue. As we will see later, argumentative reasoning will provide a powerful complement for the traditional reactive behavior adopted by nodes to route queries.

2.1 Formalizing the notion of P2P network

A P2P network is a set of interconnected devices that exchange information. The static representation of a P2P network can be defined as a graph. However, in order to capture the flow of a query through the network, we will include a distinguished function to express time, characterizing a *dynamic* P2P network. Formally:

Definition 1 *A* dynamic P2P network *(or just* P2P network*) is a 3-uple $P = (\mathcal{N}, \mathcal{E}, S_t)$ where $\mathcal{N}$ is a (finite) set of nodes, $\mathcal{E}$ is a finite collection of edges (each edge is a pair of elements in $\mathcal{N}$) and a function S_t that captures the dynamic status of a node $n \in \mathcal{N}$ at a given time t, and is defined as follows:*

$$S_t : \mathcal{N} \longrightarrow Consult \cup Forward \cup \{\textsf{\textit{reply}}, \textsf{\textit{discard}}, \textsf{\textit{idle}}\},$$

where $Consult =_{def} \{\textsf{\textit{consult}}\} \times 2^{\mathcal{N}}$ and $Forward =_{def} \{\textsf{\textit{forward}}\} \times 2^{\mathcal{N}}$.

In a P2P network a node has a finite number of possible states: consult by sending a query to a set of candidates nodes; forward a query to a set of nodes; reply a query, discard a message; and when none of the previous states apply, the node will just remain idle. We can see that each of these states corresponds with an element in the image of S_t. We will assume that the whole P2P network (i.e., every node in the network) initializes at time $t = 0$. At any given time t, a particular query will be associated with a set of nodes in the network (the ones which are processing the query at time t). We assume that the time counter will be incremented by 1 after a node processes the query. Next we show the formal characterization of the dynamic P2P network presented in Figure 2, assuming that every node at time $t = 0$ is idle.

$P = (\mathcal{N}, \mathcal{E}, S_t)$
$\mathcal{N} = \{1, 2, ..., 16\}$
$\mathcal{E}$ ={(1,2),(1,7),(1,6),(2,3),(2,1),(3,2),(3,10),(3,4), (4,3), (4,5), (5,4), (6,1), (6,13), (7,1), (7,8), (8,7), (8,9), (9,8), (10,3), (10,14), (10,11), (11,10), (11,12), (12,11), (12,15), (13,6), (13,15), (14,10), (14,15), (15,13), (15,16), (15,12), (15, 14), (16,8), (16,15)}
$\forall n \in \mathcal{N}, S_0(n) = \textsf{idle}$

3 Assumption-based Argumentation: fundamentals

Argumentation has been identified as a way for understanding and reconciling differences and similarities of various existing formalisms for non-monotonic, default reasoning as studied in AI. This line of research has led to the development of different argumentation frameworks, notably abstract argumentation (AA) [8] and assumption-based argumentation (ABA) [5]. ABA is a general-purpose argumentation framework that can be instantiated to support various applications and specialised frameworks, including default reasoning frameworks [5, 17, 14], problems in legal reasoning [11], game-theory [9], practical reasoning and decision-theory [23, 2, 4, 12]. However, whereas in AA arguments and attacks between arguments are abstract and primitive, in ABA arguments are deductions supported by assumptions. An attack by one argument against another is a deduction by the first argument of the contrary of an assumption supporting the second argument.

ABA is equipped with a computational machinery to determine the acceptability of claims by building and exploring a dialectical structure of a proponent's argument for a claim, an opponent's counter arguments attacking the argument, the proponent's arguments attacking all the opponent's counter arguments, and so on. This computation style, which has its roots in logic programming, has several advantages over other computational mechanisms for argumentation. The advantages are due mainly to the fine level of granularity afforded by interleaving the construction of arguments and determining their acceptability. For several reasons, included the previous one, ABA has turned out to be particularly applicable in the context of real-world problems (e.g. in grid computing [25]) and we will also use it as the underlying framework for our approach.

In what follows, we will provide the main definitions and concepts for ABA, in order to make this paper self-contained.For a complete description of assumption-based argumentation and its applications, the reader is referred to [24].

ABA frameworks

An ABA framework is a tuple $\langle$ *L,R,A,*$\rangle$ where

- $\langle$ *L,R*$\rangle$ is a deductive system, with L the *language* and R a set of *rules*, that we assume of the form $\sigma_0 \leftarrow \sigma_1, ..., \sigma_m (m \geq 0)$ with $\sigma_i \in L(i = 0, ..., m)$; σ_0 is referred to as the head and $\sigma_1, ..., \sigma_m$ as the *body* of the rule $\sigma_0 \leftarrow \sigma_1, ..., \sigma_m$;
- $A \subseteq L$ is a (non-empty) set, referred to as *assumptions*;
- $^{-}$is a total mapping from A into L; $\bar{\text{a}}$ is referred to as the *contrary* of a.

In ABA, *arguments* are deductions of claims using rules and supported by sets of assumptions, and *attacks* are directed at the assumptions in the support of arguments:

- *an argument for (the claim)* $\sigma \in L$ *supported by* A$\subseteq A$ (A $\vdash \sigma$ in short) is a deduction for σ supported by A (and some $R \subseteq R$)

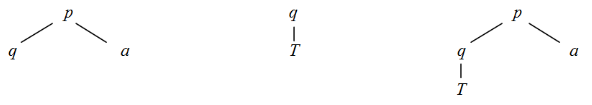

Fig. 3. Deductions for p (left), q (middle) and p (right) for the example presented in the "ABA frameworks" section.

– *an argument* $A_1 \vdash \sigma_1$ *attacks* an argument $A_2 \vdash \sigma_2$ iff σ_1 is the contrary of one of the assumptions in A_2.

Example Let $\langle L, R, A\rangle$ be an ABA framework, where: L=$\{a,b,c,p,q,r,s,t\}$, R=$\{p \leftarrow q, a; q \leftarrow; r \leftarrow b, c\}$, A=$\{a,b,c\}$, and it holds that $\overline{a} = r$; $\overline{b} = s$; $\overline{c} = t$. Then the following arguments can be obtained:

$$\{\} \vdash q, \{a\} \vdash p, \{b,c\} \vdash r, \{a\} \vdash a, \{b\} \vdash b, \{c\} \vdash c$$

Arguments can be depicted as trees, as shown in Figure 3 In this particular example, the following attack relationships hold: $\{b,c\} \vdash r$ attacks $\{a\} \vdash p$ as well as $\{a\} \vdash a$.

Several *semantics* are available in ABA (e.g. grounded skeptical semantics, admissible semantics, etc.). In this paper we adopt the *grounded skeptical* semantics, which sanctions only one set of arguments as "winning". For space reasons the reader is referred to [24] for technical details.

4 The ArgP2P framework

As discussed before, dynamic P2P networks take a black-box approach to deal with the query routing problem. Black-box approaches do not involve any form of inference, and simply rely on a basic analysis of certain local variables associated with an individual node to decide how to proceed. Such variables involve node features such as availability for answering an incoming query, relevance of the query to the nodes' interests, etc. These features are analyzed by each node by means of hard-wired code, resulting in reactive behavior and limited problem solving capabilities. Every query Q in thematic P2P search is always associated with a topic T (e.g. query = "prime numbers" and topic = "math") and each node of the network has a finite set of *topics* of interest (e.g node 1 is interested in the set {math, physics}). A node interested in a specific topic is a potential node to reply queries about this topic, and every node in the network has knowledge about other nodes and their interest. This prompts the following definition:

Definition 2 *Let $P = (\mathcal{N}, \mathcal{E}, S_t)$ be a P2P network. For every node $n \in \mathcal{N}$, we define a* dynamic knowledge base KB_n *with the following form:*

$$KB_n = \langle (topic_1, \mathcal{N}_{topic_1}), (topic_2, \mathcal{N}_{topic_2}), ..., (topic_n, \mathcal{N}_{topic_n}) \rangle,$$

where $\mathcal{N}_{topic_i}$ stands for a set of nodes associated with $topic_i$.

The problem of this approach is that one or more nodes could be disconnected from other communities of nodes with the same topic or two communities[1] of nodes with the same topics could be disconnected from each other. Consequently some messages could never find an answer. The case study given in Section 5 will illustrate this problem. In order to solve it, we will extend the model by adding a decision-making component: the Argumentative Decision-Making System, that allows a node to decide whether it is necessary to explore the network, sending queries by flooding, which helps to discover other communities of nodes that share common interests.

In order to capture the relevant features associated with reactive and argumentative behavior, distinguished vectors of Boolean variables will be defined. Formally:

Definition 3 *Let $P = (\mathcal{N}, \mathcal{E}, S_t)$ be a P2P network. For every node $n \in \mathcal{N}$, an incoming query Q associated with a topic T, and a KB_n we define the following vectors of Boolean variables:*

$$RB_n(Q,T) = \langle availability(Q,T), relevance(Q,T), awareness(Q,T)\rangle,$$

where:

- *$availability(Q,T)$ is true iff node n (interested in topic T) can answer an incoming query Q associated with topic T;*
- *$relevance(Q,T)$ is true iff the topic T of the incoming query Q is relevant to node n;*
- *$awareness(Q,T)$ is true iff node n knows other nodes interested in the topic T of the query Q.*

$$AB_n(KB_n) = \langle interest(KB_n), reliability(KB_n), congestion(KB_n), need\_to\_explore(KB_n)\rangle,$$

where

- *interest is true iff node n contains information in its knowledge base KB_n about other nodes and their interests;*
- *reliability is true iff the knowledge stored in KB_n provides reasons to believe that certain nodes are reliable;*
- *congestion is true iff the knowledge stored in KB_n provides reasons to believe that there exist paths to certain nodes that are congested;*
- *need_to_explore is true iff the knowledge stored in KB_n provides reasons to believe that it is necessary to explore the network.*

Definition 4 *Let $P = (\mathcal{N}, \mathcal{E}, S_t)$ be a P2P network. A* Reactive Decision-Making System *for a node $n \in \mathcal{N}$, denoted $RDMS_n$, is a hard-wired black box system that given a query Q associated with a topic T, takes $RB_n(Q,T)$ and KB_n and returns a decision D indicating how to route Q in P. Based on the query Q, its topic T and the knowledge base KB_n associated with the node n, $RDMS_n$ can choose one of the following courses of action:*

[1] A community is a set of nodes with interest in related topics.

– Discard *the query* Q *if its TTL has expired;*
– Forward the query Q to nodes interested in topic T *when there is information in* KB_n *indicating to do so;*
– Forward the query Q to adjacent nodes *whenever the previous situation is not the case.*

Definition 5 *Let* $P = (\mathcal{N}, \mathcal{E}, S_t)$ *be a P2P network. An* Argumentative Decision-Making System *for a node* $n \in \mathcal{N}$*, a query* Q *and a topic* T *is denoted* $ADMS_n$ *and is an ABA program (based on* KB_n,$AB_n(KB_n)$,Q *and* T*) used to determine whether the decision* $need\_to\_explore(KB_n)$ *is warranted under grounded skeptical semantics.*[2] *If* $need\_to\_explore(KB_n)$ *is true, the system returns* true *and flooding is used as a strategy for routing the query, otherwise* false *is returned and the strategy for routing the query is the strategy obtaining from the* $RDMS_n$ *.*

Definition 6 *Let* $P = (\mathcal{N}, \mathcal{E}, S_t)$ *be a P2P network. We define* **RDMS** *as the set of all the* $RDMS_n$ *systems for all* $n \in \mathcal{N}$*. Analogously, we define* **ADMS** *as the set of all the* $ADMS_n$ *systems for all* $n \in \mathcal{N}$*.*

Definition 7 *Let* $P = (\mathcal{N}, \mathcal{E}, S_t)$ *be a dynamic P2P network, an* ArgP2P framework *is a 3-uple* $(P, \mathbf{RDMS}, \mathbf{ADMS})$ *where:*

– P *is a P2P network,* $P = (\mathcal{N}, \mathcal{E}, S_t)$*,*
– **RDMS** *represents the Reactive Decision-Making component of* P*, and*
– **ADMS** *represents the Argumentative Decision-Making component of* P*.*

In Figure 4 we present a graphical representation of the ArgP2P framework for a node n. This framework augments an already tested and implemented application [18] by adding an Argumentative Decision-Making component (indicated by the dotted square in the figure). In algorithm 1 we present the high-level code of the RunNode algorithm for every node j in an ArgP2P framework. Algorithm 2 presents the high-level code of the Reactive Decision-Making System. Finally, the high-level code of the Argumentative Decision-Making System is presented in algorithm 3. These algorithms model the most important features of the ArgP2P framework.

The main algorithm exhibits the interaction between the Reactive Decision-Making System ($RDMS_j$) and the Argumentative Decision-Making System ($ADMS_j$) in a node j. In order not to slow down the average response time of the node, note that $ADMS_j$ is invoked 1 out of every 15 executions of the code (this parameter is configurable, allowing to set the frequency for the operation of the Argumentative Decision-Making System in each node). The $RDMS_j$ checks the variables of the RB_j vector using the knowledge base KB_j and then makes a decision about how to route the query Q according to the knowledge

[2] We are aware of different possible semantics for ABA. For illustrative purposes we have adopted the grounded skeptical semantics, even though other alternative semantics could be used. An in-depth analysis of these alternatives is outside the scope of this paper.

Algorithm 1: RunNode Code for every node j in a P2P framework.

```
Input: A query $Q$ associated with a topic $T$.
Output: A decision $D$ about what action to take after the arrival of the query
        $Q$.
$count$:= $count + 1$; \\ count is a global variable associated with every particular
node, initially set to 0, and incremented every time RunNode is executed.
LoadKnowledgeBase($KB_j$); \\ Initializes the knowledge base with stored
information.
Decision $D$ := ReactiveDecisionMaking($RB_j$, $Q$, $T$, $KB_j$);
if (count mod 15 = 0) then
    if (ArgumentativeDecisionMaking($AB_j$, $KB_j$)) then
        $D = Forward\_to\_Adjacent\_Node$;
    end

end
return $D$.
```

Algorithm 2: ReactiveDecisionMaking for every node j ($RDMS_j$)

```
Input: A query $Q$ associated with a topic $T$, a vector of variables $RB_j$ and a
       knowledge base $KB_j$.
Output: A decision $D$ about what action to take after the arrival of the query
        Q.
for each $p_i \in RB_j$ do
    check($p_i$, $KB_j$);
end
Decision $D$ := makeDecision($Q$, $T$,$RB_j(Q,T)$, $KB_j$);
return $D$.
```

Algorithm 3: ArgumentativeDecisionMaking for every node j ($ADMS_j$)

```
Input: A knowledge base $KB_j$ and a vector of variables $AB_j(KB_j)$.
Output: Returns true iff $need\_to\_explore(KB_j)$ is warranted using grounded
        skeptical semantics, otherwise it returns false.
boolean result;
constructABAprogram($KB_j$,$AB_j(KB_j)$);
result:= executeABAprogram($need\_to\_explore(KB_j)$);
return result.
```

acquired. The $ADMS_j$ builds an ABA program from vector AB_j and the knowledge base KB_j. Then it executes the program to reach a final decision about whether or not it is necessary to explore the network to increment its global knowledge.

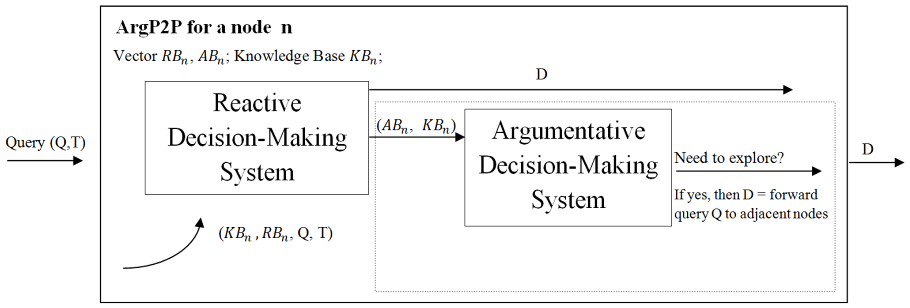
ArgP2P for a node n
Vector RB_n, AB_n; Knowledge Base KB_n;
Query (Q,T)
Reactive Decision-Making System
D
(AB_n, KB_n)
Argumentative Decision-Making System
Need to explore?
If yes, then D = forward query Q to adjacent nodes
D
(KB_n, RB_n, Q, T)

Fig. 4. High-level description of the components in every node n of an ArgP2P framework.

5 Case study

Consider the P2P network in Figure 2. Suppose that nodes 1, 2, 8 and 10 are interested in the topic *math* and have the knowledge shown in Table 1 about the rest of the network.

KB_n	Nodes associated with the topic "math"		
KB_1	5	8	10
KB_5	1	8	10
KB_8	1	5	10
KB_{10}	1	5	8

Table 1. Nodes' knowledge about the topic "math".

Suppose that node 1 generates a query message and sends it to the nodes that, according to its knowledge, are interested in the topic *math*. Figure 5(a) shows the initial paths followed by these messages, which are represented with dotted arrows. The formal representation of the network at time ($t = 1$) is the following:

$$S_1(n) = \mathsf{idle}\ \forall\ n \in \mathcal{N} - \{1\},$$
$$S_1(1) = (consult, \{5, 8, 10\}).$$

Then node 5 receives the forwarded message and it cannot answer the query so it forwards the message to the candidate nodes that it knows from KB_5 (except node 1 which is where the query was originated). Figure 5(b) illustrates this situation. Nodes 8 and 10 cannot answer the query either and forward the message by taking into account KB_8 and KB_{10} respectively. Figures 5(c) and 5(d) show this scenario. All these actions take place at time $t = 2$ and the formal representation is:

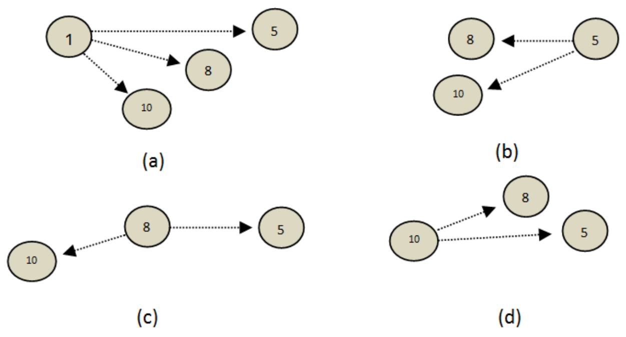

Fig. 5. Graphical representation of nodes sending messages at different time points when routing a query in a P2P network (case study).

$$\begin{gathered}S_2(n) = \mathsf{idle}\ \forall\ n \in \mathcal{N} - \{5, 8, 10\},\\ S_2(5) = (forward, \{8, 10\}),\\ S_2(8) = (forward, \{5, 10\}),\\ S_2(10) = (forward, \{5, 8\}).\end{gathered}$$

After forwarding the messages, node 5 receives a message from node 8 that was generated by node 1 and a message from node 10 that was also generated by node 1; finally node 10 receives a message from node 5 generated by node 1 and another from node 8 also originated in node 1. All these messages are discarded at time $t = 3$ because the receptor nodes already contain a message in their records with that identifier indicating that the node has already processed the message, and consequently node 1 will not be able to find the answer of the query. As a result, the actions at time $t = 3$ are the following:

$$\begin{gathered}S_3(n) = \mathsf{idle}\ \forall\ n \in \mathcal{N} - \{5, 8, 10\},\\ S_3(5) = \mathsf{discard},\\ S_3(8) = \mathsf{discard},\\ S_3(10) = \mathsf{discard}.\end{gathered}$$

Assume that in order to solve this problem, node 8 decides to break the protocol and explore the network. Therefore, it will send messages to nodes 7, 9 and 16 at $t = 3$. Node 7 only has node 1 as an adjacent node, which is the same node that generated the query and therefore it is not considered. Node 9 has no potentially useful adjacent nodes to forward the message (the message arrived from node 8, so it is not considered). Node 16 will forward the message to node 15 (at time $t = 4$). Finally, at time $t = 5$, node 15 is able to answer the query originated by node 1.

$$\begin{gathered}S_3(n) = \mathsf{idle}\ \forall\ n \in \mathcal{N} - \{5, 8, 10\},\\ S_3(5) = \mathsf{discard},\\ S_3(8) = (forward, \{7, 9, 16\}),\\ S_3(10) = \mathsf{discard}.\end{gathered}$$

$$S_4(n) = \textsf{idle} \ \forall \ n \in \mathcal{N} - \{7, 9, 16\},$$
$$S_4(7) = \textsf{discard},$$
$$S_4(9) = \textsf{discard},$$
$$S_4(16) = (forward, \{15\}).$$

$$S_5(n) = \textsf{idle} \ \forall \ n \in \mathcal{N} - \{15\},$$
$$S_5(15) = \textsf{reply}.$$

When the $ADMS_8$ system is executed, an ABA program is built. The assumptions, rules and contraries that constitute the program can be automatically generated from knowledge that is readily available to the node from AB_8 and KB_8. This program allows node 8 to determine whether it is better or not to explore the network. In order to accomplish this, the node needs to deal with incomplete and potentially inconsistent information. As a consequence argumentative reasoning provides the appropriate mechanisms to reach a decision. If the node decides to explore, then it discards $RDMS_8$'s initial decision to forward the query message to potentially useful nodes. The following logic program corresponds to the ABA program generated by $ADMS_8$, as well as the associated derivation tree representing how the decision *"need to explore"* is reached. The resulting derivation tree is shown in Figure 6. This example illustrates the fact that the ABA program will have to deal with inconsistent information. In particular the decision $\overline{need\_to\_explore}$ is inconsistent with the decision $forward(1)$. Similarly, the assumption $\overline{notCongested(1)}$ is inconsistent with the conclusion $congested(1)$.

$A =\{notCongested(1)$, $notCongested(5)$, $notCongested(10)$,
$goodCandidate(1)$, $goodCandidate(5)$, $goodCandidate(10)$,
$alwaysCongested(1)$, $neverCongested(5)$, $neverCongested(10)$,
$longTimeToRespond(1)$, $need\_to\_explore\}$

$R =\{math(1) \leftarrow;$ $math(5) \leftarrow;$$math(10) \leftarrow;$$goodResponseRecord(1) \leftarrow;$
$goodResponseRecord(5) \leftarrow;$
$forward(1) \leftarrow math(1), notCongested(1), reliable(1);$
$forward(5) \leftarrow math(5), notCongested(5), reliable(5);$
$forward(10) \leftarrow math(10), notCongested(10), reliable(10);$
$congested(1) \leftarrow longTimeToRespond(1), alwaysCongested(1);$
$congested(5) \leftarrow longTimeToRespond(5), alwaysCongested(5);$
$congested(10) \leftarrow longTimeToRespond(10), alwaysCongested(10);$
$reliable(1) \leftarrow goodResponseRecord(1), goodCandidate(1);$
$reliable(5) \leftarrow goodResponseRecord(5), goodCandidate(5);$
$reliable(10) \leftarrow goodResponseRecord(10), goodCandidate(10);$
$notExplore \leftarrow forward(1);$
$notExplore \leftarrow forward(5);$
$notExplore \leftarrow forward(10)\}$

$\overline{notCongested(1)} = congested(1);$

$\overline{notCongested(5)} = congested(5);$
$\overline{notCongested(10)} = congested(10);$
$\overline{goodCandidate(1)} = badCandidate(1);$
$\overline{goodCandidate(5)} = badCandidate(5);$
$\overline{goodCandidate(10)} = badCandidate(10);$
$\overline{alwaysCongested(1)} = neverCongested(1);$
$\overline{neverCongested(5)} = alwaysCongested(5);$
$\overline{neverCongested(10)} = alwaysCongested(10);$
$\overline{longTimeToRespond(1)} = lowTimeToRespond(1);$
$\overline{need\_to\_explore} = forward(1);$
$\overline{need\_to\_explore} = forward(5);$
$\overline{need\_to\_explore} = forward(10);$
$\overline{need\_to\_explore} = notExplore;$

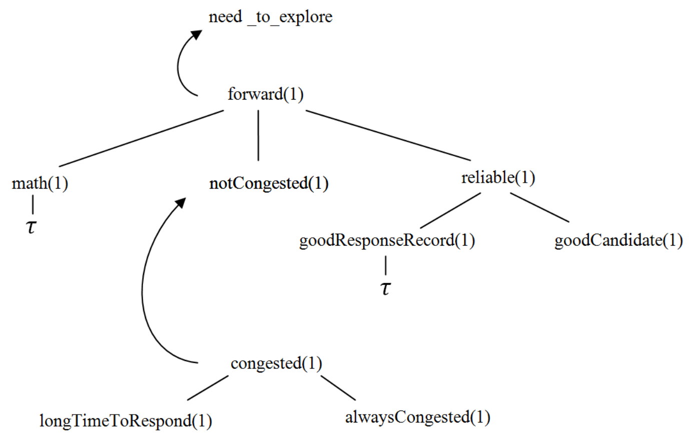

Fig. 6. Arguments and attacks involved in a derivation tree supporting the decision *"need_to_explore"*. Pointing arrows denote attacks between arguments.

6 Discussion and related work

The concept of "knowledge sharing" is an emerging topic generally based upon cooperation between peers. This concept represents the trade-off between sharing knowledge to improve the global knowledge of the network and the cost of sending the messages with this information. The first step to sharing knowledge is to establish friendship relationships in a similar way as it is implemented in Self Learning Query Routing (SLPS) [6]. In the SLPS approach, the interests of the peers are learned based on their search result history, which allow to automatically construct friend relations based on the similarity of users' interests. In this routing model queries are routed to friend nodes following an unstructured pattern. If the searches in friend nodes fail, broadcast search will be executed. Each node ranks their friends from high to low according to the number of shared files and chooses the first top K nodes. As time goes on, a node can learn which are those friend nodes that share the same interests with it.

The identification of semantic communities can help predict the performance of a P2P search application. Akavipat et al. [1] evaluate algorithms for searching in P2P networks based on the concept of semantic similarity. Their experiments with different routing algorithms on their peer Web Search 6S (unstructured network) suggest that despite the use of an unstructured overlay network, peers' interactions lead to the formation of semantic communities.

All these approaches provide P2P whith some kind of intelligence in order to route queries to specific peers, but no one has the capability of determining whether or not the decision that was made is the best one to increment the global knowledge of the network. In an argumentative context a node with the ability to discern becomes an "agent". Kowalski [16] classified agents as *rational* or *reactive*. In general terms the difference between them is that a reactive agent receives a perception of the environment and translates it into a specific action; in contrast, a rational agent receives a perception of the environment and takes a certain time to "think" what action to take. Given these definitions, each of the nodes becomes a rational agent able to make fully independent decisions from which the node, its community or the whole network can benefit.

In the last years, there has been particular interest in deploying argumentation within P2P infrastructures. In particular, the ArguGRID project [25] aims at supporting service selection and composition in distributed environments, including the Grid and Service-oriented architectures, by means of argumentative agents, an agent environment, a service-composition environment, Peer-to-Peer technology and Grid middleware. Agents are argumentative in that they use argumentation-based decision-making and argumentation-supported negotiation of services and contracts. The integration of all technologies gives rise to the overall ArguGRID platform. In contrast with ArguGRID, our proposal introduces a framework where every node has an autonomous two-stage decision-making process with the only goal of routing queries efficiently in a P2P network, without negotiation between peers.

7 Conclusions and future work

In this paper we have outlined a novel model for thematic search in P2P networks, where every node in the network has the ability to combine both reactive and rational (argumentative) behavior. The argumentative inference engine is provided by an ABA program, which allows to enhance the decision-making capabilities in every node, based on knowledge acquired by the node during its lifetime. A configuration parameter allows to set the frequency by which argumentation is used during the decision making process (e.g. by setting the global variable *count* to 1, the node performs argumentative reasoning every time a query routing decision is to be made).

Part of our future work involves an empirical comparison between the ArgP2P framework and a purely reactive P2P framework. In addition, we plan to contrast the performance of different argumentative semantics and different *count values* for the nodes. Research in this direction is being pursued.

Acknowledgments: This research was funded by CONICET (PIP 112-201201-00487 and PIP 112-201101-01000),Universidad Nacional del Sur (PGI-UNS 24/N029) and ANPCyT(PICT 2014-0624).

References

1. Akavipat, R., Wu, L.-S., Menczer, F., and Maguitman, A. G. (2006). Emerging semantic communities in peer web search. In *Proceedings of the international workshop on Information retrieval in peer-to-peer networks*, P2PIR '06, pages 1–8, New York, NY, USA. ACM.
2. amaury Matt, P. and Toni, F. Argumentation-based agents for eprocurement.
3. Androutsellis-Theotokis, S. and Spinellis, D. (2004). A survey of peer-to-peer content distribution technologies. *ACM Computing Surveys (CSUR)*, 36(4):335–371.
4. Besnard, P. et al. (2008). Towards argumentation-based contract negotiation. *Computational Models of Argument: Proceedings of COMMA 2008*, 172:134.
5. Bondarenko, A., Toni, F., and Kowalski, R. A. (1993). An assumption-based framework for non-monotonic reasoning. In *LPNMR*, volume 93, pages 171–189.
6. Chen, H., Gong, Z., and Huang, Z. (2005). Self-learning routing in unstructured p2p network. *International journal of information technology*, 11(12):59–67.
7. Du, N., Wang, B., and Wu, B. (2008). Community detection in complex networks. *Journal of Computer Science and Technology*, 23:672–683.
8. Dung, P. M. (1995a). On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial intelligence*, 77(2):321–357.
9. Dung, P. M. (1995b). On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, 77:321–357.
10. Dung, P. M., Kowalski, R. A., and Toni, F. (2009). Assumption-based argumentation. In *Argumentation in Artificial Intelligence*, pages 199–218. Springer.
11. Dung, P. M. and Thang, P. M. (2008). Towards an argument-based model of legal doctrines in common law of contracts. *Proc. CLIMA IX*, 7.
12. García, A. J. and Simari, G. R. (2004). Defeasible logic programming: An argumentative approach. *Theory and practice of logic programming*, 4(1+ 2):95–138.
13. Jin, X. and Chan, S.-H. G. (2010). Unstructured peer-to-peer network architectures. In *Handbook of Peer-to-Peer Networking*, pages 117–142. Springer.
14. Kakas, K. and Toni, F. (1999). Computing argumentation in logic programming. *Journal of Logic and Computation*, 9(4):515–562.
15. Korzun, D. and Gurtov, A. (2012). *Structured peer-to-peer systems: fundamentals of hierarchical organization, routing, scaling, and security.* Springer Science & Business Media.

16. Kowalski, R. A. (1995). Using meta-logic to reconcile reactive with rational agents. *Meta-logics and logic programming*, pages 227–242.
17. Kowalski, R. A. and Toni, F. (1996). Abstract argumentation. *Artificial intelligence and law*, 4(3-4):275–296.
18. Nicolini, A. L., Lorenzetti, C. M., Maguitman, A. G., and Chesñevar, C. I. (2013). Intelligent algorithms for reducing query propagation in thematic p2p search. In *Anales del XIX Congreso Argentino de Ciencias de la Computación (CACIC)*, pages 71–79, Mar del Plata, Buenos Aires, Argentina.
19. Radicchi, F., Castellano, C., Cecconi, F., Loreto, V., and Parisi, D. (2004). Defining and identifying communities in networks. *Proceedings of the National Academy of Sciences of the United States of America*, 101(9):2658–2663.
20. Rosenfeld, A., Goldman, C. V., Kaminka, G. A., and Kraus, S. (2009). Phirst: A distributed architecture for {P2P} information retrieval. *Information Systems*, 34(2):290 – 303.
21. Schollmeier, R. (2001). A definition of peer-to-peer networking for the classification of peer-to-peer architectures and applications. In *Peer-to-Peer Computing, 2001. Proceedings. First International Conference on*, pages 101–102.
22. Tang, C., Xu, Z., and Dwarkadas, S. (2003). Peer-to-Peer Information Retrieval Using Self-Organizing Semantic Overlay Networks. In *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '03, pages 175–186, New York, NY, USA. ACM.
23. Toni, F. Assumption-based argumentation for selection and composition of services. In *In: Proceedings of the 8th International Workshop on Computational Logic in Multi-Agent Systems (CLIMA VIII) (2007*.
24. Toni, F. (2014). A tutorial on assumption-based argumentation. *Argument and Computation*, 5:89–117.
25. Toni, F., Grammatikou, M., Kafetzoglou, S., Lymberopoulos, L., Papavassileiou, S., Gaertner, D., Morge, M., Bromuri, S., McGinnis, J., Stathis, K., Curcin, V., Ghanem, M., and Guo, L. (2008). The argugrid platform: An overview. In Altmann, J., Neumann, D., and Fahringer, T., editors, *Grid Economics and Business Models*, volume 5206 of *Lecture Notes in Computer Science*, pages 217–225. Springer Berlin Heidelberg.
26. Voulgaris, S., Kermarrec, A., Massouli, L., and van Oteen, M. (2004). Exploiting semantic proximity in peer-to-peer content searching. In *Proceedings of the 10th IEEE International Workshop on Future Trends of Distributed Computing Systems*, pages 238–243, Washington, DC, USA. IEEE Computer Society.
27. Wang, L. (2011). Sofa: An expert-driven, self-organization peer-to-peer semantic communities for network resource management. *Expert Systems with Applications*, 38:94–105.