Argumentation Based Contract Monitoring in Uncertain Domains

Nir Oren, Timothy J. Norman, Alun Preece
Department of Computing Science, University of Aberdeen, Aberdeen, Scotland
noren,tnorman,apreece@csd.abdn.ac.uk

Abstract

Few existing argumentation frameworks are designed to deal with probabilistic knowledge, and none are designed to represent possibilistic knowledge, making them unsuitable for many real world domains. In this paper we present a subjective logic based framework for argumentation which overcomes this limitation. Reasoning about the state of a literal in this framework can be done in polynomial time. A dialogue game making use of the framework and a utility based heuristic for playing the dialogue game are also presented. We then show how these components can be applied to contract monitoring. The dialogues that emerge bear some similarity to the dialogues that occur when humans argue about contracts, and our approach is highly suited to complex, partially observable domains with fallible sensors where determining environment state cannot be done for free.

1 Introduction

Most research in argumentation theory has focused on interactions between arguments [Prakken and Vreeswijk, 2002]. Researchers have proposed various underlying logics capable of representing different domain attributes. For example, a large body of work exists on representing and using defaults for legal argumentation [Prakken and Sartor, 1998]. A number of researchers have also looked at how argument takes place in domains with uncertainty. Most of the work in this area is probability theory based, and includes Bayesian based argumentation frameworks [Vreeswijk, 2004] and Pollock's OSCAR system [Pollock, 1995].

Due to its basis in plausible reasoning, we believe that uncertainty theory is more applicable than probabilistic reasoning to many domains of argumentation. In this paper, we present a subjective logic [Jøsang, 2002] based approach to argumentation. Using subjective logic, we are able to naturally present arguments in which uncertainty exists, and can encapsulate concepts such as accrual of arguments [Prakken, 2005], which many existing frameworks handle poorly.

After describing the argumentation framework, we present a dialogue game which allows agents to make use of the framework. We then present a utility based heuristic that allows agents to participate in the dialogue game.

One possible application of such a framework is contract monitoring and enforcement. That is, given a contract and an environment, determining current contract state as well as what sanctions should come into effect. Not only is our approach able to operate in partially observable domains with fallible sensors, but we also take into account the fact that probing a sensor might be associated with a cost. Other systems for contract monitoring often require fully observable domains with "honest" sensors (e.g. [Xu and Jeusfeld, 2003]). Even when able to handle faulty sensors, some approaches assume that all domain states can be probed, and that all such probing is free [Daskalopulu *et al.*, 2002].

In the next section, we provide an overview of subjective logic. After that, we present our framework, and provide an illustrative example. We conclude the paper by examining related research, and examine possible avenues of future work.

2 Subjective Logic

Subjective logic[1] is an approach for combining and assessing subjective evidence from multiple sources based on Dempster-Schafer theory. An opinion ω about an atomic proposition ϕ is represented as a triple $\omega(\phi) = \langle b(\phi), d(\phi), u(\phi)\rangle$ where $b(\phi)$ represents the level of belief that ϕ holds; $d(\phi)$ stands for the level of disbelief (i.e. the probability that ϕ does not hold), and $u(\phi)$ represents the level of uncertainty of the opinion. $b(\phi) + d(\phi) + u(\phi) = 1$, and $b(\phi), d(\phi), u(\phi) \in [0..1]$.

A number of operations on opinions have been defined, allowing one to create compound opinions from atomic ones. These are listed in Figure 1.

Subjective logic allows us to use common logical operators such as conjunction, disjunction and negation. The last of these is shown in the figure. The remaining operators in the figure are unique to subjective logic. The $\otimes$ operator is called the *recommendation* operator, where $\omega(\phi) \otimes \omega(\psi)$ yields an opinion about $\omega(\psi)$ given an opinion $\omega(\phi)$ about the source informing the computing agent of $\omega(\psi)$. The *consensus* operator $\oplus$, yields a combination of two opinions about the same

[1] We present a simplified form of subjective logic here, similar to the one presented in [Daskalopulu *et al.*, 2002].

$$\neg\omega(\phi) = \langle d(\phi), b(\phi), u(\phi)\rangle$$
$$\omega(\phi) \otimes \omega(\psi) = \langle b(\phi)b(\psi), b(\phi)d(\psi), d(\phi) + u(\phi) + b(\phi)u(\psi)\rangle \text{ where } \phi \text{ is an opinion about } \psi$$
$$\omega_A(\phi) \oplus \omega_B(\phi) = \langle \frac{b_A(\phi)u_B(\phi)+u_A(\phi)b_B(\phi)}{k}, \frac{d_A(\phi)u_B(\phi)+u_A(\phi)d_B(\phi)}{k}, \frac{u_A(\phi)u_B(\phi)}{k}\rangle, k = u_A(\phi) + u_B(\phi) - u_A(\phi)u_B(\phi)$$

Figure 1: The Subjective Logic negation, recommendation and consensus operators

thing. Note that the consensus operator is undefined when no uncertainty in opinions exists.

3 Formalism

In this section, we present three strands of work that tie up to form our framework. At the lowest level lies the argumentation framework. It is here that we define arguments, and how they interact with each other. Afterwards, we present a dialogue game, as well as the agents which play the game and the heuristics they use for deciding which arguments to advance and which sensors to probe. Finally, we describe how to transform contracts into a form usable by our framework.

3.1 The Argumentation Framework

Many frameworks for argumentation ignore probabilistic and possibilistic concepts, focusing instead purely on the interactions (such as rebutting and undercutting attacks) between arguments. While important in their own right, applying frameworks like these to domains in which uncertain evidence exists is difficult. We thus propose a new argument framework which, while borrowing ideas from existing work, suggests a different way of thinking about arguments.

We define our language of discourse Σ as the set of literals and their negation, while Σ^+ is the set containing only unnegated literals. A literal l has an opinion $\omega(l) = \langle b_l, d_l, u_l\rangle$ associated with it.

An argument A is a tuple (P, c) where $P \in 2^\Sigma$ and $c \in \Sigma$. We assume that if a literal p appears in P, then $\neg p$ does not appear, and vice-versa. P represents the premises of an argument, while c is the argument's conclusion. A fact a can be represented by an argument $(\{\}, a)$ (which we will also write as $(, a)$). Let $Args(\Sigma)$ represent the set of all possible arguments in the language Σ.

Given a set of arguments and facts, as well as opinions stating how likely those facts are to be true, we would like to determine what conclusions we may reasonably be allowed to draw. This is done by repeatedly computing the conclusions of arguments from known (opinions about) facts, until nothing more can be deduced. At this stage, we can draw conclusions based on what literals are justified within the system.

A literal may be assigned a value in one of three ways. First, it could be the conclusion of an argument. Second, it might be based on a fact. Finally, it could be based on a combination of values from both arguments and facts.

Given an argument $A = (P, c)$, we assign an opinion to it using a slightly modified form of statistical syllogism. Let ω_{p_i} be the opinion for premise $p_i \in P$ [2]. Then $\omega(A) = \langle min(b_{p_i}), 0, 1 - b_{p_i}\rangle$ if c is not negated, and $\langle 0, min(b_{p_i}), 1 - b_{p_i}\rangle$ otherwise. This definition captures the intuition that an argument is as strong as its weakest element.

Facts are a special type of argument, as their conclusions may not be negated literals. An opinion $\omega(F)$ may be associated with a fact $F = (, l)$ where $l \in \Sigma^+$.

Given a set of arguments about a literal l and its negation, as well as a set of supporting facts, we may compute a final opinion regarding the value of the literal by utilising the consensus operator $\oplus$. Slightly more formally, assuming we are given a set of arguments and facts S as well as their opinions $\omega(s), \forall s \in S$, we may compute an opinion $\omega(l)$ for a literal l as $\bigoplus_s \omega(s) \bigoplus_t \neg\omega(t)$ for all $s = (P_s, c_s), t = (P_t, c_t) \in S$ such that $c_s = l$ and $c_t = \neg l$. A special case arises when no uncertainty exists. We handle this case in an intuitively justifiable manner as follows: given opinions $\langle b_1, d_1, 0\rangle$ and $\langle b_2, d_2, 0\rangle$, we compute $b_c = b_1 + b_2 - d_1 - d_2$ and create the combined opinion $\langle b_c, 0, 1-b_c\rangle$ if $b_c > 0$, and $\langle 0, -b_c, 1+b_c\rangle$ otherwise.

We have shown how to compute the value of a literal given all the applicable arguments and facts, as well as opinions on their values. To complete the description of the argumentation framework, we must show how we can determine whether an argument is applicable, as well as give a procedure to decide the values of all literals in an instance of a dialogue.

Models of varying complexity have been proposed regarding whether an argument can be advanced [Toulmin, 1958]. In this paper, we use a simple model stating that if the premises of an argument hold, it can be used. For a premise to hold, it should exceed some threshold strength. Both computing a premise's strength, as well as setting the threshold are domain specific tasks. For example, the strength of a literal could be based on its level of uncertainty, strength of belief, or some other combination of its attributes (see [Jøsang, 2002] for more details). Terms from human legal reasoning such as "beyond reasonable doubt" and "on the balance of probabilities" show us the subjective nature of the threshold. Nevertheless, we assume that some function $admissible(\omega) \rightarrow [true, false]$ exists and allows us to compute whether an opinion exceeds the required threshold.

Before describing the algorithm to determine the state of literals formally, we provide an informal overview. We can represent the set of arguments under consideration as a directed graph containing two types of nodes; those representing arguments, and those representing literals. An edge connects literals to arguments in which they appear as premises, while edges also exist from arguments to those literals which appear in their conclusion. As shown in Figure 3, such a graph has two types of edges, namely edges linking negative literals to positive ones (or vice-versa), and all other edges. One weakening assumption we make (discussed further in Section 5) is that the resulting graph is acyclic.

Apart from the arguments, we assume that a set of facts has

[2] Note that if p_i is a negated literal, then the subjective logic negation operator must be applied when computing its contribution to the argument's opinion.

been put forward. Our algorithm operates by repeatedly forward chaining applicable arguments from the set of facts. In other words, we compute what literals we can derive from the facts, and propagate those forward into the arguments, computing literals from the derived arguments, and repeating this procedure until nothing new can be computed. Two things complicate this algorithm: accrual of arguments/evidence, and the threshold of acceptability of literals. We cannot compute an opinion regarding the value of a literal until all the arguments and facts regarding the literal have been taken into account. Similarly, we cannot determine the status of an argument until the value of all its premises have been computed. As mentioned previously, if a literal's strength falls below a certain level of acceptability, the literal cannot be applied.

To deal with these issues, a node's opinion is only computed once all influencing opinions have been calculated. If a literal falls below the threshold of acceptability, we erase all the arguments in which it appears as a premise. A literal exceeding this threshold is referred to as $admissible$.

More formally, given a set of arguments A, and defining the non-negated set of all literals appearing in A as L, we create a directed graph of arguments $GA = (A, L; PE, NE)$ with nodes A and L. For convenience, we write $\mathbf{E} = PE \cup NE$.

An edge e appears in PE, going from $a \in A$ to $l \in L$ if a is of the form (P, l). An edge would also appear in PE, going from $l \in L$ to $a \in A$ if $a = (P, c)$ and $l \in P$. An edge will appear in NE, going from $a \in A$ to $l \in L$ if a is of the form $(P, \neg l)$. Alternatively, an edge will be a member of NE, going from $l \in L$ to $a \in A$ if $a = (P, l)$ and $\neg l \in P$. In other words, an edge is in NE if it links the negated and non-negated form of a literal, and is in PE otherwise. Differentiating between these types of edges allows us to compute an opinion correctly while only storing non-negated literals in our graph.

Given a set of opinions on facts $\omega(F)$ where $F \subseteq A$ with the restriction that $\forall f = (P, c) \in F, P = \{\}$, a graph representing a set of arguments generated as described above, and an admissibility function mapping opinions to truth and falsehood, we can perform reasoning according to the algorithm shown in Figure 2.

Given a graph $G = (A, L; PE, NE)$,
a set of opinions Ω over $a = (, c) \subseteq A$ where $c \in \Sigma$
an admissibility function $admissible$

$sourceNodes(t) = \{s | (s, t) \in \mathbf{E}\}$
$targetNodes(s) = \{t | (s, t) \in \mathbf{E}\}$

repeat
 for each $\omega(n) \in \Omega$ such that $n \notin V$
  $V = V \cup n$
  for each $t \in targetNodes(n)$
   if $sourceNodes(t) \subseteq V$
    compute $\omega(t)$
    if $t \in L$ and $\neg admissible(\omega(t))$
     for each $r \in targetNodes(t)$
      for each $s \in targetNodes(r)$
       $\mathbf{E} = \mathbf{E} \setminus \{(r, s)\}$
      for each $s \in sourceNodes(r)$
       $\mathbf{E} = \mathbf{E} \setminus \{(s, r)\}$
     $A = A \setminus targetNodes(t)$
    $\Omega = \Omega \cup \{\omega(t)\}$
until $\nexists \omega(n) \in \Omega$ such that $n \notin V$
for each $l \in L$ such that $l \notin V$
 $\Omega = \Omega \cup \{\omega(l) = \langle 0, 0, 1 \rangle\}$

Figure 2: The algorithm for propagating opinions through the argumentation network. Note that $\mathbf{E} = PE \cup NE$

3.2 Agents and the Dialogue Game

The argumentation framework described above allows us to statically analyse groups of arguments; letting us compute an opinion about specific literals. The process of arguing is dynamic, with agents introducing arguments at various points in the conversation. In this section, we specify what our agents should look like, what environment they argue within, and the protocol they use to argue with each other. This protocol is referred to as a dialogue game. The aim of our dialogue game is "fairness", that is, allowing an agent to utilise any arguments at its disposal to win. While many dialogue games provide restrictions on agent utterances so as to maintain the focus of a dispute, we leave this task to the heuristic layer of our framework.

An agent α, is a tuple $(Name, KB, \Omega_\alpha, G, C)$, consisting of its name, $Name$, a private knowledge base $KB \subseteq 2^\Sigma$ containing arguments, a private opinion base $\Omega_\alpha = \{\omega_\alpha(l_1), \ldots, \omega_\alpha(l_n)\}$ storing opinions about literals $l_1, \ldots, l_n \in \Sigma$, a function mapping the state of proven and unproven literals to a utility value $G : \Sigma, \Sigma^+ \rightarrow \Re$,and a record of how much probing actions performed to date have cost $C \in \Re$. Thus, for example, $G(\{\neg a, b\}, \{c, d\})$ would return theutility for the state where $\neg a$ and b are proven, and nothing is known about the status of c and d.

Agents operate within an environment. This environment is a tuple of the form $(Agents, CS, S, admissible, PC)$ where $Agents$ is a set of agents, and contains a record of all the utterances made by an agent ($CS \subseteq 2^{Args(\Sigma)}$), as well as a set of sensors S. The function $admissible$ operates over opinions, while PC maps probing actions to costs.

A sensor is a structure (Ω_s, Ω_p). The set Ω_s contains an opinion for a set of literals $L \subset \Sigma^+$, and represents the environment's opinion about the reliability of the sensor. The opinion for a literal $l \in L$, is $\omega_s(l) \in \Omega_s$. The set of probed literals Ω_p, stores the sensor's opinions regarding the value of the literal l, $\omega_p(l) \in \Omega_p$. We compute a sensor's opinion for l as $\omega_s(l) \otimes \omega_p(l)$.

Agents take turns to put forward a line of argument and probe sensors to obtain more information about the environment. Such an action is referred to as an agent's utterance. In each turn, the contents of an agent's utterance is added to the global knowledge base; any sensors probed are marked as such (a sensor may not be probed more than once for the value of a specific literal), and costs are updated. We are now in a position to formally define the dialogue game.

The utterance function $utterance : Environment \times Name \rightarrow 2^{Args(\Sigma)} \times Probes$ takes in an environment and

an agent (via its name), and returns the utterance made by the agent. The first part of the utterance lists the arguments advanced by the agent, while the second lists the probes the agent would like to undertake. $Probes \in 2^{S \times \Sigma^+}$ is the power set of all possible probes. The domain specific function $PC : 2^{Probes} \rightarrow \Re$ allows us to compute the cost of performing a probe.Probing costs are defined in this way to allow for discounts on simultaneous probes, which in turn allows for richer reasoning by agents.

We define a function $turn : Environment \times Name \rightarrow Environment$ that takes in an environment and an agent label, and returns a new environment containing the effects of an agent's utterances. Given an environment Env and an agent α, we define the turn function as follows: $turn(Env, Name) = (NewAgents, CS \cup Ar, NewSensors, PC)$ where $Ar, NewAgents$ and $NewSensors$ are computed from the results of the utterance function. If $utterance(Env, Name) = (Ar, Pr)$ then $NewAgents = Agents \setminus \alpha \cup (Name, KB, \Omega_\alpha, G, C + PC(Pr))$ and, $\forall (s, l) \in Pr$, where l is a literal sensor s is able to probe, $NewSensors = Sensors \setminus s \cup (\Omega_s, \Omega_p \cup \omega_p(l))$.

It should be noted that the $utterance$ depends on agent strategy; we will define one possible $utterance$ function in the next section. Before doing so, we must define the dialogue game itself.

We may assume that our agents are named $Agent_0, Agent_1, \ldots, Agents_{n-1}$ where n is the number of agents participating in the dialogue. We can define the dialogue game in terms of the turn function by setting $turn_0 = turn((Agents, CS_0, S, admissible, PC), Agent_0)$, and then having $turn_{i+1} = turn(turn_i, Agent_{i \bmod n})$. The game ends when $turn_i \ldots turn_{i-n+1} = turn_{i-n}$.

CS_0 and Ω_0 contains any initial arguments and opinions, and are usually empty. Note that an agent may make a null utterance $\{,\}$ during its move to (eventually) bring the game to an end. In fact, given a finite number of arguments and sensors, our dialogue is guaranteed to terminate as eventually, no utterance will be possible that will change the public knowledge base CS.

We can compute an agent's utility by combining its utility gain (for achieving its goals) with the current costs. At any stage of the dialogue, given the environment's CS, and the set of all opinions probed by the sensors $\{\Omega_p | s = (\Omega_s, \Omega_p) \in S)$, and the environment's admissibility function, we can run the reasoning algorithm to compute a set of proven literals as $proven = \{l | l$ is a literal in CS and $admissible(l)\}$. Literals which fall below the proven threshold, or for which nothing is known are unproven: $unproven = \{$ literals in $CS \setminus proven\}$. Then the net utility gain for an agent α is $U(\alpha, proven) = G(proven, unproven) - C$.

At the end of the dialogue, we assume that agents agree that literals in the $proven$ set hold in the world.

3.3 The Heuristic

Agents are able to use the underlying argumentation framework and dialogue game to argue. In this section, we will describe a heuristic which agents may use to argue. Informally, this heuristic attempts to make utterances which maximise agent utility using one step lookahead. If multiple arguments still remain, one is selected at random.

When making an utterance, an agent has to decide what arguments to advance, as well as what probing actions it should undertake. Given an environment Env and an agent α, let the set of possible arguments be $PA = 2^{KB}$. We define the set of all possible probes as $PP = \{(s, l) | s = (\Omega_s, \Omega_p) \in S$ such that $\omega_s(l) \in \Omega_s$ and $\omega_p(l) \notin \Omega_p\}$. The cost for a $probe \subseteq PP$ is $PC(probe)$.

Our agents use multiple sources of information to estimate the results of a probe. First, they have opinions regarding literals, stored in Ω_α. Second, they can compute the current most likely value of the probe by examining sensors that already contain $\omega_p(l)$. Lastly, they can infer the most likely value of l by computing the effects of a probing action.

A naive agent will believe that an unprobed sensor s will return an opinion $\omega_\alpha(l)$. A non-naive agent will alter its beliefs based on existing probed values. One possible combination function will have the agent believe that the sensor will return an opinion $\bigoplus_{t \in P}(\omega_s(l) \otimes \omega_p^t(l)) \oplus \omega_\alpha(l)$ for a literal l given an agent's personal belief $\omega_\alpha(l)$, a set of sensors P containing $\omega_p^t(l)$ for any $t \in P$, and the unprobed sensor's $\omega_s(l)$. Informally, this means that a naïve agent will believe that a probed sensor will share the same opinion of a literal as the agent has, while a non-naïve agent will weigh its expected opinion due to probes that have already been executed. Different combination functions are possible due to different weights being assigned to different opinions.

For each possible utterance in $PA \times 2^{PP}$, the agent calculates what the environment will look like, as well as the net utility gain (using the functions PC and U). It then selects the utterance that will maximise its utility in the predicted future environment.

3.4 Contracts

Having fully defined the environment in which argument can take place, we now examine one possible application domain, namely contract monitoring. Since contract representations are highly domain dependent, most of this section provides only guidelines as to how to transform contracts into a form suitable for use within the framework. To simplify the discussion, only two party contracts are examined.

To make use of the framework, a number of things must be defined. The (domain dependent) sensor set S and probing cost function PC must be initialised, and the agent's goal function G must also be specified. The agent's private knowledge base KB and the environment's starting knowledge base CS_0 must also be populated, as well as the agent's prior opinions.

While nothing can be said about the generation of the sensor set except that it should reflect the sensors within the environment, probing costs can often be obtained from the contract. Contract based sanctions and rewards can also be used to generate G. For example, if in a certain contract state an agent α must pay β a certain amount of money, the literal representing that state might have an associated negative utility for α, and positive utility for β. One point that should be noted when assigning utility to various states is the unevenness of burden of proof [Prakken, 2001]. Assume that β gains

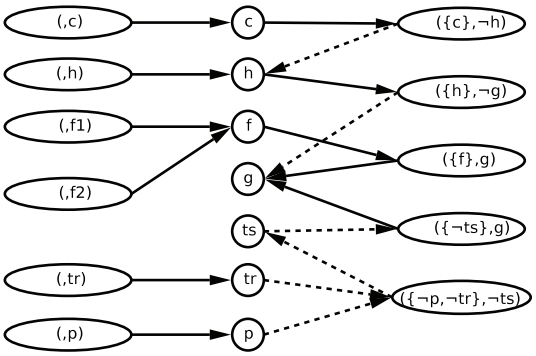

Figure 3: An argument graph. Dashed lines represent negated edges, while solid lines are normal edges.

utility for proving a literal l, while α gains if the first agent does not (e.g. an agent has to pay a penalty to another in a certain state). In such a situation, the second agent loses utility if l is true. Utility is not lost if l is either false or unproven.

We assume that a trivial mapping exists between contract clauses and arguments (for example, a clause stating "if a and b then c" can be represented as $(\{a, b\}, c)$). Then one must decide whether to place the translated clauses into the agent's private knowledge base KB, or CS_0. Putting them into KB will mean that agents will advance those clauses relevant to the dispute during the course of the argument. Having the arguments in CS_0 represents a more realistic scenario wherein the contract is public knowledge. KB can also contain domain specific arguments the agents might want to use in the course of the dialogue. Generating the agent's opinions about the state of literals is also outside the scope of our work.

4 Example

The example we present here is based on a simplified contract from the mobile multi–media domain. A supplier has agreed to provide a consumer with a set of multi–media services, and the consumer believes that the package it has received has not lived up to the the provider's contractual obligations. Assume that we have a contract of the form

$$frameRate < 25 \rightarrow giveSanction$$
$$horrorMovie \rightarrow \neg giveSanction$$
$$cinderella \rightarrow \neg horrorMovie$$
$$\neg textSent \rightarrow giveSanction$$
$$\neg phoneFull, \neg textReceived \rightarrow \neg textSent$$

From this, we can generate the argument graph shown in Figure 3, with the letter in the node being the first letter of the literal in the contract.

Assume further that we have sensors for whether the movie's name is "cinderella" (c), whether it is a horror movie (h), two sensors to detect whether the $frameRate$ is less than 25 ($f1, f2$), whether a text was received (tr), and whether the phone's memory was full (p). Let the opinion sensors for these ratings be $\omega_s(c) = \langle 1, 0, 0\rangle$, $\omega_s(h) = \langle 0.7, 0.2, 0.1\rangle$, $\omega_s(f1) = \langle 0.8, 0.1, 0.1\rangle$, $\omega_s(f2) = \langle 0.6, 0.2, 0.2\rangle$, $\omega_s(tr) = \langle 0.8, 0.0, 0.2\rangle$, $\omega_s(p) = \langle 1, 0, 0\rangle$, and let the cost of probing each of these sensors be 20, except for the sensor probing $f1$, which has a utility cost of 50.

Finally, let the utility for agent α showing g holds be 200, with agent β having utility -100 for this state. Assume α believes that literals $c, h, f, \neg tr, \neg p$ hold with opinion strength $\langle 0.9, 0, 0.1\rangle$. Also assume that the contract is initially stored in KB, and the admissibility function is $admissible(\langle b, d, u\rangle) = b > 0.5$ ($d > 0.5$) for a literal (or its negation) to be admissible.

Agent α might begin with the following utterance $(\{(, tr), (, p), (\{\neg p, \neg tr\}, \neg ts), (\{\neg ts\}, g)\}, \{p, tr\})$. This will cost it 40 utility. Given its reasoning algorithm, it will assume that once probed, $\omega(tr) = \langle 0, 0.72, 0.18\rangle$, $\omega(ts) = \omega(g) = \langle 0, 0.72, 0.18\rangle$. However, the sensor probing p returns $\langle 0.8, 0.1, 0.1\rangle$, making $\neg p$ fall below the threshold of acceptability, thus invalidating this chain of argument. Agent β passes.

Agent α tries again, probing $f1$ and advancing the argument $(\{f\}, g)$, which returns $\langle 0.7, 0.1, 0.2\rangle$ resulting in an opinion $\langle 0.56, 0.08, 0.36\rangle$ after the $\otimes$ operator is applied. This means that g is now valid, with opinion $\langle 0.64, 0.08, 0.28\rangle$.

Agent β counters, probing h and claiming $(\{h\}, \neg g)$. The probe returns $\langle 0.8, 0, 1, 0.1\rangle$, which means the argument is of sufficient strength to make g inadmissible (as $\omega(g) \approx \langle 0.47, 0.34, 0.19\rangle$).

Agent α undercuts β's argument by probing c and advancing $(\{c\}, \neg h)$. Given the sensor's opinion $\langle 0.6, 0.2, 0.2\rangle$, this reinstates g by disallowing argument $(\{h\}, \neg g)$.

Assume that β believes that f's true value is $\langle 0, 0.8, 0.2\rangle$. It would then probe $f2$ with the utterance $(, \{f2\})$. Assume this returns an opinion $\langle 0.1, 0.8, 0.1\rangle$. g would then remain admissible ($\omega(g) = \langle 0.51, 0.28, 0.21\rangle$).

Both agents now pass. Since g is admissible α has a net utility gain of 90, while β's utility loss is 140.

5 Discussion

As seen in the preceding example, our framework is able to represent standard argumentation concepts such as rebutting attacks, defeat and reinstatement, as well as higher level concepts such as accrual of arguments, which many frameworks have difficulty dealing with. When compared to standard argumentation systems, the main shortcoming of our approach lies in its inability to handle looping arguments. However, many other frameworks suffer from a similar (though weaker) problem [Prakken and Vreeswijk, 2002]. The use of possibilistic rather than probabilistic reasoning makes our framework suitable for situations where uncertainty exists regarding the probabilities to be assigned to an event. Our sample domain is one area where such reasoning capabilities are useful.

Most argument frameworks can be described in terms of Dung's [1995] semantics. These semantics are based on the notion of defeat. Since defeat in our framework is dynamically calculated, applying these semantics to our framework is difficult. Investigating the semantics of our framework more closely, and looking at how they compare to those of

standard argumentation frameworks is one interesting direction in which this research can be taken.

Some minor modifications of our framework might be necessary so that it can be used in other domains. For example, when arguing in the legal domain, the loser is often obliged to pay the winner's costs. By changing the way costs are computed in the turn function, our dialogue game is able to function in such an environment.

Our heuristic has high computational complexity. The unoptimised version presented here runs in exponential time ($O(2^n)$) for n arguments in the agent's knowledge base) and while optimisations are possible, exponential running time is still possible in the worst case. However, it is easy to generate solutions of increasing length (and thus probably increasing utility cost). By relaxing the requirement that the heuristic find the best possible move (given one move lookahead), computation can be stopped at any time. It is trivial to show that the algorithm presented in Figure 2 runs in $O(n)$ time where n is the number of graph edges. Extending our heuristic to look ahead further is possible, but requires some model of the opponent's knowledge base and utility costs.

As a contract monitoring mechanism, our framework is intended to operate "between" actions. Whenever a state change occurs, the agent should reason about whether a contract enforcement action should begin. If one does occur, it should finish before another state change happens. By integrating our framework with an expressive contract monitoring language, this limitation can be overcome.

Another useful avenue for future work involves examining how default arguments can best be incorporated in the framework. The presence of uncertainty allows us to represent many default rules with the simple addition of a conflicting literal edge. However, we have not examined how generic default rules can best be represented.

Our framework cannot deal with loops in arguments, and our main short term focus involves addressing this weakness. In the longer term, we hope to add opinions and weightings to the arguments themselves (for example stating that one argument is more persuasive than another). By also having a more complicated model for when an argument may be used (such as Toulmin's [1958] model of argument), this path of research should allow us to represent argument schemes [Walton and Krabbe, 1995] in our framework.

Finally, there are clear parallels between our work and the work done on trust and reputation frameworks [Teacy *et al.*, 2006]. We intend to investigate whether any of the work done on opinions in that field will be of benefit to our model.

6 Conclusions

In this paper we have put forward a new subjective logic based framework for analysing combinations of arguments and opinions. We then showed how this framework could be used to have agents discuss the state of literals in an environment where probing for environment state has an associated utility cost. After providing strategies for agents to argue in such an environment, we demonstrated its applicability as a technique for performing contract monitoring. While our framework has some limitations, it promises to be a springboard for further research in this area of argumentation, and has applications as a powerful mechanism for contract monitoring in complex domains.

7 Acknowledgements

This work is supported by the DTI/EPSRC E-Science Core Program and BT, via the CONOISE-G project (http://www.conoise.org).

References

[Daskalopulu *et al.*, 2002] A. Daskalopulu, T. Dimitrakos, and T. Maibaum. Evidence-based electronic contract performance monitoring. *Group Decision and Negotiation*, 11(6):469–485, 2002.

[Dung, 1995] Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, 77(2):321–357, 1995.

[Jøsang, 2002] A. Jøsang. Subjective evidential reasoning. In *Proc. of the 9th Int. Conf. on Information Processing and Management of Uncertainty in Knowledge-Based Systems*, pages 1671–1678, July 2002.

[Pollock, 1995] J. L. Pollock. *Cognitive Carpentry*. Bradford/MIT Press, 1995.

[Prakken and Sartor, 1998] Henry Prakken and Giovanni Sartor. Modelling reasoning with precedents in a formal dialogue game. *Artificial Intelligence and Law*, 6(2-4):231–287, 1998.

[Prakken and Vreeswijk, 2002] H. Prakken and G. Vreeswijk. Logics for defeasible argumentation. In *Handbook of Philosophical Logic, 2nd Edition*, volume 4, pages 218–319. Kluwer, 2002.

[Prakken, 2001] H. Prakken. Modelling defeasibility in law: Logic or procedure? *Fundamenta Informaticae*, 48(2-3):253–271, 2001.

[Prakken, 2005] Henry Prakken. A study of accrual of arguments, with applications to evidential reasoning. In *Proc. of the 10th Int. Conf. on Artificial Intelligence and Law*, pages 85–94, 2005.

[Teacy *et al.*, 2006] W. T. L. Teacy, J. Patel, N. R. Jennings, and M. Luck. Travos: Trust and reputation in the context of inaccurate information sources. *Autonomous Agents and Multi-Agent Systems*, 12(2):183–198, 2006.

[Toulmin, 1958] Stephen E. Toulmin. *The Uses of Argument*. Cambridge University Press, 1958.

[Vreeswijk, 2004] G. A. W. Vreeswijk. Argumentation in Bayesian belief networks. In *Proc. of ArgMAS 2004*, number 3366 in LNAI, pages 111–129. Springer, 2004.

[Walton and Krabbe, 1995] D. N. Walton and E. C. W. Krabbe. *Commitment in Dialogue*. State University of New York Press, 1995.

[Xu and Jeusfeld, 2003] L. Xu and M. A. Jeusfeld. *Proactive monitoring of electronic contracts*, volume 2681 of *LNCS*, pages 584–600. Springer, 2003.