Tractable Inquiry in Information-Rich Environments*

Barbara Dunin-Kęplicz, Alina Strachocka
Institute of Informatics, University of Warsaw
Banacha 2, 02-097 Warsaw, Poland
keplicz,astrachocka@mimuw.edu.pl

Abstract

In the contemporary autonomous systems the role of complex interactions such as (possibly relaxed) dialogues is increasing significantly. In this paper we provide a paraconsistent and paracomplete implementation of **inquiry dialogue** under realistic assumptions regarding availability and quality of information. Various strategies for dealing with unsure and inconsistent information are analyzed. The corresponding dialogue outcomes are further evaluated against the (paraconsistent and paracomplete) distributed beliefs of the group.

A specific 4-valued logic underpins the presented framework. Thanks to the qualities of the implementation tool: a rule-based query language 4QL, our solution is both expressive and tractable.

1 Paraconsistent Nonmonotonic Dialogues

The synergistic effect of collaborating agents is achieved by their proper communication. However, in dynamic and unpredictable environments up-to-date, sure and complete information is hardly obtainable. This leads to conflicts, uncertainty and paracompleteness, particularly when handling information originating from multiple sources of diverse credibility, quality or significance. In this paper we introduce a new approach to logical modeling of conversing agents, which are prepared to handle inconsistency and ignorance.

To this end, a paracomplete and paraconsistent (i.e., tolerating inconsistencies) logic is necessary, supported by two new truth values: *unknown* (u) and *inconsistent* (i). In line with other paraconsistent approaches to modeling dialogues [Takahashi and Sawamura, 2004; Prakken, 2010; Black and Hunter, 2009], inconsistency does not trivialize reasoning but is treated as first-class citizen alike *true* (t) and *false* (f). In our system the following choices have been made.

- The four-valued logic of [Vitória *et al.*, 2009] underpins the solution.
- Unknown or inconsistent conclusions do not enforce termination of the reasoning process.
- Such conclusions can be handled via lightweight forms of nonmonotonic reasoning.

Entailment in logic amounts to deriving conclusions from theories that can be seen as complex knowledge bases. However, instead of querying arbitrary theories, in order to reduce complexity we tailor them to their tractable versions like specific rule-based systems. Thus, instead of reasoning in logical systems of high complexity, we query paraconsistent knowledge bases. Only recently has a sufficiently expressive tool existed for creating and querying them in polynomial time: 4QL - a DATALOG$\neg\neg$-like four-valued rule-based query language. Following this shift in methodology, the contribution of this paper is an implementation of a tractable, paraconsistent and paracomplete multi-party **inquiry dialogue** suitable for agents situated in information-rich environments. The overall goal of inquiry is to collectively solve a theoretic problem, resulting in the common knowledge about the solution.

Consider a multi-agent system where each *swarm agent* is specialized in gathering different type of information via a polling system, and a *supervisor agent* which verifies certain information for the human user. Suppose the human user asks if it is safe to travel to place X ($safe(X)?$) and none of the agents knows the answer. Engaging in inquiry on the topic $safe(X)$ allows agents to share only the relevant pieces of their (possibly vast) knowledge and collectively arrive at a final recommendation for the human user. Although conflicts may naturally appear on many different levels of such group activity [Dunin-Kęplicz *et al.*, 2014], it is not the goal of inquiry but rather persuasion to resolve them.

Unlike the classical case [Walton and Krabbe, 1995], our approach to inquiry permits 4-valued statements. It turns out that the initial valuation of the topic separates *Inquiry-What* from *Inquiry-That*. In both cases, several strategies to handle missing and inconsistent information are presented and formally investigated. The final outcomes of such dialogues are compared against the (possibly inconsistent and incomplete) distributed knowledge of the conversing group [Fagin *et al.*, 1995]. In this regard the *soundness* of a strategy means that whenever a dialogue terminates with a given conclusion, the same result would be obtained by an individual reasoning from the union of all the agents' belief bases. Accordingly, if a solution is obtainable from the union of agents beliefs, an inquiry under a *complete* strategy will reach it. The main result of this research concerns soundness and completeness

*This research is partially supported by Warsaw Center of Mathematics and Computer Science.

of the *open-minded* inquiry strategy (Theorem 4).

Enriching the modeling perspective allows us to contemplate several new cognitive situations in communication (see e.g., [Dunin-Keplicz *et al.*, 2015]), occurring also in inquiry. Arguably, other normative models of dialogues would benefit from the 4-valued approach.

The paper is structured as follows. First, in Section 2, the notions underpinning our solution are recalled. Section 3 concerns the formalization of inquiry, its strategies and properties. Finally, Section 4 concludes the paper.

2 Language and Implementation Tool

Our formal inquiry dialogue system uses the logical language of [Małuszyński and Szałas, 2013; Szałas, 2013; Małuszyński and Szałas, 2011]. Agents' informational stance is encoded in the rule-based query language 4QL[1] defined in [Małuszyński and Szałas, 2013], further developed in [Szałas, 2013] and based on a 4-valued logic of [Vitória *et al.*, 2009]. 4QL allows for negation both in premisses and conclusions of rules. Importantly, negation in the conclusions may lead to inconsistencies. Even though openness of the world is assumed, rules can be used to close the world locally or globally. Below, the formal language underlying 4QL will be briefly introduced.

In what follows all sets are finite except for sets of formulas. We deal with the classical first-order language over a given vocabulary without function symbols. We assume that *Const* is a fixed set of constants, Var is a fixed set of variables and Rel is a fixed set of relation symbols.

Definition 1 A *literal* is an expression of the form $R(\bar{\tau})$ or $\neg R(\bar{\tau})$, $\bar{\tau}$ being a sequence of parameters, $\bar{\tau} \in (\mathit{Const} \cup Var)^k$, where k is the arity of $R \in Rel$. *Ground literals over Const*, denoted by $\mathcal{G}(\mathit{Const})$, are literals without variables, with all constants in *Const*. If $\ell = \neg R(\bar{\tau})$ then $\neg\ell \stackrel{\text{def}}{=} R(\bar{\tau})$.◁

Though we use classical first-order syntax, the semantics substantially differs from the classical one as truth values $\mathtt{t}, \mathtt{i}, \mathtt{u}, \mathtt{f}$ (true, inconsistent, unknown, false) are explicitly present; the semantics is based on sets of ground literals rather than on relational structures. Intuitively:

- a is $\mathtt{t}$ if all sources claim a,
- a is $\mathtt{f}$ if all sources claim $\neg a$,
- a is $\mathtt{u}$ if no sources claim a nor $\neg a$,
- a is $\mathtt{i}$ if some sources claim a, other claim $\neg a$.

For semantics of propositional connectives see Table 1. The definitions of $\wedge$ and $\vee$ reflect minimum and maximum with respect to the truth ordering

$$\mathtt{f} < \mathtt{u} < \mathtt{i} < \mathtt{t}. \tag{1}$$

Whenever truth values are restricted to $\{\mathtt{f}, \mathtt{t}\}$, the semantics is compatible with the semantics of classical first-order logic.

Let $v : \mathit{Var} \to \mathit{Const}$ be a *valuation of variables*. For a literal ℓ, by $\ell(v)$ we mean the ground literal obtained from ℓ by substituting each variable x occurring in ℓ by constant $v(x)$.

[1]Open-source implementation of 4QL is available at `4ql.org`.

Table 1: Truth tables for $\wedge$, $\vee$, $\to$ and $\neg$.

$\wedge$	f	u	i	t
f	f	f	f	f
u	f	u	u	u
i	f	u	i	i
t	f	u	i	t

$\vee$	f	u	i	t
f	f	u	i	t
u	u	u	i	t
i	i	i	i	t
t	t	t	t	t

$\to$	f	u	i	t
f	t	t	t	t
u	t	t	t	t
i	f	f	t	f
t	f	f	t	t

	$\neg$
f	t
u	u
i	i
t	f

Definition 2 The *truth value* $\ell(L, v)$ of a literal ℓ w.r.t. a set of ground literals L and valuation v, is defined by:

$$\ell(L, v) \stackrel{\text{def}}{=} \begin{cases} \mathtt{t} & \text{if } \ell(v) \in L \text{ and } (\neg\ell(v)) \notin L; \\ \mathtt{i} & \text{if } \ell(v) \in L \text{ and } (\neg\ell(v)) \in L; \\ \mathtt{u} & \text{if } \ell(v) \notin L \text{ and } (\neg\ell(v)) \notin L; \\ \mathtt{f} & \text{if } \ell(v) \notin L \text{ and } (\neg\ell(v)) \in L. \end{cases}$$

◁

For a formula $\alpha(x)$ with a free variable x and $c \in \mathit{Const}$, by $\alpha(x)^x_c$ we understand the formula obtained from α by substituting all free occurrences of x by c. Definition 2 is extended to all formulas in Table 2, where α denotes a first-order formula, v is a valuation of variables, L is a set of ground literals, and the semantics of propositional connectives appearing at righthand sides of equivalences is given in Table 1.

Table 2: Semantics of first-order formulas.

- if α is a literal then $\alpha(L, v)$ is defined in Definition 2;
- $(\neg\alpha)(L, v) \stackrel{\text{def}}{=} \neg(\alpha(L, v))$;
- $(\alpha \circ \beta)(L, v) \stackrel{\text{def}}{=} \alpha(L, v) \circ \beta(L, v)$, where $\circ \in \{\vee, \wedge, \to\}$;
- $(\forall x \alpha(x))(L, v) = \min_{a \in \mathit{Const}} (\alpha^x_a)(L, v)$, where $\min$ is the minimum w.r.t. ordering (1);
- $(\exists x \alpha(x))(L, v) = \max_{a \in \mathit{Const}} (\alpha^x_a)(L, v)$, where $\max$ is the maximum w.r.t. ordering (1).

In 4QL beliefs are distributed among *modules*. Each module can be treated as a finite set of literals. For specifying rules, multisource formulas and querying modules, we apply the language of [Szałas, 2013].

Definition 3 A *multisource formula* is an expression of the form: $m.A$ or $m.A \in T$, where:

- m is a module name;
- A is a first-order or a multisource formula;
- v is a valuation;
- $T \subseteq \{\mathtt{t}, \mathtt{i}, \mathtt{u}, \mathtt{f}\}$.

We write $m.A = v$ (respectively, $m.A \neq v$) to stand for $m.A \in \{v\}$ (respectively, $m.A \notin \{v\}$). ◁

The intuitive meaning of a multisource formula $m.A$ is:

> "return the answer to query expressed by formula A, computed within the context of module m".

The value of '$m.A \in T$ ' is:

$$\begin{cases} \mathsf{t} & \text{when the truth value of } A \text{ in } m \text{ is in the set } T; \\ \mathsf{f} & otherwise. \end{cases}$$

Let $A(X_1, \ldots, X_k)$ be a multisource formula with $X_1, \ldots, X_k$ being its all free variables and D be a finite set of literals (a belief base). Then A, understood as a query, returns tuples $\langle d_1, \ldots, d_k, tv \rangle$, where $d_1, \ldots, d_k$ are database domain elements and the value of $A(d_1, \ldots, d_k)$ in D is tv.

From now on we assume that the domain and language are fixed and the programs and rules are ground. If S is a set, then $\text{FIN}(S)$ represents the set of all finite subsets of S. In what follows let $\mathbb{C} \stackrel{\text{def}}{=} \text{FIN}(\mathcal{G}(\textit{Const}))$ be the set of all finite sets of ground literals over constants in *Const*.

Definition 4

- *Rules* are expressions of the form:

$$\ell :\!- b_{11}, \ldots, b_{1i_1} \mid \ldots \mid b_{m1}, \ldots, b_{mi_m}. \qquad (2)$$

  where the *conclusion* ℓ is a positive or negative literal and the *premisses* $b_{11}, \ldots, b_{1i_1}, \ldots, b_{m1}, \ldots, b_{mi_m}$ are multisource formulas and ',' and '|' abbreviate conjunction and disjunction, respectively.
- A *fact* is a rule with empty premisses (evaluated to t).
- A *module* is a syntactic entity encapsulating a finite number of facts and rules.
- A 4QL *program* is a set of modules, without cyclic references to modules involving multisource formulas of the form $m.A \in T$. ◁

If δ is a rule, by $head(\delta)$ we mean the rule conclusion. If δ is a fact, $head(\delta) = \delta$.

The key concepts of modules and multisource formulas allow us to deal with unknown or inconsistent conclusions without enforcing termination of the reasoning process. Technically, any literal l corresponds to a multisource formula $M.l$, thus $l \in M$.

The semantics of 4QL is defined by *well-supported models* [Małuszyński and Szałas, 2013; Szałas, 2013], i.e., models consisting of (positive or negative) ground literals, where each literal is a conclusion of a derivation starting from facts. For any set of rules, such a model is uniquely determined and computable in deterministic polynomial time $O(N^k)$ where N is the size of domain and $k = max(s, t)$ where s is the maximal arity of relations and t is the maximum number of free variables. As we deal with ground programs, $t = 0$. When s is a bound constant, which is the case in practical applications of 4QL (qualitative not quantitative reasoning), we achieve tractability. Notice that it is the same complexity as SQL with recursion.

Definition 5 Let P be a 4QL program, A a formula, and $\mathcal{M}_P$ the well-supported (unique) model of P. Then: $P \models A$ iff for any valuation v we have $\mathcal{M}_P \models v(A)$.

As an example, consider program $P = \{top, su\}$ consisting of two modules top and su (for surveillance).

$$\begin{array}{rll} top = \{ & enter(b) & :\!- \; isAt(s,b), \neg has(s,h). , \\ & isAt(s,b) & :\!- \; isArmed(s), hearShotsAt(b). , \\ & isAt(s,b) & :\!- \; su.isAt(s,b) \in \{\mathsf{u}, \mathsf{i}, \mathsf{t}\}. , \\ & & \quad \neg has(s,h), \\ & & \quad has(s,h), \\ & & \quad isArmed(s)\} \\ su = \{ & isAt(s,b) & :\!- \; see(s,b), \neg conditions(fog). , \\ & & \quad see(s,b), \\ & & \quad \neg conditions(fog)\} \end{array} \qquad (3)$$

The literals s, b, h represent $suspect, building$ and $hostage$, respectively. The program uniquely determines the following well-supported model for module su:

$$\mathcal{M}_{su} = \{\neg conditions(fog), see(s,b), isAt(s,b)\} \qquad (4)$$

and the following well-supported model for module top:

$$\begin{array}{l} \mathcal{M}_{top} = \{enter(b), \neg enter(b), isAt(s,b), \\ isArmed(s), has(s,h), \neg has(s,h)\}. \end{array} \qquad (5)$$

Definition 6 Let ℓ be a literal and P a 4QL program. A *derivation* of ℓ from P is the well-supported model $\mathcal{M}_P$.

A dependence set of a literal ℓ from a program P consists of literals reachable via backward chaining on P from ℓ.

Definition 7 Let ℓ be a literal and P a 4QL program. A *dependence set* of ℓ from P, denoted $\mathcal{D}_{P,\ell}$ is a set of literals such that:

- $\neg\ell, \ell \in \mathcal{D}_{P,\ell}$,
- if there is a rule $\ell' :\!- b_{11}, \ldots, b_{1i_1} \mid \ldots \mid b_{m1}, \ldots, b_{mi_m}$ in P, such that $\ell' \in \mathcal{D}_{P,\ell}$ then $\forall_{j \in 1..m} \forall_{k \in 1..i_j} \; b_{jk}, \neg b_{jk} \in \mathcal{D}_{P,\ell}$. ◁

A proof of a literal ℓ from a program P is a subprogram S of P generated from the dependence set $\mathcal{D}_{P,\ell}$ by taking all rules and facts of P whose conclusions are in $\mathcal{D}_{P,\ell}$.

Definition 8 Let ℓ be a literal, P a 4QL program. A *proof* of l from P is a 4QL program $S \subseteq P$ such that $\delta \in S$ iff $head(\delta) \in \mathcal{D}_{P,\ell}$, where δ is a fact or a rule. The *size of the proof S* is the size of the program S. The *size of domain of the proof S* is the size of the dependence set $\mathcal{D}_{P,\ell}$. ◁

In order to implement dialogues, the functionality of adding a rule to a 4QL program is required.

Definition 9 We define an operation of *adding a ground rule* $M_i.\ell :\!- b$ to a 4QL program $P = \{M_1, ..., M_n\}$ as follows:
$P' = P \cup \{M_i.\ell :\!- b\} =$
$\{M_1, ..., M_{i-1}, M_i \cup \ell :\!- b, M_{i+1}, ..., M_n\}$

3 Inquiry

The purpose of inquiry is to collectively solve a theoretical problem [Walton and Krabbe, 1995]. In multi-agent systems, inquiry *"starts when some agents are ignorant about the solution to some question or open problem. The main goal is the growth of knowledge, leading to agreement about the conclusive answer of the question. This goal may be attained*

in many different ways, including an incremental process of argument which builds on established facts in drawing conclusions beyond a reasonable doubt. Both information retrieval and reasoning may be intensively used in this process" [Dunin-Kęplicz and Verbrugge, 2010]. In its classical form, inquiry seeks to prove a statement as true or false:

I1. $is(suspect, guilty) = \mathfrak{t}$: 'prove that suspect is guilty',

I2. $is(suspect, guilty) = \mathfrak{f}$: 'prove that suspect is not guilty'.

Our paraconsistent and paracomplete framework allows for contemplating other possibilities:

I3. $is(suspect, guilty) = \mathfrak{i}$: 'prove that there are inconsistent information concerning suspect's guilt',

I4. $is(suspect, guilty) = \mathfrak{u}$: 'is suspect guilty?'.

Specifically (3) exemplifies that inquiry can commence when the goal's initial valuation is $\mathfrak{i}$. Our approach allows to model such situations, common in practical applications. In contrast, until a valuation for (4) is established, no classical inquiry on this subject (finding a proof) can commence. This scenario resembles discovery dialogue, where *"we want to discover something not previously known"* [McBurney and Parsons, 2001]. In our setting, the dialogue aiming at discovering the value of a statement is just another variation of inquiry, so is structured exactly the same. Thus, two types of inquiry dialogues are distinguished:

1. **Inquiry-WHAT**, where initial valuation of s is $\mathfrak{u}$ and the goal of the dialogue is to establish the valuation v_f of s (see **I4**).
2. **Inquiry-THAT**, where initial valuation of s is $\mathfrak{t}$, $\mathfrak{f}$ or $\mathfrak{i}$, and the goal of the dialogue is to confirm or refute this by providing the proof for s (see **I1**, **I2**, **I3**).

Inquiry-WHAT succeeds if the final valuation $v_f \neq \mathfrak{u}$ and Inquiry-THAT succeeds if v_f is equal to the initial valuation v_i of s. An outcome of a successful inquiry is the valuation of the goal and the proof of it (see Definiton 13).

3.1 Query and Commitment Stores

A common approach to modeling inquiry is to keep two stores (see e.g., [Black and Hunter, 2009; Singh, 1998] and references therein): a *Commitment Store* (CS), reflecting *agents' commitments*, and a *Query Store* (QS), reflecting *current open questions*. Commitment Store is thus associated with an individual agent while Query Store - with the dialogue.

We maintain both stores associated with the dialogue (CS_d, QS_d) without any assumptions about agents' individual Commitment Stores. Technically, our Commitment Store reflects the current accumulated knowledge base. It is created empty when the dialogue begins (as no locutions have been uttered yet) and updated with every assertion relevant to the dialogue. To sum up, in our methodology, the inquiry Commitment Store is just an evolving 4QL program (see also [Alferes *et al.*, 2002]).

We assume that in the course of dialogue agents assert only relevant information, that is, rules whose conclusions match the current entries in QS. For a literal $l \in QS$, relevant responses include both l :– b and $\neg l$:– b (or ℓ and $\neg\ell$). Accordingly, two locutions crucial to inquiry are:

- $assert(S_i, \delta, d)$: agent S_i asserts a rule or a fact δ in the dialogue d. If assertion is relevant, it's content is added to CS_d and its premisses are added to QS_d.
- $requestAll(S_i, d)$: agent S_i requests content of QS_d.

Backward chaining mechanism employed here is commonly used in deductive argumentation for driving the argumentation process (see e.g., [Besnard and Hunter, 2008; Black and Hunter, 2009; Prakken, 2010]).

Definition 10 Locution m^t is *relevant* to an inquiry d at time t iff $m^t = assert(S, M_i.\ell$:– $b, d)$ and $(\neg)M_i.\ell \in QS_d^t$, where QS_d^t is the Query Store of d at time t. We will alternate between the notions of locution, message, move and utterance. ◁

To make the communication more flexible, the assumption about relevance of the locutions can be realized by a filtering mechanism. Then, instead of requiring that agents make specific moves, we allow them to utter any locutions, filtering out the irrelevant ones[2].

Definition 11 *Commitment Store* of a dialogue d at time t is a 4QL program denoted as $CS_d^t = \langle M_1^t, \ldots, M_k^t \rangle$:

- $CS_d^0 = \emptyset$
- $CS_d^t = CS_d^{t-1} \cup \{M_i.\ell$:– $b\}$, such that $m^t = assert(S, M_i.\ell$:– $b, d)$ is relevant to d at time t,
- $CS_d^t = CS_d^{t-1}$ otherwise. ◁

Next, the Query Store, is a repository of active, unresolved leads in the inquiry. It contains literals which compose the proof of the inquiry goal s. At the beginning the Query Store contains s as a single entry. The mechanism of updating QS is in fact a paraconsistent and paracomplete distributed version of backward chaining[3], as discussed in Section 3.3. However, in contrast to the classical backward chaining, here we have a number of additional options to investigate. Consequently, there may be various policies for adding literals to QS (selecting threads to follow) and removing them from QS (closing explored threads). Functions $open$ and $close$ (see Definition 12) correspond to such methods.

Definition 12 Let:

- CS_d^t be the Commitment Store of dialogue d at time t,
- m^t be the message received at time t,
- $close$: $\textsc{Fin}(\mathbb{C}) \times \textsc{Fin}(\mathbb{C}) \to \textsc{Fin}(\mathbb{C})$ be a method for removing entries from the Query Store,
- $open$: $\textsc{Fin}(\mathbb{C}) \times \textsc{Fin}(\mathbb{C}) \to \textsc{Fin}(\mathbb{C})$ be a method for adding entries to Query Store.

Then, *Query Store* of an inquiry dialogue d on subject s at time t is a finite set of literals denoted as QS_d^t such that:

- $QS_d^0 = \{s\}$

[2]Such a filter is easy to implement: upon receiving a message, QS is inspected to verify if the rule head is in the scope of inquiry.

[3]Hybrid backward-forward chaining techniques may be used if $assert$ locution contains a set of rules, e.g., a subset of proof constructed bottom-up. This is a topic for future research.

- $QS_d^t = (QS_d^{t-1} \cup B') \setminus B''$, if $m^t = assert(S, M_i.\ell :\!- b, d)$, where
$$B' = open(b, CS_d^t),$$
$$B'' = close(QS_d^{t-1} \cup B', CS_d^t),$$
- $QS_d^t = QS_d^{t-1}$ otherwise. ◁

3.2 Dialogue Outcome vs. Distributed Knowledge

Our setting consists of a finite set of n cooperative agents. The assumption that agents do not withhold information implicitly constraints the number of *requestAll* locutions per one assertion. Agents' belief bases are encoded as finite, ground 4QL programs $P_1, \ldots, P_n$, that share a common ontology and do not change during the course of dialogue. Agents communicate one-to-all without coordination. The well-supported models $\mathcal{M}_{P_1}, \ldots, \mathcal{M}_{P_n}$ of the programs express agents' final beliefs. The union of individual agents' belief bases (i.e., their distributed knowledge [Fagin *et al.*, 1995]) is expressed by the sum of their 4QL programs: $\bigcup_{i\in 1..n} P_i$. An agent can join and leave a dialogue at any time if in between $join$ and $leave$ locutions it utters at least one $assert$ion. Agents cannot repeat assertions. These assumptions allow us to verify quality and completeness of the obtained results.

Since 4QL programs are finite and agents cannot repeat utterances, there must be a moment t when no agent has anything more to utter because either it has run out of relevant moves or because the dialogue goal s has been achieved, whichever comes first. Thus, dialogue terminates at time t. The knowledge accumulated in the course of a dialogue d is expressed by the Commitment Store of that dialogue at termination time t: CS_d^t. The final conclusion depends on the dialogue strategy (see below) and is expressed as follows.

Definition 13 For an inquiry terminating at time t, with the goal s of initial valuation v_i, the value of the dialogue conclusion is $v_f = v(s, \mathcal{M}_{CS_d^t})$, where $\mathcal{M}_{CS_d^t}$ is the well-supported model of CS_d^t. Dialogue is:

- successful iff
  - $v_i = \mathfrak{u} \wedge v_f \neq \mathfrak{u}$ [Inquiry-WHAT], or
  - $v_i \neq \mathfrak{u} \wedge v_f = v_i$ [Inquiry-THAT],
- unsuccessful otherwise. ◁

The value of the goal s obtained from the union of agents' programs is expressed as $v(s, \mathcal{M}_{\bigcup_{i\in 1..n} P_i})$.

Definition 14 Let:

- $open : \textsc{Fin}(\mathbb{C}) \times \textsc{Fin}(\mathbb{C}) \to \textsc{Fin}(\mathbb{C})$,
- $close : \textsc{Fin}(\mathbb{C}) \times \textsc{Fin}(\mathbb{C}) \to \textsc{Fin}(\mathbb{C})$

be two methods for adding and removing entries to Query Store of dialogue d. Then: $ST = \langle open, close \rangle$ is a *strategy*.

Definition 15 A strategy ST is *sound* iff whenever dialogue d on subject s conducted under this strategy terminates at t with conclusion k, then if $v(s, \mathcal{M}_{CS_d^t}) = k$ then $v(s, \mathcal{M}_{\bigcup_{i\in 1..n} P_i}) = k$.

Definition 16 A strategy ST is *complete* iff whenever dialogue d on subject s conducted under this strategy terminates at t with conclusion k, then if $v(s, \mathcal{M}_{\bigcup_{i\in 1..n} P_i}) = k$ then $v(s, \mathcal{M}_{CS_d^t}) = k$.

3.3 Opening and Closing Inquiry Threads

In classical backward chaining, the inference engine selects rules whose consequents match the goal to be proved. If the antecedent of the rule is not known to be true, then it is added to the list of goals. In our paraconsistent and nonmonotonic distributed version of backward chaining, the conditions under which antecedent can be added to the list of goals differ depending on the method used. Consequently, there may be various policies for adding literals to QS (selecting threads to follow via function $open$). From a variety of possibilities, here we investigate two such methods. A literal can be added to the Query Store if:

A1. **Its valuation in the CS model is $\mathfrak{u}$**, meaning that only threads lacking any evidence whatsoever are explored.

A2. **Always**, meaning that *every* premise is investigated further, even one that is tentatively assumed to be $\mathfrak{t}$, $\mathfrak{f}$ or $\mathfrak{i}$.

Definition 17 Let CS_d^t be the Commitment Store of an inquiry dialogue d at time t and $\mathcal{M}_{CS_d^t}$ be its well-supported model. Let $m^t = assert(S, M_i.\ell :\!- b, d)$ be the message received at time t, such that: $b = b_{11}, \ldots, b_{1i_1} \mid \ldots \mid b_{m1}, \ldots, b_{mi_m}$. Then,

$$open(b, CS_d^t) \stackrel{\text{def}}{=} \begin{cases} \{b_{j_k} | j \in 1..m, k \in 1..i_j \\ \text{ and } \mathcal{M}_{CS_d^t}(b_{j_k}) = \mathfrak{u}\} & [A1] \\ \{b_{j_k} | j \in 1..m, k \in 1..i_j\} & [A2] \end{cases}$$

◁

Notice that in the nonmonotonic paraconsistent backward-chaining, obtaining a truth value for p does not necessarily close the line of reasoning about p, since the evidence put forward by other agents may change the value of p in a number of ways. This is why we conduct inquiry until all relevant information is shared by the agents.

The conditions under which a goal can be abandoned also differ depending on the policy employed. We distinguish two methods for removing literals from QS (closing explored threads via function $close$):

R1. **Once its valuation in the CS model is not $\mathfrak{u}$,** meaning that a thread is terminated whenever any evidence for it is found. In some cases it may be closed prematurely, without exposing other evidence relevant to the thread.

R2. **Never**, meaning the threads are never abandoned, as the information regarding them may grow. This will not lead to infinite dialogues, since agents cannot repeat utterances and their programs do not change during dialogue.

Definition 18 Let CS_d^t be the Commitment Store of an inquiry dialogue d at time t and $\mathcal{M}_{CS_d^t}$ be its well-supported model. Let QS_d^{t-1} be the Query Store of an inquiry dialogue d at time $t-1$ and $\mathcal{M}_{QS_d^{t-1}}$ be its well-supported model. Then,

$$close(QS_d^{t-1}, CS_d^t) \stackrel{\text{def}}{=} \begin{cases} \{x \in \mathcal{M}_{QS_d^{t-1}} | \mathcal{M}_{CS_d^t}(x) \neq \mathfrak{u}\} & [R1] \\ \emptyset & [R2] \end{cases}$$

3.4 Inquiry Strategies

The ensuing question is which combination of methods for updating QS makes sense (see Table 3) and how do resulting inquiry strategies differ. Unlike other approaches, we do not assume that the distributed knowledge of the group is complete. If the statement s cannot be proved by agents, the conclusion would simply be $\mathbf{u}$.

Table 3: Inquiry strategies defined as pairs of methods for updating QS.

	R1	**R2**
A1	narrow-minded	pragmatic
A2	forgetful	open-minded

Theorem 1 *Narrow-minded strategy is neither sound nor complete. Moreover, it is type 1 nondeterministic.*[4]

Proof. Due to the non-monotonicity of our inquiry, applying the narrow-minded strategy may result in overlooking some important information. As the counterexample, assume three agents A_1, A_2, A_3 are engaged in an inquiry dialogue with the goal $enter(b)$. Their programs are shown in Table 4 and the dialogue conduct is presented in Table 5.

Table 4: Programs of Agents A_1, A_2, A_3.

	A_1	A_2	A_3
1	$enter(b) \text{ :– } isAt(s,b), \neg has(s,h)$	$\neg su.isAt(s,b)$	$hearShotsAt(b)$
2	$isAt(s,b) \text{ :– } su.isAt(s,b) \in \{\mathbf{u}, \mathbf{i}, \mathbf{t}\}$	$isArmed(s)$	
3	$isAt(s,b) \text{ :– } isArmed(s), hearShotsAt(b)$		
4	$\neg has(s,h)$		

Table 5: Example of a Narrow-Minded Inquiry.

t	QS^t_d	m_t	$\mathcal{M}_{CS^t_d}$
0	$enter(b)$	$\emptyset$	$\emptyset$
1	$enter(b), isAt(s,b), has(s,h)$	$A_1(1)$	$\emptyset$
2	$enter(b), has(s,h), su.isAt(s,b)$	$A_1(2)$	$isAt(s,b)$
3	$enter(b), has(s,h)$	$A_2\ (1)$	$\neg su.isAt(s,b)$
4	$enter(b)$	$A_1(4)$	$\neg su.isAt(s,b), \neg has(s,h)$

For brevity, we denote assertions in Table 5 as $A_j(k)$, standing for the k-th rule of agent A_j. Dialogue terminates in step 4, since only agent A_1 has a rule with conclusion $enter(b)$ but it has already uttered it. Notice that at timepoint $t = 2$ we had to remove $isAt(s,b)$ from the Query Store, as it became true in $\mathcal{M}_{CS^2_d}$. Therefore, agent A_1 didn't have a chance to use rule (3) in the dialogue. Obviously, $v(enter(b), \mathcal{M}_{CS^4_d}) = \mathbf{u}$, whereas the conclusion obtained by merging agents' programs is $v(enter(b), \mathcal{M}_{\bigcup_{i\in 1..3} P_i}) = \mathbf{t}$. If instead in the timepoint $t = 2$ agent A_1 would have uttered rule (3), then Query Store and in consequence, the whole dialogue, would look differently, leading to a true conclusion even if agent A_1 didn't have a chance to utter rule (1). ◁

Two strategies are equal if the dialogues conducted under these strategies cannot be distinguished on the basis of the content of the stores at any time.

Definition 19 Dialogue D_1 is *equal* to dialogue D_2 iff for all finite sequences of moves $s = m^1, \ldots, m^{t_s}$, s.t. m^i is relevant at i to D_1 and to D_2, we have that $\forall i \in 1..t_s$ $CS^i_{D_1} = CS^i_{D_2}$ and $QS^i_{D_1} = QS^i_{D_2}$.

Strategies S_1 and S_2 are *equal* iff dialogues conducted under these strategies are equal. ◁

Theorem 2 *Forgetful and narrow-minded strategies are equal.*

Proof sketch. In the forgetful strategy, we add *all* literals from the rule body to QS only to remove the *known* ones afterwards. Therefore, what remains are the $unknown$ literals. Since agents cannot query QS in between adding and removing literals (in theory update of QS is atomic operation), these two strategies are indistinguishable. ◁

Theorem 3 *Pragmatic and open-minded strategies are equal in terms of dialogue conduct.*

Proof. Let's consider the pragmatic strategy and a goal s. In the first step, the rule s :– b is considered. All rule premisses (b) are either empty (when s is a fact) or unknown (since CS^0 is empty). Therefore, in the first step all premisses (b) are added to QS^0 and the initial rule s :– b (or fact s) is added to CS^0. Obviously for a literal to be $\mathbf{t}$, $\mathbf{f}$ or $\mathbf{i}$, it has to be a rule conclusion or a fact. Since only rules, whose conclusions are in QS are admitted to CS, there cannot be a $\mathbf{t}$, $\mathbf{f}$ or $\mathbf{i}$ literal which is in CS but was not in QS beforehand. ◁

Theorem 4 *Open-minded strategy is sound and complete.*

Proof sketch. Assume that $v(s, \mathcal{M}_{CS^t_d}) = k$ and $v(s, \mathcal{M}_{\bigcup_{i\in 1..n} P_i}) \neq k$. At the time of dialogue termination, CS contains all relevant messages. Each of these was uttered by at most one agent. Therefore, we can assign each message to a set CS_i where i was the sender. Obviously, $CS_i \subseteq P_i$. Therefore we have: $CS = \bigcup_{i\in 1..n} CS_i \subseteq \bigcup_{i\in 1..n} P_i$. Since $v(s, \mathcal{M}_{CS^t_d}) = k$ and $v(s, \mathcal{M}_{\bigcup_{i\in 1..n} P_i}) \neq k$, that means that there is a part of the union of programs $S \stackrel{\text{def}}{=} \bigcup_{i\in 1..n} P_i \setminus CS$, such that, adding S to CS would change the valuation of s. However, that would mean that there exists a rule (or a fact) in S whose conclusion is in premisses of CS. That means, that rule is a part of the proof for s but was not uttered by the agent, which contradicts our assumptions.

Proof of completeness is analogous. ◁

Notice that in open-minded inquiry on subject s, CS is the evolving proof of s from $\bigcup_{i\in 1..n} P_i$ and QS is the evolving dependence set of s from $\bigcup_{i\in 1..n} P_i$.

[4] Type 1 nondeterminism in logic programs means freedom to choose the rule to apply [Schöning, 2008].

3.5 Complexity

Complexity measures of proposed inquiry strategies include:

- *communication complexity*, concerning only the amount of communication among agents (who have unlimited computational power) [Kushilevitz and Nisan, 1997],
- *computational complexity* (data complexity), concerning the amount of computation (when communication is free) required to:
  - achieve dialogue termination,
  - obtain a conclusion of a terminated dialogue.

Computational complexity of both problems is expressed in terms of *data complexity* [Vardi, 1982; Papadimitriou and Mihalis, 1997], i.e., complexity of evaluating a fixed query (here: inquiry goal) on an arbitrary database (CS). Thus data complexity is given as a function of the size of CS.

In what follows we deal with terminated dialogues and thus we write CS and QS instead of CS^t_d and QS^t_d, respectively. Since open-minded strategy subsumes narrow-minded (Theorems 1 and 4), the (pessimistic) communication complexity results of open-minded strategy hold for both (see Table 6).

Theorem 5 *If the size of the domain of the proof of s is N, then the size $|QS|$ of the Query Store at the end of the open-minded inquiry is $N/2 \leq |QS| \leq N$.*

Proof. Since all literals from rule bodies are added to QS and they are never removed from QS, in fact they all take part in proving the goal s. Moreover, negative and positive literals from the proof are added to QS only once (either l or $\neg l$). ◁

Theorem 4 allows us to conclude:

Theorem 6 *If the size of the proof of s is M, then the size $|CS|$ of the Commitment Store at the end of the open-minded inquiry is $|CS| = M$.*

Proof. The total amount of information shared by all $assert$ locutions (a_i denotes number of assertions by agent i) uttered in the dialogue is:

$$\sum_{i=1}^{n} \sum_{j=1}^{a_i} |1| = |CS| = M$$

◁

Recall that n denotes the total number of agents, each holding a certain amount of (relevant) information, such that the proof of the inquiry topic from the union of all agents' belief bases is of size M (from Theorem 6). The communication complexity is polynomial in the total amount of information relevant to the proof.

Theorem 7 *Communication complexity of inquiry is $O(nM)$.*

Proof. In general there can be up to $n-1$ $requests$ per one $assert$. Thus, there can be at most M $asserts$ (agents cannot repeat assertions), $M \times (n-1)$ $requests$ and at most 2 $join$ and $leave$ locutions per one $assert$. Altogether $(n+2) \times M$ locutions exchanged before dialogue termination[5]. Therefore the communication complexity is $O(nM)$. ◁

[5]Notice that even for hybrid forward-backward chaining, this is the pessimistic time complexity.

Recall that the size of the domain of the proof is N, which is the upper limit on the size of Query Store (see Theorem 5).

Theorem 8 *Computational complexity of a narrow-minded inquiry is: $M \times O(N^k)$.*

Proof. In the narrow-minded strategy, after each $assert$ the well-supported model of the CS has to be computed, which is in $O(N^k)$ (see Section 2). Thus each such step takes $O(N^k)$. However, at the termination time, the conclusion is known (obtainable in $O(1)$). Computational complexity of narrow-minded inquiry is thus $M \times O(N^k)$. ◁

Theorem 9 *Computational complexity of termination of open-minded inquiry is: $O(1)$.*

Proof. Handling each $assert$ amounts to adding a rule to CS^t, which is in $O(1)$. Handling each $request$ is in $O(1)$ as it amounts to sending the whole QS^t back to the agent. ◁

Theorem 9 shows that the major factor in the complexity of the termination problem of the open-minded inquiry is the communication complexity.

Theorem 10 *Obtaining the conclusion of a terminated open-minded inquiry is $O(N^k)$.*

Proof. Recall that computing the well-supported model of CS is in $O(N^k)$, where N is the size of domain. For open-minded strategy the computation of the well-supported model is only needed after the dialogue terminates, i.e., once per dialogue. ◁

Characteristics	**Open-minded**	**Narrow-minded**
Open vs. Closed System	open (at least one $assert$ per $join$)	
Addressing	one-to-all	
Coordination	asynchronous	
Properties	sound and complete	not sound and not complete
Communication Complexity	$O(nM)$	$O(nM)$
Computational Complexity (Termination)	$O(1)$	$O(MN^k)$
Computational Complexity (Obtaining Conclusion)	$O(N^k)$	$O(1)$
Total Store Size	$M + N$	

Table 6: Results for open- and narrow-minded inquiries

4 Related Work and Conclusions

Exploring paraconsistency and paracompleteness in argumentation is not new: there is a number of formalisms that do not trivialize when inconsistent premises (for a survey see [Walton *et al.*, 2008; Besnard and Hunter, 2008]). In [Black and Hunter, 2009] a formal bi-party inquiry dialog system is proposed where DeLP is used to deal with ignorance and inconsistency. In [Takahashi and Sawamura, 2004] the logic of multi-valued argumentation (LMA) is used and agents can argue using multi-valued knowledge base. In [Prakken, 2010] ASPIC+, a framework for structured argumentation with possible inconsistent knowledge bases and defeasible rules is given. However, none of these formalisms

handles inconsistency and ignorance the way 4QL does. Usually the inconsistent premisses yield conclusions (e.g., 'undecided') which cannot be further dealt with.

As indicated in [Dignum and Vreeswijk, 2003; Traum, 2004], several new issues arise when contemplating the plurality of dialogue participants. Multi-party issues were also studied in [Yuan *et al.*, 2011], where a distributed argumentation system was given together with a multi-party dialogue game for computing the defensibility of an argument from consistent knowledge bases. In [Vreeswijk and Hulstijn, 2004], a simple multi-party inquiry dialogue assumed communication in turns with no termination criterion.

Leaving behind the realm of two-valued logical approaches to bi-party dialogues, we arrived at a solution for multi-party, paraconsistent and paracomplete inquiry. We investigated four inquiry strategies, conditional on different policies for opening and closing threads. The relevant results were evaluated against the paraconsistent and paracomplete distributed knowledge of the group.

The general outcome of our research calls for reconsidering normative models of dialogues by introducing two additional logical values: $\mathfrak{i}$ and $\mathfrak{u}$. Specifically, the novelty lies in understanding the very nature of the dialogue's goal, leading to a better discernment between inquiry and discovery and more applications of inquiry.

In future work, we intend to investigate hybrid forward-backward chaining techniques for a dialogue system, where the locutions can contain a set of rules. Next, we plan to research methods for handling inconsistencies and uncertainty in the Commitment Store via a $challenge$ locution.

5 Acknowledgments

The authors would like to thank Andrzej Szałas for his comments which greatly improved this paper.

References

[Alferes *et al.*, 2002] J. J. Alferes, A. Brogi, J. A. Leite, and L. M. Pereira. Evolving logic programs. In *Proceedings of JELIA 2002*, volume 2424 of *LNCS*, pages 50–61. Springer, 2002.

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

[Black and Hunter, 2009] E. Black and A. Hunter. An inquiry dialogue system. *Autonomous Agents and Multi-Agent Systems*, 19(2):173–209, 2009.

[Dignum and Vreeswijk, 2003] F. Dignum and G. Vreeswijk. Towards a testbed for multi-party dialogues. In *Workshop on Agent Communication Languages*, volume 2922 of *LNCS*, pages 212–230. Springer, 2003.

[Dunin-Kęplicz and Verbrugge, 2010] B. Dunin-Kęplicz and R. Verbrugge. *Teamwork in Multi-Agent Systems: A Formal Approach*. Wiley, 2010.

[Dunin-Kęplicz *et al.*, 2014] B. Dunin-Kęplicz, A. Szałas, and R. Verbrugge. Tractable reasoning about group beliefs. In *2nd international Workshop on Engineering Multi-Agent Systems (EMAS 2014)*, LNAI. Springer, 2014.

[Dunin-Keplicz *et al.*, 2015] Barbara Dunin-Keplicz, Alina Strachocka, Andrzej Szalas, and Rineke Verbrugge. Paraconsistent semantics of speech acts. *Neurocomputing*, 151:943–952, 2015.

[Fagin *et al.*, 1995] R. Fagin, J. Y. Halpern, Y. Moses, and M. Y. Vardi. *Reasoning about Knowledge*. The MIT Press, 1995.

[Kushilevitz and Nisan, 1997] Eyal Kushilevitz and Noam Nisan. *Communication Complexity*. Cambridge University Press, New York, NY, USA, 1997.

[Małuszyński and Szałas, 2011] J. Małuszyński and A. Szałas. Living with inconsistency and taming nonmonotonicity. In *Datalog Reloaded*, volume 6702 of *LNCS*, pages 384–398. Springer-Verlag, 2011.

[Małuszyński and Szałas, 2013] J. Małuszyński and A. Szałas. Partiality and inconsistency in agents' belief bases. In *KES-AMSTA*, volume 252 of *Frontiers in Artificial Intelligence and Applications*, pages 3–17. IOS Press, 2013.

[McBurney and Parsons, 2001] P. McBurney and S. Parsons. Chance discovery using dialectical argumentation. In *New Frontiers in Artificial Intelligence*, volume 2253 of *LNCS*, pages 414–424. Springer, 2001.

[Papadimitriou and Mihalis, 1997] Christos H. Papadimitriou and Yannakakis. Mihalis. On the complexity of database queries. In *Proceedings of the sixteenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems*. ACM, 1997.

[Prakken, 2010] H. Prakken. An abstract framework for argumentation with structured arguments. *Argument and Computation*, 1(2):93–124, 2010.

[Schöning, 2008] Uwe Schöning. *Logic for Computer Scientists*. Modern Birkhäuser Classics. Birkhäuser Boston, 2008.

[Singh, 1998] M. P. Singh. Agent communication languages: Rethinking the principles. *Computer*, 31(12):40–47, December 1998.

[Szałas, 2013] A. Szałas. How an agent might think. *Logic Journal of IGPL*, 21(3):515–535, 2013.

[Takahashi and Sawamura, 2004] T. Takahashi and H. Sawamura. A logic of multiple-valued argumentation. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pages 800–807. IEEE Computer Society, 2004.

[Traum, 2004] D. Traum. Issues in multiparty dialogues. *Advances in Agent Communication*, pages 201–211, 2004.

[Vardi, 1982] Moshe Y. Vardi. The complexity of relational query languages (extended abstract). In *Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing*, STOC '82, pages 137–146, New York, NY, USA, 1982. ACM.

[Vitória *et al.*, 2009] A. Vitória, J. Małuszyński, and A. Szałas. Modeling and reasoning with paraconsistent rough sets. *Fundamenta Informaticae*, 97(4):405–438, 2009.

[Vreeswijk and Hulstijn, 2004] G.A.W. Vreeswijk and J. Hulstijn. A free-format dialogue protocol for multi-party inquiry. In *In Proc. of the Eighth Int. Workshop on the Semantics and Pragmatics of Dialogue (Catalog '04)*, pages 273–279, 2004.

[Walton and Krabbe, 1995] D. N. Walton and E. C. W. Krabbe. *Commitment in Dialogue: Basic Concepts of Interpersonal Reasoning*. State University of New York Press, Albany (NY), 1995.

[Walton *et al.*, 2008] D. Walton, C. Reed, and F. Macagno. *Argumentation Schemes*. Cambridge University Press, 2008.

[Yuan *et al.*, 2011] Jinping Yuan, Li Yao, Zhiyong Hao, Fang Liu, and Tangming Yuan. Multi-party dialogue games for distributed argumentation system. In *IAT*, pages 329–332. IEEE Computer Society, 2011.