# Qiana: A First-Order Formalism to Quantify over Contexts and Formulas

**Simon Coumes**[1] , **Pierre-Henri Paris**[1] , **François Schwarzentruber**[2] , **Fabian M. Suchanek**[1]

[1]Telecom Paris, Institut Polytechnique de Paris
[2]Université de Rennes, IRISA, CNRS
{simon.coumes, pierre-henri.paris, fabian.suchanek}@telecom-paris.fr,
francois.schwarzentruber@ens-rennes.fr

**Abstract**

We introduce Qiana, a logic framework for reasoning on formulas that are true only in specific contexts. In Qiana, it is possible to quantify over both formulas and contexts to express, e.g., that "everyone knows everything Alice says". Qiana also permits paraconsistent logics within contexts, so that contexts can contain contradictions. Furthermore, Qiana is based on first-order logic, and is finitely axiomatizable, so that Qiana theories are compatible with pre-existing first-order logic theorem provers.

## 1 Introduction

In his "Notes on formalizing contexts" John McCarthy (1987) argued for the importance of context representation in formal logic. The core idea is that statements can be tied to specific contexts, which act as modalities on the statements. This idea is substantiated by the predicate *ist*: $ist(c, \varphi)$ means that the formula $\varphi$ is true in the context $c$. Contexts can represent different things: something can be true in the context of a newspaper article only, in the context of a piece of fiction, or in someone's beliefs. We illustrate one possible use of contexts with the final scene of the play "Romeo and Juliet" by William Shakespeare:

> *Near the end of the play, Juliet wishes to meet with Romeo, but her parents won't let her. Her friend, Friar Laurence, offers her a potion and says it will allow her to fake her death. Juliet takes the potion, hoping it will allow her to escape her family. However, the plan backfires: Romeo sees Juliet before she awakens, seemingly dead, and kills himself in despair. When Juliet later wakes up, she sees Romeo dead and kills herself.*

The key elements of the ending of the play are: (1) Friar Laurence is right in what he says (the potion will make Juliet appear dead), and (2) someone who is madly in love with someone else will kill themselves if they believe their loved one to be dead. Thus, leaving out details and overgeneralizing, we want to represent:

$$\forall \phi.\ \textit{ist(says(FriarLaurence)}, \phi) \rightarrow \phi$$

$$\forall x, y.\ \textit{madlyLoves}(x, y) \wedge \textit{ist(believes}(x), \textit{dead}(y)) \rightarrow \textit{willSuicide}(x)$$

Here $\textit{believes}(x)$ is the context of the beliefs of the agent $x$. Our example leads us to the following desiderata for expressivity:

1. **Truth Representation:** the ability to connect what is true inside a context to what is true ("$ist(c, \varphi) \rightarrow \varphi$")
2. **Formula Quantification:** the ability to quantify over formulas ("$\forall \varphi.\ ist(c, \varphi)$")
3. **Context Quantification:** the ability to quantify over contexts ("$\forall c.\ ist(c, \varphi)$"), or even certain forms of context ("$\forall x.\ \textit{ist}(\textit{believes}(x), \varphi)$")

Moreover, we want to perform automated reasoning, or at least semi-automated reasoning:

4. **Semi-decidability:** Logical entailment $\Gamma \models \phi$ should be semi-decidable.

Fulfilling these desiderata simultaneously is not trivial. One difficulty is that Desideratum 1 invites complications from the Theorem of Undefinability of Truth of Tarski (1936): A language cannot fully describe its own truth, assuming it includes basic arithmetic. This is because it allows self-referential statements, which leads to contradictions.

One way to do contextual reasoning in logic is through modal logic. However, modal logic does not consider formulas as objects that one can quantify over. Another classical way would be to use higher-order logics, but these (typically) quantify over predicates rather than the syntactic formulas themselves. Furthermore, they are usually not even semi-decidable. Moore (1981) proposes to *quote* formulas as terms within the logic. However, the notion of context in this approach is very restrictive. For example, it lacks a dedicated mechanism to express statements as simple as "*If Juliet believes all Capulets are nice, then for any Capulet $x$, she believes $x$ is nice*".

It seems that the promising idea of using object-level counterparts to formulas within first-order logic was never explored to produce a suitable framework for this form of general contextual reasoning. Thus, to the best of our knowledge, no logical framework currently satisfies all 4 desiderata simultaneously (see Table 1, discussed in the related work section).

In this paper, we propose to represent formulas within contexts not as special objects, but as terms that obey a specific axiomatization. We borrow the *ist* predicate from

McCarthy (1993). We follow the idea of Moore (1981) to build terms that are structurally similar to formulas. (This idea is itself an extension of Gödel's numbers, see Gödel (1931).) We use the idea of Tarski (1936) to introduce a special truth predicate and to make sure that this predicate cannot be quoted. We then show how these components can be axiomatized so that Desiderata 1-4 are fulfilled without falling for the complications of Tarski's theorem. The resulting framework, Qiana (Quantifying over Agents and Assertions), is finitely axiomatizable, and can thus be used with any First-Order-Logic theorem prover. We also introduce a special character *quote* to nest quotations within quotations. This allows for a larger array of manipulations around contexts, which are notably useful for our finite axiomatization process.

Qiana can model the beliefs of agents (as in our Romeo and Juliet example), but it can also be used for paraconsistent reasoning (where a context contains contradictory statements), or to describe the differences between two fictional contexts (e.g., two versions of the same story).

In what follows, Section 2 discusses the related work; Section 3 introduced notations; Section 4 explains how Qiana quotes formulas; Section 5 defines Qiana; Section 6 discussed applications of Qiana; Section 7 describes the finite axiomatization process of Qiana, which is then used in a theorem prover in Section 8; Section 9 contains additional discussions; and Section 10 concludes. Supplementary material, including the proofs of our theorems and the code of our implementation, is available at https://github.com/dig-team/Qiana.

## 2 Related Work

John McCarthy (1987) observed that many statements are true only in a specific context. Several follow-up works have elaborated on this idea, but none of them allows for Context Quantification and Formula Quantification. The first of these elaborations was by McCarthy (1993) himself. He proposed to write $\mathit{ist}(c, p)$ to say that $p$ is a proposition that is true in the context $c$. Thus, contexts are treated as objects representing a state of the universe at a given instant. However, this work was based on propositional logic. Hence, it cannot deal with first-order formulas, let alone quantify over contexts or formulas.

Buvac and Mason (1993) and Buvac, Buvac, and Mason (1994) formalized a propositional modal logic version of McCarthy's idea, which is sound, complete, and decidable. In their formalism, *ist* is treated as a binary modality over propositions. Again, there is no possibility of quantifying over contexts or formulas. Buvac (1996) extended this work to first-order logic and allowed the description of contexts through properties. For instance, $\forall c.\ p(c) \rightarrow \mathit{ist}(c, \phi)$ means that the formula $\phi$ is true in every context with the property $p$. This logic is sound and complete; the work was the first to allow quantification over contexts. However, unlike our approach, all contexts must have perfect knowledge of each other's beliefs, i.e., everyone knows what everyone else thinks. Furthermore, unlike our approach, Buvac (1996) does not allow for quantification over formulas.[1]

Moore's work on reasoning about knowledge (Moore, 1980, 1981) avoids the issues of self-reference in higher-order logic by representing formulas as terms within the logic. A special truth predicate connects these terms to their formula counterparts. This will also be done in Qiana. However, Moore's notion of context is quite restrictive: its many-worlds semantics assumes that contexts are logically omniscient (if something is true within a context, then all its consequences are also true). This is unsuitable to represent the knowledge of humans, whose reasoning depth is limited. It also lacks an equivalent to our special escape function symbol *quote*, which is used to put any given value into a quotation. Such a feature is important to present axioms that connect what is true outside of contexts to what is true within them, e.g., for statements of the form "If in a context it is true that a statement holds for all $x$, then that statement holds for all $x$ in that context". Furthermore, there is no finite axiomatization, and thus, the method does not allow the use of state-of-the-art theorem provers that Qiana permits.

Other works fall in the realm of epistemic and doxastic logics, which deal with the knowledge and beliefs of agents, respectively. The modal approach has been widely adopted for both cases (Hintikka, 1962). For instance, Halpern and Moses (1992) proposed a multi-modal logic to deal with the knowledge and beliefs of multiple agents, where each agent has its own operators. For example, $\mathsf{K}_i\phi$ means that the agent $i$ knows $\phi$, and $\mathsf{B}_i\phi$ means that $i$ believes $\phi$. Considering only the knowledge operators, this logic is equivalent to the formalism of Buvac and Mason (1993), where each context is equivalent to a specific modality. However, these modal approaches have no way to quantify over contexts. Furthermore, these approaches focus on propositional logic and cannot deal with first-order formulas like Qiana.

Giunchiglia (1993) and Giunchiglia and Bouquet (1997) treat each context as a logical theory with its own language, set of axioms, and set of rules. The main goal in this series of works is the translation of formulas from one context to another. The works in this series study only the propositional case and introduce no quantification. Ghidini and Giunchiglia (2001) propose that contexts need two principles: locality (what is known by the agent) and compatibility (enforcing a kind of coherence in viewpoints). While this approach can deal with first-order formulas, it does not allow for quantified formulas or quantified contexts.

The Knowledge Interchange Format KIF (Genesereth, 1991) is a data format for database knowledge exchange. With the help of a quotation operator, a formula can be reified and handled as a syntactic element. However, KIF does not admit any complete proof theory. It is not even semi-decidable because it goes beyond first-order logic (Väänänen, 2021). KIF's successor, Common Logic (ISO, 2018) (CL), is a framework for a family of FOL-based languages. Unlike KIF, CL has no quotation operator, and it does not have a built-in mechanism for handling contexts. While

[1] According to (Guha, McCool, and Fikes, 2004), it is also not semi-decidable. However, (Buvac, 1996) contains proofs of completeness and soundness, which entails semi-decidability.

| | Truth Representation | Formula Quantification | Context Quantification | Semi-decidable |
|---|---|---|---|---|
| (Moore, 1981) | yes | yes | NA | yes |
| (Genesereth, 1991) | yes | yes | yes | no |
| (Halpern and Moses, 1992) | yes | no | no | yes |
| (McCarthy, 1993) | yes | no | no | NA |
| (Giunchiglia, 1993) | NA | no | no | NA |
| (Buvac and Mason, 1993) | NA | no | no | yes |
| (Buvac, 1996) | NA | no | yes | no |
| (Ghidini and Giunchiglia, 2001) | yes | no | no | yes |
| (Perrussel, 2002) | no | no | yes | no |
| (Ranganathan and Campbell, 2003) | NA | no | no | yes |
| (Carroll et al., 2005) | yes | no | no | yes |
| (Brewka et al., 2011) | no | no | no | yes |
| (ISO, 2018) | yes | no | yes | NA |
| (Aljalbout, Buchs, and Falquet, 2019) | yes | no | no | yes |
| (Väänänen, 2021) | NA | no | NA | no |
| (Hartig et al., 2023) | yes | no | no | yes |
| Qiana | yes | yes | yes | yes |

Table 1: Semi-decidability and the desiderata of Truth Representation, Formula Quantification, and Context Quantification

some subsets of CL admit a complete proof theory, there is still no complete proof theory for Common Logic as a whole (ISO, 2018; Mossakowski et al., 2014; Menzel, 2013). Suchanek (2005) proposes a translation from KIF to disjunctive logic programs, but also does not offer a complete proof theory. Perrussel (2002) proposes a many-sorted modal first-order logic. This approach cannot quantify over formulas, or express formulas such as $\mathit{ist}(c, \phi) \rightarrow \phi$ since no "super context" represents the real world.

In the work of Ranganathan and Campbell (2003), contexts are first-order predicates like *Location* or *Temperature*. Hence, the approach cannot deal with quantified formulas. Brewka et al. (2011) propose an approach to deal with different sources of knowledge. Each context is a knowledge base with its language, and bridge rules allow communication between contexts and handle inconsistencies. Intrinsically, it is not possible to quantify over formulas or contexts. More recently, Aljalbout, Buchs, and Falquet (2019) proposed a two-dimensional ontology language that allows defining context-dependent classes, properties, and axioms. It also allows expressing knowledge about contexts to reason on contextualized triples. However, it is impossible to quantify over formulas or contexts. Furthermore, the work uses description logics, which has limited expressiveness w.r.t. first-order logic.

Other approaches of the Semantic Web, like RDF-star (Hartig et al., 2023) and named graphs (Carroll et al., 2005), can handle context but not truth representation. They can handle neither quantification over contexts nor over formulas. One may think that second-order logic (Väänänen, 2021) could be of help. However, classical higher-order logics allow quantification over predicates, not over formulas. Taprogge and Steen (2023) extend the prover Leo-III with a form of higher-order modal logic, but still does not allow quantification over formulas.

We thus conclude that no semi-decidable framework currently satisfies the desiderata of Truth Representation, Formula Quantification, and Context Quantification.

## 3 Notations

Our work relies on the usual notions of first-order logic (FOL) (see, e.g., Enderton (2001) for a primer). We use standard syntactic sugar notations, writing, e.g., $\varphi \rightarrow \psi$ for $\neg\varphi \vee \psi$. We also use the usual substitution meta-notation: $\varphi[x \leftarrow t]$ denotes the formula obtained by recursively replacing all occurrences of variable $x$ with $t$ in $\varphi$ until a quantification over $x$ is reached.

For our purposes, a *signature* $S$ is a tuple $(F, P, V_\infty, \delta)$, where $F$ is a set of function symbols, $P$ is a set of predicate symbols, $V_\infty$ is an infinite set of variables, and $\delta : P \cup F \rightarrow \mathbb{N}$ a function that gives the arity of each symbol. Constant symbols are function symbols of arity 0. A given signature defines a set $\mathcal{T}$ of terms, and a set $\mathcal{L}$ of formulas.

A *model* is a tuple $(D, [\![]\!])$ where $D$ is a non-empty set called the *domain of the model*, and $[\![]\!]$ is the interpretation mapping that maps each function symbol $f$ to a function $[\![f]\!] \in D^{\delta(f)} \rightarrow D$, and each predicate symbol $p$ to a function $[\![p]\!] \in D^{\delta(p)} \rightarrow \{0, 1\}$. Given an assignment for the free variables $\sigma$, we recall that terms (e.g., $1 + x$) are interpreted as elements in the domain (e.g., $1 + x$ is interpreted as the element $[\![+]\!]([\![1]\!], \sigma(x))$), and formulas (e.g., $p(x)$) are interpreted as true/false (e.g., the semantics of $p(x)$ is $[\![p]\!](\sigma(x))$, which is either 0 or 1).

Recall that a theory is a set of formulas, and an axiom schema is a formula with meta-variables (such as $\neg\neg\varphi \rightarrow \varphi$, where $\varphi$ is a meta-variable that stands for a formula). We will occasionally write the name of the axiom schema to stand for the set of all its instantiations in a given signature.

Let $M$ be a model, $\sigma$ an assignment of values in the domain of $M$ to free variables, and $H$ a theory. We write

$M, \sigma \models \phi$ to say that the formula $\phi$ is true in model $M$, where all the free variables of $\phi$ are defined in $\sigma$. We omit $\sigma$ if there are no free variables. We write $H \models \phi$ to say that $\phi$ is a semantic consequence of $H$. A closed formula that is true in at least one model is coherent. A theory for which there is a model that makes all the formulas true is also called coherent.

## 4 Quoting and unquoting formulas

A quoted formula is a term that represents a formula of the logic; they will be useful to represent what is true or false in a context. Intuitively speaking, quoting a formula consists of replacing each logical connective, variable, predicate, and function symbol with a fresh function symbol. We denote the quoted counterpart of a symbol $z$ by $\underline{z}$:

**Example 1.** *The quotation of formula $p(x) \wedge (1 + x = 2)$ is the term $\underline{p}\,(\underline{x}) \underline{\wedge} (\underline{1} \underline{+} \underline{x} \underline{=} \underline{2})$ where $\underline{p}, \underline{\wedge}, \underline{1}, \underline{+}, \underline{x}, \underline{=}, \underline{2}$ are "quotation" symbols, counterparts to the original symbols $p, \wedge, 1, +, x, =, 2$.*

We need to quote also formulas that already contain quotations. To this end, we introduce a special function symbol *quote*, which acts as an escape character and provides a way to nest quotations:

**Example 2.** *The quotation of the formula $ist(x, \underline{happy}(\underline{y}))$ is $\underline{ist}(\underline{x}, quote(\underline{happy}(\underline{y})))$.*

To accommodate all these additional symbols, we extend our signature. We write $\sqcup$ for the disjoint union and define:

**Definition 1** (augmented signature)**.** *Given a signature $S_b$= $(F_b, P_b, V_\infty, \delta_b)$ and a finite $V \subseteq V_\infty$, the augmented signature $S$ is the tuple $(F, P, V_\infty, \delta)$ with*

- $P = P_b \sqcup \{\mathbf{T}\}$
- $F = F_b \sqcup \underline{F} \sqcup \underline{P} \sqcup \underline{V} \sqcup \{\underline{\wedge}, \underline{\neg}, \underline{\forall}, quote\}$ *where*
  - $\underline{F} = \{\underline{f} \mid f \in F_b\}$
  - $\underline{P} = \{\underline{p} \mid p \in P\}$
  - $\underline{V} = \{\underline{x} \mid x \in V\}$
- $V_\infty$ *remaining the same*
- $\delta$ *specifying that $\underline{\wedge}$, $\underline{\forall}$, $\underline{\neg}$, and quote are of arities 2, 2, 1, and 1, respectively. Furthermore, $\underline{f}$ has the same arity as $f$, and $\underline{p}$ has the same arity as $p$. The arity of $\underline{x}$ is 0 for all $x$. The arity of $\mathbf{T}$ is 1.*

Without loss of generality, we assume that all the new symbols we introduce are not already in $S_b$. In what follows, we assume a fixed signature $S_b$ with an augmentation $S$. This signature implicitly defines the set $\mathcal{T}$ of all terms. We write $a \underline{\vee} b$ as syntactic sugar for $\underline{\neg}(\underline{\neg} a \underline{\wedge} \underline{\neg} b)$, and $a \underline{\rightarrow} b$ as syntactic sugar for $\underline{\neg}(a) \underline{\vee} b$.

### 4.1 Quotation sets

Now that we have a quotation-compatible signature, we want to define the quotation function $\mu$, which takes a formula and returns its quoted equivalent. For this purpose, we have to introduce a number of subsets of the set $\mathcal{T}$ of all terms, which can be roughly described as follows:

- $\underline{\mathcal{T}}$ is the subset of all quotations of well-formed terms. For example, $\underline{\mathcal{T}}$ contains the terms $\underline{+}(\underline{1}, \underline{1})$ (which is the quotation of the term $+(1, 1)$) and $\underline{f}(\underline{quote}(\underline{1}))$ (which the quotation of the term $f(\underline{1})$).
- $\underline{\mathcal{L}}$ is the subset of all quotations of well-formed (possibly not closed) formulas. For example, $\underline{\mathcal{L}}$ contains the terms $\underline{p}(\underline{x})$ (which is the quotation of the formula $p(x)$) and $\underline{p}(\underline{quote}(\underline{1}))$ (which the quotation of the formula $p(\underline{1})$).
- $\mathcal{Q}$ is the set of *all* terms made up of quotation symbols. The subset $\mathcal{Q}$ includes both $\underline{\mathcal{T}}$ and $\underline{\mathcal{L}}$. However, it also contains 'quotations' of non-well-formed terms/formulas: for example, $\mathcal{Q}$ contains the term $\underline{p}(\underline{x} \underline{\wedge} \underline{1})$ (which is the 'quotation' of the non-well-formed expression $p(x \wedge 1)$).

For each of the subsets $\underline{\mathcal{T}}$, $\underline{\mathcal{L}}$, $\mathcal{Q}$, we introduce, respectively, $\underline{\mathcal{T}_v}$, $\underline{\mathcal{L}_v}$, $\mathcal{Q}_v$, which have similar definitions, except that they also contain the construction $quote(x)$, where $x$ is a variable in $V$. This is necessary to allow variables in quotations.

We start by defining $\mathcal{Q}$ and $\mathcal{Q}_v$ formally by two Backus-Naur Forms. Our grammars apply to quotations of both terms and formulas:

**Definition 2.** *The sets $\mathcal{Q}$ and $\mathcal{Q}_v$ are defined inductively by:*

$$\begin{aligned}
\mathcal{Q} &:= \underline{x} \mid \underline{f}(t_1, \ldots, t_n) \mid \underline{p}(t_1, \ldots, t_n) \mid \underline{\wedge}(t_1, t_2) \\
&\mid \underline{\neg}(t) \mid \underline{\forall}(\underline{x}, t) \mid quote(t) \\
&\qquad \text{for } \underline{x} \in \underline{V}, \underline{f} \in \underline{F}, \underline{p} \in \underline{P}, t, t_1, ..., t_n \in \mathcal{Q} \\
\mathcal{Q}_v &:= \underline{x} \mid \underline{f}(t_1, \ldots, t_n) \mid \underline{p}(t_1, \ldots, t_n) \mid \underline{\wedge}(t_1, t_2) \\
&\mid \underline{\neg}(t_1) \mid \underline{\forall}(\underline{x}, t) \mid quote(t) \mid quote(x) \\
&\qquad \text{for } \underline{x} \in \underline{V}, \underline{f} \in \underline{F}, \underline{p} \in \underline{P}, t, t_1, ..., t_n \in \mathcal{Q}_v
\end{aligned}$$

The subsets of $\underline{\mathcal{T}}$, $\underline{\mathcal{T}_v}$, (resp. $\underline{\mathcal{L}}$ and $\underline{\mathcal{L}_v}$) are also defined by Backus-Naur Forms; but this time, we permit only quotations of well-defined terms (resp. formulas) except with a special *quote* symbol.

**Definition 3.**

$$\begin{aligned}
\underline{\mathcal{T}} &:= \underline{x} \mid \underline{f}(t_1, \ldots, t_n) \mid quote(t_q) \\
&\qquad \text{for } \underline{x} \in \underline{V}, \underline{f} \in \underline{F}, t_q \in \mathcal{Q}, t_1, \ldots, t_n \in \underline{\mathcal{T}} \\
\underline{\mathcal{T}_v} &:= \underline{x} \mid \underline{f}(t_1, \ldots, t_n) \mid quote(t_q) \mid quote(x) \\
&\qquad \text{for } \underline{x} \in \underline{V}, \underline{f} \in \underline{F}, t_q \in \mathcal{Q}, t_1, \ldots, t_n \in \underline{\mathcal{T}_v} \\
\underline{\mathcal{L}} &:= \underline{p}(t_1, \ldots, t_n) \mid \underline{\varphi_1} \underline{\wedge} \underline{\varphi_2} \mid \underline{\neg} \underline{\varphi_1} \mid \underline{\forall}(\underline{x}, \varphi_1) \\
&\qquad \text{for } \underline{x} \in \underline{V}, t_1, \ldots, t_n \in \underline{\mathcal{T}}, \underline{p} \in \underline{P}, \varphi_1, \varphi_2 \in \underline{\mathcal{L}} \\
\underline{\mathcal{L}_v} &:= \underline{p}(t_1, \ldots, t_n) \mid \underline{\varphi_1} \underline{\wedge} \underline{\varphi_2} \mid \underline{\neg} \underline{\varphi_1} \mid \underline{\forall}(\underline{x}, \varphi_1) \\
&\qquad \text{for } \underline{x} \in \underline{V}, t_1, \ldots, t_n \in \underline{\mathcal{T}_v}, \underline{p} \in \underline{P}, \varphi_1, \varphi_2 \in \underline{\mathcal{L}_v}
\end{aligned}$$

In this definition, $\underline{\mathcal{T}}$, $\underline{\mathcal{T}_v}$ (resp. $\underline{\mathcal{L}}$, $\underline{\mathcal{L}_v}$) contain only quotations of well-formed terms (resp. formulas) – except in $quote(t_q)$ where $t_q$ may be a 'quotation' of a non-well-formed expression.

### 4.2 Quoting

We now define the quotation function $\mu$. This function takes a quotable formula or term and outputs its quotation, which is a term representing the initial formula or term. We define $\mu$ jointly with the sets of terms and formulas that can be quoted.

The set of quotable terms $\mathcal{T}_q$ is the set of terms present in quotable formulas. These terms contain only variables from $V$ and are recursively formed with non-quotation symbols (the non-underlined symbols) or with the image by $\mu$ of quotable elements. By quotable elements, we mean elements of $\mathcal{T}_q$ or $\mathcal{L}_q$. The set of quotable formulas $\mathcal{L}_q$ is the set of formulas from which we can produce quotations. These are the formulas we can quantify over. They contain only variables in $V$, do not contain the predicate $\mathbf{T}$, and use only terms from $\mathcal{T}_q$.

We define $\mathcal{T}_q$, $\mathcal{L}_q$, and $\mu$ jointly by mutual induction.

**Definition 4.**

$$\begin{aligned}\mathcal{T}_q &:= x \mid t_q \mid f(t_1, \ldots, t_n)\\ &\quad \text{for } x \in V, f \in F_b, t_1, \ldots, t_n \in \mathcal{T}_q, t_q \in \mu(\mathcal{T}_q \cup \mathcal{L}_q)\\ \mathcal{L}_q &:= p(t_1, \ldots, t_n) \mid \forall x.\, \varphi \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2\\ &\quad \text{for } p \in P_b, t_1, \ldots, t_n \in \mathcal{T}_q, x \in V, \varphi, \varphi_1, \varphi_2 \in \mathcal{L}_q\end{aligned}$$

**Definition 5.** *$\mu$ is inductively defined on $\mathcal{T}_q \sqcup \mathcal{L}_q$ by:*

$$\begin{aligned}\mu(f(t_1, \ldots, t_n)) &:= \underline{f}(\mu(t_1), \ldots, \mu(t_n))\\ \mu(t_q) &:= \mathit{quote}(t_q)\\ \mu(p(t_1, \ldots, t_n)) &:= \underline{p}(\mu(t_1), \ldots, \mu(t_n))\\ \mu(x) &:= \underline{x}\\ \mu(\phi_1 \wedge \phi_2) &:= \underline{\wedge}(\mu(\phi_1), \mu(\phi_2))\\ \mu(\neg\phi) &:= \underline{\neg}(\mu(\phi))\\ \mu(\forall x.\, \phi) &:= \underline{\forall}(\underline{x}, \mu(\phi))\end{aligned}$$

***Range:*** *$f \in \underline{F}$, $x \in V$, $t_1, \ldots, t_n \in \mathcal{T}_q$, $t_q \in \mathcal{Q}$.*

For ease of reading, whenever $\mu(\varphi)$ is defined, we write it as $\underline{\varphi}$, underlining the entire argument. For instance, we write $\underline{p(x)}$ instead of $\underline{p}\ (\underline{x})$. Instead of $\underline{\mathit{ist}}(\underline{x}, \mathit{quote}(\underline{\mathit{happy}}(\underline{y})))$, we write $\underline{\mathit{ist}(x, \underline{\mathit{happy}(y)})}$.

**Example 3.** *The term $1 + x$ with $x \in V$ is quotable, i.e., $1 + x \in \mathcal{T}_q$. $1 + y$ with $y \in V_\infty \setminus V$ is not. The term $1\underline{+}x$ is not quotable because it contains the symbol $\underline{+}$; but the term $\underline{1}\underline{+}\underline{x}$ is, because it is the image by $\mu$ of $1 + x$, which is quotable.*

**Example 4.** *The formula $p(x)$ with $x \in V$ is quotable, i.e., $p(x) \in \mathcal{L}_q$. The formula $P(1\underline{+}x)$ is not quotable since $1\underline{+}x$ is not a quotable term. The formula $p(\underline{1})$ is quotable. The formula $\mathbf{T}(1)$ is not.*

Remark that whenever we quote a formula that already contains a quotation, we put said quotation in the *quote* symbol, to add a level of quotation:

**Example 5.** $\mu(P(\underline{1} = \underline{1})) = \underline{P}(\mathit{quote}(\underline{1} = \underline{1}))$.

The formulas in $\mathcal{L}_q$ do not use the predicate $\mathbf{T}$, it can never be quoted. This is an important restriction that avoids the complications of Tarski's Theorem of the undefinability of Truth.

### 4.3 Substitution on quotations

In our axiomatization, we also need a substitution function that substitutes not variables but quoted variables. Its definition is similar to the standard substitution on first-order logic formulas:

**Definition 6.** *Given $t$ in $\mathcal{T}$, given $x$ in $V$, we define $z[\underline{x} \leftarrow t]_q$ on $z \in \mathcal{Q}$ inductively:*

$$\begin{aligned}\underline{f}(t_1, \ldots, t_n)[\underline{x} \leftarrow t]_q &= \underline{f}(t_1[\underline{x} \leftarrow t]_q, \ldots, t_n[\underline{x} \leftarrow t]_q)\\ \underline{x}[\underline{x} \leftarrow t]_q &= t\\ \underline{p}(t_1, \ldots, t_n)[\underline{x} \leftarrow t]_q &= \underline{p}(t_1[\underline{x} \leftarrow t]_q, \ldots, t_n[\underline{x} \leftarrow t]_q)\\ \underline{y}[\underline{x} \leftarrow t]_q &= \underline{y}\\ \underline{\forall}(t_1, t_2)[\underline{x} \leftarrow t]_q &= \underline{\forall}(t_1[\underline{x} \leftarrow t]_q, t_2[\underline{x} \leftarrow t]_q) \text{ if } t_1 \neq \underline{x}\\ \underline{\forall}(\underline{x}, t_2)[\underline{x} \leftarrow t]_q &= \underline{\forall}(\underline{x}, t_2)\\ t_1[\underline{x} \leftarrow t]_q &= t_1 \text{ in all other cases}\end{aligned}$$

***Range:*** *$\underline{x}, \underline{y} \in \underline{V}$ ($\underline{x} \neq \underline{y}$), $t_1, \ldots, t_n \in \mathcal{T}_q$, $\underline{f} \in \underline{F}$, $\underline{p} \in \underline{P}$*

This quoted substitution function works just like the standard substitution:

**Example 6.**

$$\underline{p}(\underline{x})\underline{\wedge}\underline{\forall}(\underline{y}, \underline{q}(\underline{x}, \underline{y}))[\underline{x} \leftarrow 1\underline{+}1]_q = \underline{p}(1\underline{+}1)\underline{\wedge}\underline{\forall}(\underline{y}, \underline{q}(1\underline{+}1, \underline{y}))$$

### 4.4 Unquoting

We are now ready to define the converse of the quoting operator $\mu$:

**Definition 7.** *The unquote operator $\mu^{-1}$ is defined on $\mathcal{Q}_v$ by the following recursive definition.*

$$\begin{aligned}\mu^{-1}(\underline{f}(t_1, \ldots, t_n)) &= f(\mu^{-1}(t_1), \ldots, \mu^{-1}(t_n))\\ \mu^{-1}(\underline{p}(t_1, \ldots, t_n)) &= p(\mu^{-1}(t_1), ..., \mu^{-1}(t_n))\\ \mu^{-1}(\mathit{quote}(t)) &= t\\ \mu^{-1}(\underline{\varphi}_1\underline{\wedge}\underline{\varphi}_2) &= \mu^{-1}(\underline{\varphi}_1) \wedge \mu^{-1}(\underline{\varphi}_2)\\ \mu^{-1}(\underline{\neg}\underline{\varphi}) &= \neg\mu^{-1}(\varphi)\\ \mu^{-1}(\underline{\forall}(\underline{x}, \underline{\varphi})) &= \forall x.\, \mu^{-1}(\underline{\varphi}[\underline{x} \leftarrow \mathit{quote}(x)]_q)\\ \mu^{-1}(\underline{x}) &= x\\ \mu^{-1}(t) &= t \text{ in all other cases}\end{aligned}$$

***Range:*** *$\underline{f} \in \underline{F}$, $\underline{p} \in \underline{P}$, $x \in V$, $t, t_1, \ldots, t_n \in \mathcal{Q}_v$*

The reason we use the notation $\mu^{-1}$ for our unquote operator is that it is the inverse of $\mu$ on the image of $\mu$. $\mu^{-1}$ is not limited to the inverse of $\mu$, but it extends it.

**Proposition 1.** *$\mu$ is injective on $\mathcal{L}_q$.*

*Proof.* We can prove by induction on $(\alpha, \beta) \in (\mathcal{T}_q \sqcup \mathcal{L}_q)^2$ that $\mu(\alpha) = \mu(\beta)$ implies $\alpha = \beta$. □

**Proposition 2.** *$\forall x \in \mathcal{L}_q \cup \mathcal{T}_q, \mu^{-1}(\mu(x)) = x$*

*Proof.* This is proven by induction on $x \in \mathcal{L}_q \cup \mathcal{T}_q$. □

Intuitively, $\mu$ adds a level of underlining on quotable formulas and terms, and $\mu^{-1}$ removes it:

**Example 7.** $\mu^{-1}(\underline{\mathit{ist}(x, \underline{\mathit{happy}(y)})}) = \mathit{ist}(x, \underline{\mathit{happy}(y)})$

## 5 Defining Qiana

We can now define Qiana and its core axioms. The predicate $\mathbf{T}$ is designed to say that its argument is true in reality, as given in the following axiom schema (for all $\varphi \in \mathcal{L}_q$):

$$\mathbf{T}(\mu(\varphi)) \leftrightarrow \varphi \qquad (\mathrm{A}_{\text{truth}})$$

As famously shown by Tarski (1936), this form of predicate can lead to self-referential formulas and incoherent theories (see Enderton (2001) for a more modern description). Here, the fact that we did not allow the quotation of $\mathbf{T}$ will protect us from the pitfalls of Tarski's theorem. We show this with Proposition 3:

**Proposition 3.** *Let $S^{-\mathbf{T}}$ be the signature equal to $S$ without the symbol $\mathbf{T}$. Let $H^{-\mathbf{T}}$ be a coherent theory under the signature $S^{-\mathbf{T}}$. Let $H$ be the closure of $H^{-\mathbf{T}}$ under schema $A_{truth}$. $H$ is coherent.*

*Proof.* Let $M^{-\mathbf{T}}$ be a model of $H^{-\mathbf{T}}$. We define $M$ (the model of $H$) as equivalent to $M^{-\mathbf{T}}$ on all symbols except $\mathbf{T}$. For each $\underline{\varphi}$ in $\underline{\mathcal{L}}$ we check whether $\underline{\mu^{-1}}(\underline{\varphi})$ is true under $M^{-\mathbf{T}}$; $M \models \mathbf{T}(\underline{\varphi})$ if and only if that is the case. □

### 5.1 Truth axioms

The axiom schema $A_{truth}$ is not explicitly in a Qiana theory. It will be subsumed by the following axiom schemas, which conveniently admit direct counterparts in the finite axiomatization process of section (see Section 7.3). $x_1, ..., x_n$ are distinct variables.

$$\forall x_1, \ldots, x_n.\ \mathbf{T}(\underline{p}(t_1, \ldots, t_m)) \leftrightarrow p(\underline{\mu^{-1}}(t_1), \ldots, \underline{\mu^{-1}}(t_m)) \quad \text{(A1)}$$

$$\forall x_1, \ldots, x_n.\ \mathbf{T}(A \underline{\wedge} B) \leftrightarrow (\mathbf{T}(A) \wedge \mathbf{T}(B)) \quad \text{(A2)}$$

$$\forall x_1, \ldots, x_n.\ \mathbf{T}(\underline{\neg} A) \leftrightarrow (\neg \mathbf{T}(A)) \quad \text{(A3)}$$

$$\forall x_1, \ldots, x_n.\ \mathbf{T}(\underline{\forall}(\underline{x}, A)) \leftrightarrow (\forall x.\ \mathbf{T}(A[\underline{x} \leftarrow \mathit{quote}(x)]_q)) \quad \text{(A4)}$$

**Range:** $p \in P, t_1, \ldots, t_n \in \underline{\mathcal{T}_v}, A, B \in \mathcal{Q}_v, x \in V$

Axiom schema A1 concerns the truth of the quotation of an atomic formula. Axiom schemas A2 and A3 are about the Boolean connectives. Note that, different from $A_{truth}$, Schemas $A1$-4 apply also to non-well-formed terms.

**Example 8.** *The formula $\mathbf{T}(\underline{\neg} 2) \leftrightarrow \neg \mathbf{T}(2)$ is an instance of A3.*

Axiom schema A4 is the treatment of the universal quantification. We substitute the quotation $\underline{x}$ of a variable $x$ by $\mathit{quote}(x)$. Indeed, $\underline{x}$ is a constant symbol, and this mechanism enables to effectively simulate the quantification through a quotation. The following example illustrates this mechanism.

**Example 9.** *We show $A1$-4$\models \mathbf{T}(\underline{\forall}(\underline{x}, \underline{P}(\underline{x}))) \leftrightarrow \forall x.\ P(x)$. To prove this, let $M$ be a model of $A1$-4. We have:*

$$\begin{aligned}
&M \models \mathbf{T}(\underline{\forall}(\underline{x}, \underline{P}(\underline{x}))) \\
&\textit{iff } M \models \forall x.\ \mathbf{T}(\underline{P}(\underline{x})[\underline{x} \leftarrow \mathit{quote}(x)]_q) && \textit{as } M \models A4 \\
&\textit{iff } M \models \forall x.\ \mathbf{T}(\underline{P}(\mathit{quote}(x))) \\
&\textit{iff } M \models \forall x.\ P(\underline{\mu^{-1}}(\mathit{quote}(x))) && \textit{as } M \models A1 \\
&\textit{iff } M \models \forall x.\ P(x) && \textit{by Definition 7}
\end{aligned}$$

We are now ready to prove that any instance of $A_{truth}$ is a logical consequence of the theory A1-A4.

**Proposition 4.** $A1\text{-}4 \models A_{truth}$.

*Proof.* We prove this via induction on a larger property: Let $A \in \underline{\mathcal{L}_v}$ with no free quoted variables (ie, each $\underline{x}$ is quantified by a $\underline{\forall}$). Let $x_1, \ldots, x_n$ be the free variables of $A$. Then

$$A1\text{-}4 \models \forall x_1, \ldots, x_n.\ \mathbf{T}(A) \leftrightarrow \underline{\mu^{-1}}(A)$$

We detail the proof in the supplementary material. □

### 5.2 Reasoning axioms

Reasoning axioms endow contexts with some inference power. They say that contexts that "know" some things must also know some direct consequences of said things. We first introduce a few general schemas to this end, including associativity, commutativity, and distributivity. For example, schema A5 tells us that in any context (represented by variable $x_c$) if the conjunction of two formulas is true (represented by their quotations through variables $x_1$ and $x_2$), then the first of these formulas is also true.

$$\forall x_c, x_1, x_2.\ \mathit{ist}(x_c, x_1 \underline{\wedge} x_2) \rightarrow \mathit{ist}(x_c, x_1) \quad \text{(A5)}$$

$$\forall x_c, x_1, x_2.\ \mathit{ist}(x_c, x_1 \underline{\wedge} x_2) \leftrightarrow \mathit{ist}(x_c, x_2 \underline{\wedge} x_1) \quad \text{(A6)}$$

$$\forall x_c, x_1.\ \mathit{ist}(x_c, \underline{\neg\neg} x_1) \leftrightarrow \mathit{ist}(x_c, x_1) \quad \text{(A7)}$$

$$\forall x_c, x_1, x_2, x_3.\ \mathit{ist}(x_c, (x_1 \underline{\wedge} x_2) \underline{\wedge} x_3) \leftrightarrow \mathit{ist}(x_c, x_1 \underline{\wedge} (x_2 \underline{\wedge} x_3)) \quad \text{(A8)}$$

$$\forall x_c, x_1, x_2, x_3.\ \mathit{ist}(x_c, (x_1 \underline{\wedge} x_2) \underline{\vee} x_3) \leftrightarrow \mathit{ist}(x_c, (x_1 \underline{\vee} x_3) \underline{\wedge} (x_2 \underline{\vee} x_3)) \quad \text{(A9)}$$

Other properties of associativity, commutativity, and distributivity can be deduced from the above, also with the help of the definition of $(a \underline{\vee} b)$ as $\underline{\neg}(\underline{\neg} a \underline{\wedge} \underline{\neg} b)$. Next, we introduce the disjunctive syllogism (modus ponens), which says that if an agent knows $\phi$ and $\phi \Rightarrow \psi$, then it also knows $\psi$:

$$\forall x_c, x_1, x_2.\ \mathit{ist}(x_c, x_1 \underline{\vee} x_2) \wedge \mathit{ist}(x_c, \underline{\neg} x_1) \rightarrow \mathit{ist}(x_c, x_2) \quad \text{(A10)}$$

We also introduce an axiom schema that gives a context some ability to handle $\forall$: A quoted formula can be replaced by each of its instantiations.

$$\forall c.\ \mathit{ist}(c, \underline{\forall}(\underline{x}, \underline{\varphi})) \rightarrow \forall x.\ \mathit{ist}(c, \underline{\varphi}[\underline{x} \leftarrow \mathit{quote}(x)]_q) \quad \text{(A11)}$$

**Range:** $x \in V, \underline{\forall}(\underline{x}, \underline{\varphi}) \in \underline{\mathcal{L}}$

We illustrate the use of schema A11 with the example from the introduction:

**Example 10.**

$$\mathit{ist}(\mathit{believes}(J), \underline{\forall x.\mathit{capulet}(x) \rightarrow \mathit{nice}(x)}) \rightarrow \forall x.\ \mathit{ist}(\mathit{believes}(J), \underline{\mathit{capulet}}(\mathit{quote}(x)) \underline{\rightarrow} \underline{\mathit{nice}}(\mathit{quote}(x)))$$

### 5.3 Qiana

We can now formally define a Qiana-closure theory:

**Definition 8** (Qiana-closure theory)**.** *Let $H$ be a theory. The Qiana-closure of $H$, denoted by $H_C$, is the theory*

$$H_C = H \cup \textit{A1-A11}.$$

As an immediate property, we have semi-decidability:

**Proposition 5** (Semi-decidability)**.** *If a theory $H$ is recursively enumerable, then the problem of deciding whether its Qiana closure $H_C$ entails some formula $\varphi$ is semi-decidable.*

*Proof.* Axiom schemas A1-A11 are recursive. Any recursively enumerable theory $\Sigma$ leads to semi-decidability of the entailment problem: given $\phi$, decide whether $\Sigma \models \phi$. □

# 6 Applications

## 6.1 Reasoning in Epistemic Contexts

Let us now reconsider the example of Romeo and Juliet from the introduction. For simplicity's sake, we will not consider time modeling, which is vastly orthogonal to the features of our formalism. This choice will result in seemingly absurd simultaneity but should not hamper understanding. We start with our hypotheses from the introduction. We skip the description of suicide and consider death a direct consequence of believing one's love to be dead.

$$\forall \phi.\ \mathit{ist}(\mathit{says}(\mathit{FriarLaurence}), \phi) \rightarrow \mathbf{T}(\phi) \quad (1)$$

$$\forall x, y.\ \mathit{madlyLoves}(x, y) \wedge \mathit{ist}(\mathit{believes}(x), \underline{\mathit{dead}}(y)) \rightarrow \mathit{dead}(x) \quad (2)$$

Next, we state some obvious facts from the tragedy:

$$\mathit{madlyLoves}(\mathit{Romeo,\ Juliet}) \quad (3)$$

$$\mathit{madlyLoves}(\mathit{Juliet,\ Romeo}) \quad (4)$$

$$\mathit{ist}(\mathit{says}(\mathit{FriarLaurence}), \underline{\forall}(\underline{x}, \underline{\mathit{drinkPotion}}(\underline{x}) \underline{\rightarrow} \underline{\mathit{appearDead}}(\underline{x}))) \quad (5)$$

$$\mathit{drinkPotion}(\mathit{Juliet}) \quad (6)$$

Finally, we need some world knowledge: by definition, people can see if someone appears dead. They can also see if someone is dead.

$$\forall c, x.\ \mathit{appearDead}(x) \rightarrow \mathit{ist}(c, \underline{\mathit{appearDead}}(x)) \quad (7)$$

$$\forall x, y.\ \mathit{dead}(y) \rightarrow \mathit{ist}(x, \underline{\mathit{dead}}(y)) \quad (8)$$

The next hypothesis is perhaps best summed up as "Romeo does not know how to check someone's pulse":

$$\forall x.\mathit{ist}(\mathit{believes}(\mathit{Romeo}), \underline{\mathit{appearDead}}(x) \rightarrow \underline{\mathit{dead}}(x)) \quad (9)$$

We can now see the tragedy unfold:

$$\forall x.\mathit{drinkPotion}(x) \rightarrow \mathit{appearDead}(x) \text{ from 1, 5} \quad (10)$$

$$\mathit{appearDead}(\mathit{Juliet}) \text{ from 6 and 10} \quad (11)$$

$$\mathit{ist}(\mathit{believes}(\mathit{Romeo}), \underline{\mathit{appearDead}}(\underline{\mathit{Juliet}})) \text{ from 7, 11} \quad (12)$$

$$\mathit{ist}(\mathit{believes}(\mathit{Romeo}), \underline{\mathit{dead}}(\underline{\mathit{Juliet}})) \text{ from 12 and 9} \quad (13)$$

$$\mathit{dead}(\mathit{Romeo}) \text{ from 13, 3, and 2} \quad (14)$$

$$\mathit{ist}(\mathit{believes}(\mathit{Juliet}), \underline{\mathit{dead}}(\underline{\mathit{Romeo}})) \text{ from 14 and 8} \quad (15)$$

$$\mathit{dead}(\mathit{Juliet}) \text{ from 15, 2, and 4} \quad (16)$$

## 6.2 Paraconsistency

In first-order logic, an inconsistent theory can be used to deduce anything: if $H \vdash (\varphi \wedge \neg\varphi)$ then $H \vdash \mathit{alive}(\mathit{Elvis})$. This phenomenon is called *the principle of explosion*. While this is still true in Qiana theories, it is not true of the beliefs modeled inside contexts: a context can contain both a statement and its negation, and no axiom schema of Qiana allows deducing arbitrary statements from such beliefs (neither inside the context nor outside). This can be useful, e.g., to model contradictory beliefs. In our running example of Romeo and Juliet, let us assume for a moment that Romeo did notice that Juliet had a pulse. While this should tell him that Juliet is alive, he is too desperate to draw this conclusion:

$$H := \{\mathit{dead}(\mathit{Juliet}), \mathit{hasPulse}(\mathit{Juliet}), \forall x.\ \neg(\mathit{alive}(x) \wedge \mathit{dead}(x)), \forall x.\ \mathit{hasPulse}(x) \rightarrow \mathit{alive}(x)\}$$

In normal first-order logic, this is an inconsistent theory, and it can thus be used to deduce anything: $H \vdash \mathit{alive}(\mathit{Elvis})$. Qiana, in contrast, emulates a paraconsistent logic inside contexts. Hence, the principle of explosion does not apply inside contexts:

$$H' := \{\mathit{ist}(\mathit{believes}(\mathit{Romeo}), \underline{\mathit{dead}}(\underline{\mathit{Juliet}})),\ \mathit{ist}(\mathit{believes}(\mathit{Romeo}), \underline{\mathit{hasPulse}}(\underline{\mathit{Juliet}})),\ \mathit{ist}(\mathit{believes}(\mathit{Romeo}), \underline{\forall}(\underline{x}, \underline{\neg}(\underline{\mathit{alive}}(\underline{x}) \underline{\wedge} \underline{\mathit{dead}}(\underline{x})))),\ \mathit{ist}(\mathit{believes}(\mathit{Romeo}), \underline{\forall}(\underline{x}, \underline{\mathit{hasPulse}}(\underline{x}) \underline{\rightarrow} \underline{\mathit{alive}}(\underline{x})))\}$$

These contradictory thoughts now entail:

$$H' \vdash \mathit{ist}(\mathit{believes(Romeo)}, \underline{\mathit{dead}}(\underline{\mathit{Juliet}}))$$

$$H' \vdash \mathit{ist}(\mathit{believes(Romeo)}, \underline{\neg\mathit{dead}}(\underline{\mathit{Juliet}}))$$

However, they do not imply that Romeo believes anything:

$$H' \nvdash \mathit{ist}(\mathit{believes(Romeo)}, \underline{\mathit{alive}}(\underline{\mathit{Elvis}}))$$

If we want to keep the principle of explosion inside contexts, we can add the following axiom schema to our theories (for all $\varphi, \psi \in \mathcal{L}_q$):

$$\forall c.\ ist(c, \underline{\varphi}) \rightarrow ist(c, \underline{\varphi \vee \psi}) \quad (17)$$

Together with modus ponens, it allows to deduce anything from a contradiction. From $ist(c, \varphi)$ we deduce $ist(c, \underline{\varphi \vee \psi})$. From $ist(c, \underline{\neg}\varphi)$ and $ist(c, \underline{\varphi \vee \psi})$ we deduce $ist(c, \psi)$.

## 6.3 Mixing different types of contexts

Until now, we have shown how to use contexts to model beliefs, and we have considered the play itself as the truth. However, contexts can also encapsulate a story. To illustrate this, let us consider two versions of the story of Romeo and Juliet: the original and a fanfiction variant. In the fanfiction variant, Romeo decides to check Juliet's pulse and notices that she is alive. He waits for her to come to her senses, and then they leave together and live happily ever after.

We start by declaring that the fanfiction and the original are both stories:

$$\mathit{story}(\mathit{fanfiction}) \wedge \mathit{story}(\mathit{original})$$

In the fanfiction, Romeo checks Juliet's pulse; in the original, he does not:

$$ist(\mathit{fanfiction}, \underline{\mathit{checkPulse}}(\underline{R}, \underline{J}))$$

$$ist(\mathit{original}, \underline{\neg\mathit{checkPulse}}(\underline{R}, \underline{J}))$$

In all stories, if Romeo checks Juliet's pulse, he knows she is alive. In all stories, for all persons, if Romeo does not feel their pulse and they appear dead, he thinks they are dead.

$$\forall s.\ \mathit{story}(s) \rightarrow ist(s, \underline{\mathit{checkPulse}(R, J) \rightarrow ist(\mathit{believes}(R), \underline{\mathit{alive}(J)})})$$

$$\forall s.\ \mathit{story}(s) \rightarrow ist(s, \underline{\forall x.\ \mathit{appearDead}(x) \wedge \neg\mathit{checkPulse(R,x)} \rightarrow ist(\mathit{believes}(R), \underline{\neg\mathit{alive}(x)})})$$

In all stories, Juliet appears dead. In all stories, if Romeo knows Juliet is alive, he does not kill himself, but he does if he thinks she is dead. In all stories, the protagonists will be either both dead or both alive:

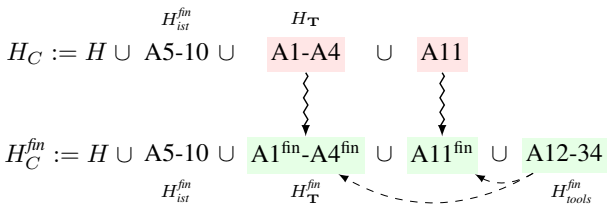


Figure 1: Overview of the process of finite axiomatization. Zizag arrows indicate the finite sets are counterparts to the top infinite ones. Dotted arrows indicate the the elements of $H_{tools}^{fin}$ are used to define said counterparts.

$$\forall s.\ \textit{story}(s) \rightarrow ist(s, \underline{appearDead(J)})$$
$$\forall s.\ \textit{story}(s) \rightarrow ist(s, \underline{ist(\textit{believes}(R), \underline{alive(J)) \rightarrow alive(R)}})$$
$$\forall s.\ \textit{story}(s) \rightarrow ist(s, \underline{ist(\textit{believes}(R), \underline{\neg alive(J)) \rightarrow \neg alive(R)}})$$
$$\forall s.\ \textit{story}(s) \rightarrow ist(s, \underline{alive(R) \leftrightarrow alive(J)})$$

These formulas – together with the axioms of Qiana – are enough to deduce that there is the usual ending in the original version of the story and a vastly more fortunate one in the fanfiction story:

$$ist(\textit{original}, \underline{\neg alive(J) \wedge \neg \textit{alive}(R)})$$
$$ist(\textit{fanfiction}, \underline{alive(J) \wedge \textit{alive}(R)})$$

# 7 Finite axiomatization

## 7.1 Structure and overview

We will now show how the Qiana closure of any finite theory can be finitely axiomatized. Let $H$ be a given finite theory on a quotation-compatible signature $S$. The Qiana-closure of $H$ is $H_C = H \cup$ A1-A4 $\cup$ A5-A10 $\cup$ A11 (see Definition 8). Both A1-A4 and A11 are infinite. Hence, $H_C$ is also infinite. In this section, we will present a process to define another theory $H_C^{fin}$ that is both finite and equisatisfiable with $H_C$. This will allow us to test the satisfiability of $H_C$ by feeding finitely many formulas (the elements of $H_C^{fin}$) to a theorem prover. Subsections 7.2 and 7.3 will introduce secondary sets of use in this finite axiomatization. Subsection 7.4 concludes the presentation of this process and gives the relevant results.

We write $H_{ist}^{fin} =$ A5-A10. Fortunately, $H_{ist}^{fin}$ is finite. However, we need to replace the infinite axiom schemas A1-A4 and A11. To do so, we extend the signature $S$ to another signature $S'$, which contains new symbols that we describe with additional (finite) schemas. Together, these new schemas form the set $H_{tools}^{fin}$. We present these symbols and the set $H_{tools}^{fin}$ in Subsection 7.2. Resting on these symbols and their definition schemas, we introduce finite counterparts to our infinite schemas in Subsection 7.3. These are the sets $H_{\mathbf{T}}^{fin}$ and $H_{\underline{\forall}}$. Together, these sets allow to define a new set $H_C^{fin} = H \cup$ A5-A10 $\cup H_{tools}^{fin} \cup H_{\mathbf{T}}^{fin} \cup H_{\underline{\forall}}$, which is finite and equisatisfiable with $H_C$ (see Figure 1). We present some interesting properties of the process in Subsection 7.4, the most important being the correctness of the process.

## 7.2 Additional symbols

We consider a fixed and finite theory $H$ on $S$. Without loss of generality, we introduce fresh function symbols *Sub, E, Wft, isTerm,* and a 2-ary predicate symbol $=$. By adding these symbols to $S$ we obtain a larger signature $S'$. We now give the axiom schemas that describe the behavior of these symbols. The symbol = is the standard equality predicate defined by the following axioms, in which $x, y, x_1, \ldots, x_n, y_1, \ldots, y_n$ are distinct variables:

$$\forall x.\ x = x \tag{A12}$$
$$\forall x, y.\ x = y \rightarrow y = x \tag{A13}$$
$$\forall x, y, z.\ x = y \wedge y = z \rightarrow x = z \tag{A14}$$
$$\forall x_1, .., x_n, y_1, .., y_n.\ x_1 = y_1 \wedge .. \wedge x_n = y_n \rightarrow f(x_1, .., x_n) = f(y_1, .., y_n) \tag{A15}$$
$$\forall x_1, .., x_n, y_1, .., y_n.\ x_1 = y_1 \wedge .. \wedge x_n = y_n \rightarrow p(x_1, .., x_n) \leftrightarrow p(y_1, .., y_n) \tag{A16}$$

**Range:** $f \in F, p \in P$

The symbol *isTerm* checks whether its argument can be expressed as a term. More precisely, *isTerm*$(t)$ is true if $t$ is a closed term or has all its open variables behind a *quote* statement:

$$\forall x.\ \textit{isTerm}(\textit{quote}(x)) \tag{A17}$$
$$\forall t_1, \ldots, t_n.\ (\textit{isTerm}(t_1) \wedge \cdots \wedge \textit{isTerm}(t_n)) \rightarrow \textit{isTerm}(f(t_1, \ldots, t_n)) \tag{A18}$$

**Range:** $f \in F$

The symbol *Wft* stands for "well-formed term". Intuitively, *Wft*$(t)$ is true iff $t$ represents a quotation of a well-formed term (i.e., $t \in \mathcal{Q}_v$):

$$\forall y.\ \textit{Wft}(\textit{quote}(y)) \tag{A19}$$
$$\textit{Wft}(\underline{x}) \tag{A20}$$
$$\forall t_1, \ldots, t_n.\ (\textit{Wft}(t_1) \wedge \cdots \wedge \textit{Wft}(t_n)) \rightarrow \textit{Wft}(\underline{f}(t_1, \ldots, t_n)) \tag{A21}$$

**Range:** $\underline{x} \in \underline{V}, \underline{f} \in \underline{F}$

The symbol E is the counterpart to $\mu^1$ on quoted terms. More precisely, E$(t)$ is defined to inductively evaluate to the value that $t$ is a quotation of, when applicable:

$$\forall t.\ \textit{isTerm}(t) \rightarrow E(quote(t)) = t \tag{A22}$$
$$\forall t_1, .., t_n.\ \textit{isTerm}(t_1) \wedge .. \wedge \textit{isTerm}(t_n) \rightarrow E(\underline{f}(t_1, .., t_n)) = f(E(t_1), .., E(t_n)) \tag{A23}$$
$$\forall t_1, .., t_n.(\textit{isTerm}(t_1) \wedge .. \wedge \textit{isTerm}(t_n)) \rightarrow E(\underline{p}(t_1, .., t_n)) = \underline{p}(t_1, .., t_n) \tag{A24}$$
$$\forall t_1, t_2.\ E(\underline{\wedge}(t_1, t_2)) = \underline{\wedge}(t_1, t_2) \tag{A25}$$
$$\forall t_1, t_2.\ E(\underline{\forall}(t_1, t_2)) = \underline{\forall}(t_1, t_2) \tag{A26}$$
$$\forall t.\ E(\underline{\neg}(t)) = \underline{\neg}(t) \tag{A27}$$
$$E(\underline{x}) = \underline{x} \tag{A28}$$

**Range:** $\underline{x} \in \underline{V}, f \in F, p \in P$

When no step of this induction can be carried out, we have $E(t) = t$ (Axiom Schema A24-A28).

The symbol *Sub* is an in-logic counterpart of the substitution operator in Definition 6. The term $Sub(t_1, t_2, t_3)$ represents $t_1[t_2 \leftarrow t_3]_q$:

$$\forall t.\ isTerm(t) \rightarrow Sub(\underline{x}, \underline{x}, t) = t \quad \text{(A29)}$$

$$\forall t.\ isTerm(t) \rightarrow Sub(\underline{x}, \underline{y}, t) = \underline{y} \quad \text{(A30)}$$

$$\forall t, t_1, \ldots, t_n.\ (isTerm(t_1) \wedge \cdots \wedge isTerm(t_n)) \rightarrow Sub(\underline{x}, \underline{f}(t_1, \ldots, t_n), t) = \underline{f}(Sub(\underline{x}, t_1, t), \ldots, Sub(\underline{x}, t_n, t)) \quad \text{(A31)}$$

$$\forall t_1, t_2.\ (isTerm(t_1) \wedge isTerm(t_2)) \rightarrow Sub(\underline{\forall}(\underline{x}, t_1), \underline{x}, t_2) = \underline{\forall}(\underline{x}, t_2) \quad \text{(A32)}$$

$$\forall t_1, t_2.\ (isTerm(t_1) \wedge isTerm(t_2)) \rightarrow Sub(\underline{\forall}(\underline{y}, t_1), \underline{x}, t_2) = \underline{\forall}(\underline{y}, Sub(t_1, \underline{x}, t_2)) \quad \text{(A33)}$$

$$\forall t_1, t_2.\ (isTerm(t_1) \wedge isTerm(t_2)) \rightarrow Sub(quote(t_1), \underline{x}, t_2) = quote(t_1) \quad \text{(A34)}$$

**Range:** $\underline{x}, \underline{y} \in \underline{V}$ $(\underline{x} \neq \underline{y})$, $f \in F, p \in P$

We group all these helper axiom schemas together as a theory $H^{fin}_{tools}$:

**Definition 9.** $H^{fin}_{tools} := A12\text{-}A34$.

### 7.3 Finite counterparts to infinite schemas

Let us write $H_{\mathbf{T}} = A1\text{-}4$. The set $H_{\mathbf{T}}$ defines the behavior of $\mathbf{T}$ on well-formed formula quotations. Now that we have introduced new symbols to act as in-logic counterparts to the most important meta-operators of these schemas, we can introduce new finite schemas that mimic the behavior of $A1\text{-}4$ with finitely many formulas. This is done with $H^{fin}_{\mathbf{T}}$:

**Definition 10.** $H^{fin}_{\mathbf{T}} = A1^{fin}\text{-}A4^{fin}$.

$$\forall t_1, .., t_n.\ (Wft(t_1) \wedge .. \wedge Wft(t_n)) \rightarrow \mathbf{T}(\underline{p}(t_1, .., t_n)) \leftrightarrow p(E(t_1), .., E(t_n)) \quad \text{(A1}^{\text{fin}}\text{)}$$

$$\forall t_1, t_2.\ (isTerm(t_1) \wedge isTerm(t_2)) \rightarrow \mathbf{T}(t_1 \underline{\wedge} t_2) \leftrightarrow (\mathbf{T}(t_1) \wedge \mathbf{T}(t_2)) \quad \text{(A2}^{\text{fin}}\text{)}$$

$$\forall t_1.\ isTerm(t_1) \rightarrow \mathbf{T}(\underline{\neg} t_1) \leftrightarrow (\neg \mathbf{T}(t_1)) \quad \text{(A3}^{\text{fin}}\text{)}$$

$$\forall t_1.\ isTerm(t_1) \rightarrow \mathbf{T}(\underline{\forall}(\underline{x}, t_1)) \leftrightarrow (\forall x.\ \mathbf{T}(Sub(t_1, \underline{x}, quote(x)))) \quad \text{(A4}^{\text{fin}}\text{)}$$

***Range:*** $p \in P \setminus \{\mathbf{T}\}, x \in V$

Likewise, we define schema $\text{A11}^{\text{fin}}$ as a finite counterpart to schema A11.

$$\forall t_1, t_2.\ isTerm(t_1) \rightarrow ist(t_2, \underline{\forall}(\underline{x}, t_1)) \rightarrow \forall x.\ ist(t_2, Sub(t_1, \underline{x}, quote(x))) \quad \text{(A11}^{\text{fin}}\text{)}$$

**Range:** $x \in V$

### 7.4 Correctness

We can now formally definite the finite axiomatization of Qiana on $H$ as the set $H^{fin}_C$:

**Definition 11.**

$$H^{fin}_C := H \cup H^{fin}_{ist} \cup H^{fin}_{tools} \cup H^{fin}_{\mathbf{T}} \cup A11^{fin}$$

Recall that the Qiana-closure of $H$ is $H_C = H \cup H^{fin}_{ist} \cup A1\text{-}4 \cup A11$. The following theorem and corollary say that it is equivalent to reason with the theories $H_C$ or $H^{fin}_C$:

**Theorem 1.** *$H_C$ is coherent if and only if $H^{fin}_C$ is coherent.*

*Proof.* Recall that "$H_C$ is coherent" is equivalent to $H_C \not\models \bot$. Hence the theorem becomes $H_C \not\models \bot$ iff $H^{fin}_C \not\models \bot$. In the supplementary material, we prove both directions of this equivalence in Propositions **??** and **??**. □

**Corollary 1.** *$H_C \models \varphi$ if and only if $H^{fin}_C \models \varphi$, for all $\varphi$*

*Proof.* Apply Theorem 1 to the theory $H_C \cup \{\neg\varphi\}$. □

The following proposition says that the number of formulas created by the finite axiomatization process is quadratic in the total number of symbols, excluding the variables that are not quotable.

**Proposition 6.** *The cardinal of $H^{fin}_{ist} \cup H^{fin}_{tools} \cup H^{fin}_{\mathbf{T}}$ is in $\mathcal{O}(|S|^2)$, where $|S|$ is the total number of symbols in $S$, excluding $V_\infty \setminus V$.*

To make this finite axiomatization process possible, we had to introduce a finite number of quoted variables. Indeed, each quoted variable needs to appear at least once within the finite axiomatization. Since the process uses finitely many formulas of finite length, it cannot handle an infinite number of quoted variables. Nevertheless, any reasoning that can be carried out with an infinite number of variables can also be done with a finite number of variables:

**Proposition 7.** *Let $H$ be a theory, and let $H^n_C$ be the Qiana closure of $H$ where $V$ has size $n \in \mathbb{N}$, and let $H^\infty_C$ be the Qiana closure of $H$ obtained by allowing the set $V$ to be infinite. Let $\varphi$ be any well-formed closed formula. Then if $H^\infty_C \models \varphi$ then there is some $n \in \mathbb{N}$ such that $H^n_C \models \varphi$.*

*Proof.* Any proof derivation of $\varphi$ from $H^\infty_C$ uses finitely many formulas, which are all in $H^n_C$ for some $n$. □

Thus, the finiteness of the set $V$ of quotable variables is not a limitation on the reasoning power. When checking some entailment, we can iteratively increase the size of $V$ to check if the entailment appears. Considering that our theory is semi-decidable rather than decidable to begin with, we do not lose any deductive power.

## 8 Using Qiana in Theorem Provers

Our finite axiomatization allows us to transform the Qiana closure of any finite theory into an equisatisfiable finite first-order logic theory, which can then be fed into a theorem prover. We have implemented a translator (in Python) that accepts a set of Qiana formulas, derives their signature, and outputs a finite set of Qiana axioms in the TPTP syntax (Sutcliffe, 2009).

This allows us to run the Romeo and Juliet example from Section 6 in the Vampire theorem prover Riazanov and Voronkov (2001). The reasoning takes 0.05 seconds on an 8th-generation Intel CPU laptop. Vampire duly proves that both Romeo and Juliet die. The code and the example are available at https://github.com/dig-team/Qiana.

## 9 Discussion

**Distinguishing contexts.** Formulas such as Formula (7) in Section 6 imply that every object in the domain of discourse is a context. This is not a problem for our formalism, but it is undeniably counter-intuitive. This issue could be addressed by resorting to many-sorted first-order logic (Enderton, 2001), or to a special dedicated predicate for contexts.

In this paper, we chose to stick to standard (single-sorted) first-order logic and we did not introduce a predicate for contexts. This decision was made to maintain the simplicity of our presentation and definitions and to minimize restrictions on compatible theorem provers.

**The definition of $V$.** We defined $V$ as a finite subset of $V_\infty$, which is in bijection with $\underline{V}$. In formulas such as Axiom A4, we connect $x$ (an element of $V$) with $\underline{x}$ (the corresponding element of $\underline{V}$). This is a convenient way to present our axioms without a lengthy discussion on fresh variables and the like. However, in first-order logic, bound variables can be freely renamed with fresh variables. In some schemes (like Formula A4), we limited the range of some $x$ to $V$ rather than $V_\infty$. But all the bound variables can be replaced with elements of $V_\infty$ without issue. The only aspect of $V$ that matters is that is has the same size as $\underline{V}$. All that the notion of "quotable variable" amounts to is a limitation on the number of distinct variables in a quotable formula; along with giving us a convenient way to state our axioms.

## 10 Conclusion

We have introduced Qiana, a formalism based on first-order logic that allows reasoning on contexts, quantifying over contexts, and quantifying over formulas. Thanks to our finite axiomatization process, Qiana theories can be used with any TPTP compatible theorem prover. We have shown that Qiana can be used to model beliefs, stories, and paraconsistency.

We expect Qiana to be usable for and adaptable to various types of contextual reasoning cases, including reasoning on hypothetical scenarios, fake news, and different points of view. Future work can investigate how Qiana can be used in automated reasoning that integrates data from different sources that are not deemed trustworthy by default.

## References


Aljalbout, S.; Buchs, D.; and Falquet, G. 2019. Introducing contextual reasoning to the semantic web with OWL ˆc. In *ICCS*.

Brewka, G.; Eiter, T.; Fink, M.; and Weinzierl, A. 2011. Managed multi-context systems. In *IJCAI*.

Buvac, S., and Mason, I. A. 1993. Propositional logic of context. In *AAAI*.

Buvac, S.; Buvac, V.; and Mason, I. A. 1994. The semantics of propositional contexts. In *ISMIS*.

Buvac, S. 1996. Quantificational logic of context. In *AAAI*.

Carroll, J. J.; Bizer, C.; Hayes, P.; and Stickler, P. 2005. Named graphs. *Journal of Web Semantics* 3(4):247–267.

Enderton, H. B. 2001. *A Mathematical Introduction to Logic*. HARCOURT/Academic Press.

Genesereth, M. R. 1991. Knowledge interchange format. In *KR*.

Ghidini, C., and Giunchiglia, F. 2001. Local models semantics, or contextual reasoning=locality+compatibility. *Artif. Intell.* 127(2):221–259.

Giunchiglia, F., and Bouquet, P. 1997. Introduction to contextual reasoning: an artificial intelligence perspective. In *European Summer School in Cognitive Science*.

Giunchiglia, F. 1993. Contextual reasoning. *Epistemologia, special issue on I Linguaggi e le Macchine*.

Gödel, K. 1931. Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I. *Monatshefte für mathematik und physik* 38:173–198.

Guha, R. V.; McCool, R.; and Fikes, R. 2004. Contexts for the semantic web. In *ISWC*.

Halpern, J. Y., and Moses, Y. 1992. A guide to completeness and complexity for modal logics of knowledge and belief. *Artif. Intell.* 54(2).

Hartig, O.; Champin, P.-A.; Kellogg, G.; and Seaborne, A. 2023. Rdf-star and sparql-star.

Hintikka, K. J. J. 1962. Knowledge and belief: An introduction to the logic of the two notions. *The Philosophical Review*.

ISO. 2018. Common logic (cl)–a framework for a family of logic-based languages. *International Organization for Standardization* 24707.

McCarthy, J. 1987. Generality in artificial intelligence. *ACM* 30(12).

McCarthy, J. 1993. Notes on formalizing context. In *IJCAI*.

Menzel, C. 2013. Completeness theorem for logic with a single type.

Moore, R. C. 1980. Reasoning about knowledge and action. Technical Report ADA126244, Massachusetts institute of Technology.

Moore, R. C. 1981. Reasoning about knowledge and action. In Webber, B. L., and Nilsson, N. J., eds., *Readings in Artificial Intelligence*. Morgan Kaufmann. 473–477.

Mossakowski, T.; Codescu, M.; Kutz, O.; Lange, C.; and Grüninger, M. 2014. Proof support for common logic. In *ARQNL@IJCAR*.

Perrussel, L. 2002. First-order contextual reasoning. In *Brazilian Symposium on Artificial Intelligence*.

Ranganathan, A., and Campbell, R. H. 2003. An infrastructure for context-awareness based on first order logic. *Pers. Ubiquitous Comput.* 7(6):353–364.

Riazanov, A., and Voronkov, A. 2001. Vampire 1.1 (system description). In *IJCAR*.

Suchanek, F. M. 2005. Ontological reasoning for natural language understanding. Master's thesis, Saarland University, Germany.

Sutcliffe, G. 2009. The TPTP problem library and associated infrastructure. *J. Autom. Reason.* 43(4).

Taprogge, M., and Steen, A. 2023. Flexible automation of quantified multi-modal logics with interactions. In *KI*.

Tarski, A. 1936. The concept of truth in formalized languages. In *Logic, Semantics, Metamathematics*. Oxford University Press.

Väänänen, J. 2021. Second-order and Higher-order Logic. In *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University.