Journal of Artificial Intelligence Research 1 (1993) 1-15 Submitted 6/91; published 9/91

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

Simon Coumes SIMON.COUMES@TELECOM-PARIS.FR
Telecom Paris, Institut Polytechnique de Paris,
91120 Palaiseau, France

Pierre-Henri Paris PIERRE-HENRI.PARIS@UNIVERSITE-PARIS-SACLAY.FR
Université Paris-Saclay, 3 rue Joliot Curie,
91190 Gif-sur-Yvette, France

François Schwarzentruber FRANCOIS.SCHWARZENTRUBER@ENS-LYON.FR
ENS Lyon, 15 parvis René Descartes,
69342 Lyon, France

Fabian Suchanek FABIAN.SUCHANEK@TELECOM-PARIS.FR
Telecom Paris, Institut Polytechnique de Paris,
91120 Palaiseau, France

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. We show how Qiana can be used to represent temporality, event calculus, and modal logic. We also discuss different design alternatives of Qiana.

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 φ is true in the context c. Contexts can represent different things: Something can be true only in the context of a newspaper article, 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 link truth in reality and in contexts ("$ist(c, \varphi) \rightarrow \varphi$")

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).

This article is an extended version of our previous conference paper (Coumes, Paris, Schwarzentruber, & Suchanek, 2024), which proposes representing formulas within contexts as regular terms of the logic 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 ensure 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 $\mathbb{Q}$ to nest quotations within quotations. This allows for a larger array of manipulations around contexts, which are notably useful for our finite axiomatization process.

In this paper, we improve the presentation of Qiana found in (Coumes et al., 2024), we extend Qiana to reason about temporality and events, we present an alternative version of Qiana that is based on typed logic, and we discuss how usual modal logics can be represented within Qiana.

Qiana can model agents' beliefs (as in our Romeo and Juliet example). Still, 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). The first part of this paper largely follows our original paper on Qiana (Coumes et al., 2024): Section 2 discusses the related work; Section 3 introduced notations; Section 4 explains how Qiana quotes formulas; Section 5 defines Qiana; Section 6 discussed simple applications of Qiana; and Section 7 describes the finite axiomatization process of Qiana for use with automated theorem provers.

In the second part of this paper, we provide more discussion that goes beyond the original paper (Coumes et al., 2024). In Section 8, we extend Qiana to reason about temporality and events. In Section 9, we present an alternative version of Qiana that is based on typed logic. In Section 10, we discuss how usual modal logics can be represented in Qiana. Finally, Section 12 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

	Truth Representation	Formula Quantif	Context Quantif	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 (1997)	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, Bizer, Hayes, and Stickler (2005)	yes	no	no	yes
Brewka, Eiter, Fink, and Weinzierl (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, Champin, Kellogg, and Seaborne (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

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 $ist(c, p)$ 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 ϕ 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 $\mathbb{Q}$, 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 ϕ, and $\mathsf{B}_i\phi$ means that i believes ϕ. 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 (1997) 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,

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.

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 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, Codescu, Kutz, Lange, & Grüninger, 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 et al. (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 φ until a quantification over x is reached.

For our purposes, a *signature* S is a tuple (F, P, V_∞, δ), where F is a set of function symbols, P is a set of predicate symbols, V_∞ 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 $[\![p]\!] \in D^{\delta(p)} \rightarrow \{0, 1\}$. An assignment is a partial function $\sigma : V_\infty \rightarrow D$. Given an assignment for the free variables σ, 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 $\neg\neg\varphi \to \varphi$, where φ 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, σ an assignment of values in the domain of M to free variables, and H a theory. We write $M, \sigma \models \varphi$ to say that the formula φ is true in model M, where all the free variables of φ are defined in σ. We omit σ if there are no free variables. We write $H \models \varphi$ to say that φ 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

The main idea of Qiana is to represent formulas that are true only in a specific context by *quoted formulas*. Technically, a quoted formula is a term that represents a formula. FOL does not allow to manipulate formulas as objects, which is why we need to introduce our quoted formulas, which are terms that serve as counterparts of formulas we can manipulate. 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 $\underline{\wedge}(\underline{p}(\underline{x}), \underline{=}(\underline{+}(\underline{1}, \underline{x}), \underline{2}))$ where $\underline{p}, \underline{\wedge}, \underline{1}, \underline{+}, \underline{x}, \underline{=}, \underline{2}$ are quoted counterparts to the original symbols p, $\wedge$, 1, +, x, =, and 2. For convenience and readability, we can write it $\underline{p}\ (\underline{x}) \ \underline{\wedge}\ (\underline{1} \ \underline{+} \ \underline{x} \ \underline{=} \ \underline{2})$*

We need to quote formulas that already contain quotations. To this end, we introduce a special function symbol $\mathbb{Q}$ (read "quote"), which acts as an escape character and provides a way to nest quotations. The symbol $\mathbb{Q}$ is called the *escape operator* or the *quote operator*.
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_∞, δ) with*

- $P = P_b \sqcup \{\mathbb{T}\}$
- $F = F_b \sqcup \underline{F} \sqcup \underline{P} \sqcup \underline{V} \sqcup \{\underline{\wedge}, \underline{\neg}, \underline{\forall}, \mathbb{Q}\}$ *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_∞ *remaining the same*
- δ *specifying that $\underline{\wedge}$, $\underline{\forall}$, $\underline{\neg}$, and $\mathbb{Q}$ 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 $\mathbb{T}$ 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 $a \underline{\to} b$ 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 μ, 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 $\underline{f}(\mathbb{Q}(\underline{1}))$ (which is 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 $\underline{p}(\mathbb{Q}(\underline{1}))$ (which is the quotation of the formula $p(\underline{1})$).
- $\mathcal{Q}$ is the set of *all* terms made up of quotation symbols. The set $\mathcal{Q}$ includes both $\underline{\mathcal{T}}$ and $\underline{\mathcal{L}}$. However, it also contains quotations of non-well-formed terms and formulas. For example, $\mathcal{Q}$ contains the term $\underline{p}(\underline{x} \mathbin{\underline{\wedge}} \underline{1})$ (which is the "quotation" of the non-well-formed expression $p(x \wedge 1)$).

We provide Backus-Naur definitions of $\underline{\mathcal{T}}$, $\underline{\mathcal{L}}$, and $\mathcal{Q}$ below. For each of the subsets $\underline{\mathcal{T}}$, $\underline{\mathcal{L}}$, and $\mathcal{Q}$ we introduce (respectively) $\underline{\mathcal{T}_v}$, $\underline{\mathcal{L}_v}$, and $\mathcal{Q}_v$, which have similar definitions except that they also contain the construction $\mathbb{Q}(x)$ with x 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:*

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

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 $\mathbb{Q}$ symbol.

Definition 3. *The sets $\underline{\mathcal{T}}, \underline{\mathcal{T}_v}, \underline{\mathcal{L}}, \underline{\mathcal{L}_v}$ are defined inductively by*

$$\begin{aligned}
\underline{\mathcal{T}} &:= \underline{x} \mid \underline{f}(t_1,\ldots,t_n) \mid \mathbb{Q}(t_q) \\
&\quad \text{for } \underline{x} \in \underline{V}, \underline{f} \in \underline{F}, t_q \in \mathcal{Q}, t_1,\ldots,t_n \in \underline{\mathcal{T}} \\
\underline{\mathcal{T}_v} &:= \underline{x} \mid \underline{f}(t_1,\ldots,t_n) \mid \mathbb{Q}(t_q) \mid \mathbb{Q}(x) \\
&\quad \text{for } \underline{x} \in \underline{V}, \underline{f} \in \underline{F}, t_q \in \mathcal{Q}, t_1,\ldots,t_n \in \underline{\mathcal{T}_v} \\
\underline{\mathcal{L}} &:= \underline{p}(t_1,\ldots,t_n) \mid \underline{\varphi}_1 \underline{\wedge} \underline{\varphi}_2 \mid \underline{\neg}\underline{\varphi}_1 \mid \underline{\forall}(\underline{x},\varphi_1) \\
&\quad \text{for } \underline{x} \in \underline{V}, t_1,\ldots,t_n \in \underline{\mathcal{T}}, \underline{p} \in \underline{P}, \varphi_1, \varphi_2 \in \underline{\mathcal{L}} \\
\underline{\mathcal{L}_v} &:= \underline{p}(t_1,\ldots,t_n) \mid \underline{\varphi}_1 \underline{\wedge} \underline{\varphi}_2 \mid \underline{\neg}\underline{\varphi}_1 \mid \underline{\forall}(\underline{x},\varphi_1) \\
&\quad \text{for } \underline{x} \in \underline{V}, t_1,\ldots,t_n \in \underline{\mathcal{T}_v}, \underline{p} \in \underline{P}, \varphi_1, \varphi_2 \in \underline{\mathcal{L}_v}
\end{aligned}$$

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 $\mathbb{Q}(t_q)$ where t_q may be a "quotation" of a non-well-formed expression.

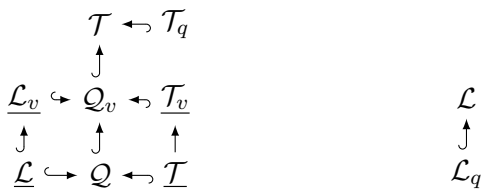

Figure 1: Inclusion relationship ($\hookrightarrow$) between subsets of the set of terms $\mathcal{T}$ and subset $\mathcal{L}_q$ of $\mathcal{L}$.

4.2 Quoting

We now define the quotation function μ. This function takes a quotable formula or term and outputs its quotation. We define μ jointly with the sets of terms and formulas it can be applied to.

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 μ 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 for which we can produce quotations. They contain only variables in V, do not contain the predicate $\mathbb{T}$, and use only terms from $\mathcal{T}_q$. These are the formulas we can handle as objects of the logic, the formulas we can say are true or false in a context.

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

Definition 4. *The sets $\mathcal{T}_q, \mathcal{L}_q$ are defined inductively by*

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

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

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

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 $\underline{ist}(\underline{x}, \mathbb{Q}(\underline{happy}(\underline{y})))$, we write $\underline{ist(x, happy(y))}$.

Example 2. *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 of the symbol $\underline{+}$; but the term $\underline{1\underline{+}x}$ is quotable, because it is the image by μ of $1 + x$, which is quotable.*

Example 3. *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.*

Note that whenever we quote a formula that already contains a quotation, we put said quotation in the $\mathbb{Q}$ symbol, to add a level of quotation:

Example 4. $\underline{P(\underline{1 = 1})} = \mu(P(\underline{1 = 1})) = \underline{P}(\mathbb{Q}(\underline{1 = 1}))$.

The formulas in $\mathcal{L}_q$ do not use the predicate $\mathbb{T}$, it can never be quoted. This is an important restriction that avoids the complications of Tarski's Theorem of the undefinability of Truth (see Proposition 3 in Section 5).

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 as:*

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

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 5. *Applying the quoted substitution function:*

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

4.4 Unquoting

We are now ready to define the converse of the quoting operator μ:

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

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

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 $\mu^{\dot{-}1}$ for our unquote operator is that it is the inverse of μ on the image of μ. $\mu^{\dot{-}1}$ is not limited to the inverse of μ, but it extends it.

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

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

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

Example 6. $\mu^{\dot{-}1}(\underline{ist(x,\underline{happy(y)})})=ist(x,\underline{happy(y)})$

5. Defining Qiana

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

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

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 $\mathbb{T}$ will protect us from the pitfalls of Tarski's theorem. We show this with Proposition 3:

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

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

5.1 Truth axioms

The axiom schema $\mathrm{A}_{\text{truth}}$ is not explicitly in a Qiana theory. It will be subsumed by axiom schemas A1 to A4, which conveniently admit direct counterparts in the finite axiomatization process of Section 7.3. Here, $x_1, ..., x_n$ are distinct variables.

$$\forall x_1, \ldots, x_n.\ \mathbb{T}(\underline{p}(t_1, \ldots, t_m)) \leftrightarrow p(\dot{\mu^{-1}}(t_1), \ldots, \dot{\mu^{-1}}(t_m)) \tag{A1}$$

$$\forall x_1, \ldots, x_n.\ \mathbb{T}(A \underline{\wedge} B) \leftrightarrow (\mathbb{T}(A) \wedge \mathbb{T}(B)) \tag{A2}$$

$$\forall x_1, \ldots, x_n.\ \mathbb{T}(\underline{\neg} A) \leftrightarrow (\neg \mathbb{T}(A)) \tag{A3}$$

$$\forall x_1, \ldots, x_n.\ \mathbb{T}(\underline{\forall}(\underline{x}, A)) \leftrightarrow (\forall x.\ \mathbb{T}(A[\underline{x} \leftarrow \mathbb{Q}(x)]_q)) \tag{A4}$$

Range: $p \in P, t_1, \ldots, t_n \in \underline{\mathcal{T}_v}, A, B \in \mathcal{Q}_v, x \in V$
Schema A1 concerns the truth of the quotation of an atomic formula. Schema A2 and Schema A3 are about the Boolean connectives. Schema A4 handles universal quantification through substituion and the $\mathbb{Q}$ predicate; we illustrate this behavior in Example 8 bellow. Note that, different from $\mathrm{A}_{\text{truth}}$, Schemas A1-4 apply also to non-well-formed terms.

Example 7. *The formulas* $\mathbb{T}(\underline{\neg P}()) \leftrightarrow \neg\mathbb{T}(\underline{P}())$ *and* $\mathbb{T}(\underline{\neg} 2) \leftrightarrow \neg\mathbb{T}(2)$ *are both instances of A3.*

Axiom schema A4 is the treatment of the universal quantification. We substitute the quotation $\underline{x}$ of a variable x by $\mathbb{Q}(x)$. 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 8. *We show* A1-4$\models \mathbb{T}(\underline{\forall}(\underline{x}, \underline{P}(\underline{x}))) \leftrightarrow \forall x.\ P(x)$.
To prove this, let M *be a model of* A1-4*. We have:*

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

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

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

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

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

We detail the proof in the supplementary material. □

5.2 Axioms for reasoning in contexts

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.

$$\forall x_c, x_1, x_2.\ \mathit{ist}(x_c, x_1 \underline{\wedge} x_2) \to \mathit{ist}(x_c, x_1) \tag{A5}$$
$$\forall x_c, x_1, x_2.\ \mathit{ist}(x_c, x_1 \underline{\wedge} x_2) \leftrightarrow \mathit{ist}(x_c, x_2 \underline{\wedge} x_1) \tag{A6}$$
$$\forall x_c, x_1.\ \mathit{ist}(x_c, \underline{\neg\neg} x_1) \leftrightarrow \mathit{ist}(x_c, x_1) \tag{A7}$$
$$\forall x_c, x_1, x_2, x_3.\ \mathit{ist}(x_c, (x_1 \underline{\wedge} x_2) \underline{\wedge} x_3) \leftrightarrow \mathit{ist}(x_c, x_1 \underline{\wedge} (x_2 \underline{\wedge} x_3)) \tag{A8}$$
$$\forall x_c, x_1, x_2, x_3.\ \mathit{ist}(x_c, (x_1 \underline{\wedge} x_2) \underline{\vee} x_3) \leftrightarrow \mathit{ist}(x_c, (x_1 \underline{\vee} x_3) \underline{\wedge} (x_2 \underline{\vee} x_3)) \tag{A9}$$

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 ϕ and $\phi \Rightarrow \psi$, then it also knows ψ:

$$\forall x_c, x_1, x_2.\ \mathit{ist}(x_c, x_1 \underline{\vee} x_2) \wedge \mathit{ist}(x_c, \underline{\neg} x_1) \to \mathit{ist}(x_c, x_2) \tag{A10}$$

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.

$$\forall c.\ \mathit{ist}(c, \underline{\forall}(\underline{x}, \underline{\varphi})) \to \forall x.\ \mathit{ist}(c, \underline{\varphi}[\underline{x} \leftarrow \mathbb{Q}(x)]_q) \tag{A11}$$

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

We illustrate the use of schema A11 with the example of Romeo and Juliet from the introduction (*Cap* stands for being a member of Juliet's family, the Capulets):

Example 9. *Juliet believes that all Capulets are nice.*

$$\mathit{ist}(\mathit{believes}(J), \underline{\forall x.\ \mathit{Cap}(x) \to \mathit{nice}(x)}) \to \forall x.\ \mathit{ist}(\mathit{believes}(J), \underline{\mathit{Cap}}(\mathbb{Q}(x)) \underline{\to} \underline{\mathit{nice}}(\mathbb{Q}(x)))$$

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*

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

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 φ is semi-decidable.*

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

6. Examples and discussion

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 model time in this example. 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.

$$\forall \phi.\ \mathit{ist}(\mathit{says}(\mathit{FriarLaurence}), \phi) \to \mathbb{T}(\phi) \tag{1}$$

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

Next, we state some obvious facts from the tragedy:

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

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

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

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

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

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

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

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

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

We can now see the tragedy unfold:

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

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

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

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

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

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

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

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 but did not conclude that Juliet was alive.

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

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 $\varphi, \psi \in \mathcal{L}_q$):

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

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

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

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:

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

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:

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

6.4 On the use of the quote symbol $\mathbb{Q}$

The quote symbol $\mathbb{Q}$ is used to inject a value directly within a quotation. When we apply the unquote operator μ^{-1} to a quotation containing $\mathbb{Q}$, the content of $\mathbb{Q}$ will not be unquoted. For example:

$$\mu^{-1}(\underline{2}) = \mu^{-1}(\mathbb{Q}(2)) = 2$$

$$\mu^{-1}(\underline{P}(\mathbb{Q}(2))) = P(2)$$

To avoid any confusion, we recall that $\mathbb{Q}$ is a symbol of the logic not to be confused with μ, which is a meta operator called the 'quotation operator". μ exists outside the logic itself and returns the quotation of a given formula or term, preserving its structure. Most of the time, we write $\underline{\varphi}$ instead of $\mu(\varphi)$. By contrast, $\mathbb{Q}$ is just a function symbol within the logic. It represents a function of the domain of discourse, just like any other function symbol in FOL.
The two main purposes of $\mathbb{Q}$ are:

1. Nesting quotations
2. Using a variable inside a quotation

We first illustrate nested quotations. Let us say that Romeo believes that Juliet believes that he is smart. We can write that Juliet believes that Romeo is smart as follows:

$$\textit{ist}(\textit{believes}(J), \underline{\textit{Smart}(\underline{R})})$$

To say that Romeo believes that Juliet believes that he is smart, we need to quote the formula above. To this end, we need the symbol $\mathbb{Q}$:

$$\mathit{ist}(\mathit{believes}(R), \underline{\mathit{ist}}(\underline{\mathit{believes}}(\underline{J}), \mathbb{Q}(\underline{\mathit{Smart}}(\underline{R}))))$$

Using our notations, we could write this more compactly as follows:

$$\mathit{ist}(\mathit{believes}(R), \underline{\mathit{ist}(\mathit{believes}(J), \underline{\mathit{Smart}(R)})})$$

If we instead wrote the formula without $\mathbb{Q}$, we would end up with:

$$\mathit{ist}(\mathit{believes}(R), \underline{\mathit{ist}}(\underline{\mathit{believes}}(\underline{J}), \underline{\mathit{Smart}}(\underline{R})))$$

But $\underline{\mathit{ist}}(\underline{\mathit{believes}}(\underline{J}), \underline{\mathit{Smart}}(\underline{R}))$ is not a well-formed quotation. If we were to try to unquote it, we end up with the formula below. It is not a well-formed first-order formula, as *Smart* is a predicate and, therefore, cannot appear in the argument of the predicate *ist*.

$$\mu^{-1}(\underline{\mathit{ist}}(\underline{\mathit{believes}}(\underline{J}), \underline{\mathit{Smart}}(\underline{R}))) = \mathit{ist}(\mathit{believes}(J), \mathit{Smart}(R))$$

Hence we need the symbol $\mathbb{Q}$ to nest quotations.

Now we illustrate the second use of $\mathbb{Q}$: using a variable inside a quotation. Let us say that Romeo is very naive: all liars have successfully convinced him that they are honest. This does not mean that Romeo believes that "all liars are honest", which would be written as:

$$\mathit{ist}(\mathit{believes}(R), \underline{\forall}(c_x, \underline{\mathit{Liar}}(c_x) \underline{\rightarrow} \underline{\mathit{Honest}}(c_x)))$$

Instead, we want to say that for all liars, Romeo believes that person is honest. We can write this as:

$$\forall x.\, \mathit{Liar}(x) \rightarrow \mathit{ist}(\mathit{believes}(R), \underline{\mathit{Honest}}(\mathbb{Q}(x)))$$

Without the symbol $\mathbb{Q}$, we might try to write the following:

$$\forall x.\, \mathit{Liar}(x) \rightarrow \mathit{ist}(\mathit{believes}(R), \underline{\mathit{Honest}}(x))$$

But this does not work. We can notice this by instantiating the quantifier $\forall$ in the formula above.

$$\mathit{Liar}(\mathit{Juliet}) \rightarrow \mathit{ist}(\mathit{believes}(R), \underline{\mathit{Honest}}(\mathit{Juliet}))$$

This makes no sense because $\underline{\mathit{Honest}}(\mathit{Juliet})$ is not a well-formed quotation.

7. Finite axiomatization and theorem provers

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 $H_C = H \cup \text{A1-A4} \cup \text{A5-A10} \cup \text{A11}$ (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

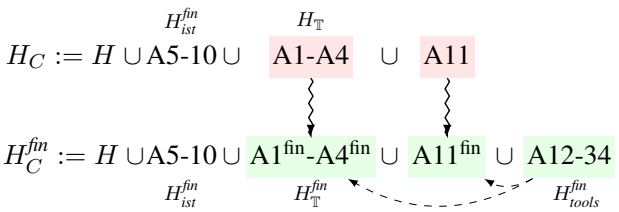

Figure 2: Overview of the process of finite axiomatization. Zizag arrows indicate the finite sets are counterparts to the top infinite ones. Dotted arrows indicate the the elements of H_{tools}^{fin} are used to define said counterparts.

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 must 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. Based on these symbols and their definition schemas, we introduce finite counterparts to our infinite schemas in Subsection 7.3. These are the sets $H_{\mathbb{T}}^{fin}$ and $H_{\underline{\forall}}$. Together, these sets allow us to define a new set $H_C^{fin} = H \cup$ A5-A10$\cup H_{tools}^{fin} \cup H_{\mathbb{T}}^{fin} \cup H_{\underline{\forall}}$, which is finite and equisatisfiable with H_C (see Figure 2). We present some interesting properties of the process in Subsection 7.4, the most important being the correctness of the process.

7.2 Utility symbols for the finite axiomatization

We will now introduce new symbols that will allow us to define the finite axiomatization of Qiana. We consider a fixed and finite theory H on S. Without loss of generality, we introduce two fresh function symbols *Sub* and E and three predicate symbols *Wft*, *Term*, and $=$. 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 $x, y, z, x_1, \ldots, x_n, y_1, \ldots, y_n$ are distinct variables:

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

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

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

$$\forall x.\ \textit{Term}(\mathbb{Q}(x)) \tag{A17}$$
$$\forall t_1, \ldots, t_n.\ (\textit{Term}(t_1) \wedge \cdots \wedge \textit{Term}(t_n)) \to \textit{Term}(f(t_1, \ldots, t_n)) \tag{A18}$$

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., $t \in \mathcal{Q}_v$):

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

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

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

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

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 $t_1[t_2 \leftarrow t_3]_q$:

$$\forall t.\ \mathit{Term}(t) \rightarrow \mathit{Sub}(\underline{x}, \underline{x}, t) = t \tag{A29}$$
$$\forall t.\ \mathit{Term}(t) \rightarrow \mathit{Sub}(\underline{x}, \underline{y}, t) = \underline{x} \tag{A30}$$
$$\forall t, t_1, .., t_n.\ (\mathit{Term}(t_1) \wedge .. \wedge \mathit{Term}(t_n)) \rightarrow$$
$$\mathit{Sub}(\underline{f}(t_1, .., t_n), \underline{x}, t) = \underline{f}(\mathit{Sub}(t_1, \underline{x}, t), .., \mathit{Sub}(t_n, \underline{x}, t)) \tag{A31}$$
$$\forall t_1, t_2.\ (\mathit{Term}(t_1) \wedge \mathit{Term}(t_2)) \rightarrow \mathit{Sub}(\underline{\forall}(\underline{x}, t_1), \underline{x}, t_2) = \underline{\forall}(\underline{x}, t_1) \tag{A32}$$
$$\forall t_1, t_2.\ (\mathit{Term}(t_1) \wedge \mathit{Term}(t_2)) \rightarrow \mathit{Sub}(\underline{\forall}(\underline{y}, t_1), \underline{x}, t_2) = \underline{\forall}(\underline{y}, \mathit{Sub}(t_1, \underline{x}, t_2)) \tag{A33}$$
$$\forall t_1, t_2.\ (\mathit{Term}(t_1) \wedge \mathit{Term}(t_2)) \rightarrow \mathit{Sub}(\mathbb{Q}(t_1), \underline{x}, t_2) = \mathbb{Q}(t_1) \tag{A34}$$

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 $H^{\mathit{fin}}_{\mathit{tools}}$:

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

7.3 Finite counterparts to infinite schemas

Let us write $H_{\mathbb{T}} = A1\text{-}4$. The set $H_{\mathbb{T}}$ defines the behavior of $\mathbb{T}$ 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 $H_{\mathbb{T}}^{fin}$:

Definition 10. $H_{\mathbb{T}}^{fin} = A1^{fin}\text{-}A4^{fin}$.

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

$$\forall t_1, t_2.\ (\mathit{Term}(t_1) \wedge \mathit{Term}(t_2)) \rightarrow \mathbb{T}(t_1 \underline{\wedge} t_2) \leftrightarrow (\mathbb{T}(t_1) \wedge \mathbb{T}(t_2)) \tag{A2$^{\text{fin}}$}$$

$$\forall t_1.\ \mathit{Term}(t_1) \rightarrow \mathbb{T}(\underline{\neg} t_1) \leftrightarrow (\neg \mathbb{T}(t_1)) \tag{A3$^{\text{fin}}$}$$

$$\forall t_1.\ \mathit{Term}(t_1) \rightarrow \mathbb{T}(\underline{\forall}(\underline{x}, t_1)) \leftrightarrow (\forall x.\ \mathbb{T}(\mathit{Sub}(t_1, \underline{x}, \mathbb{Q}(x)))) \tag{A4$^{\text{fin}}$}$$

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

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

$$\forall t_1, t_2.\ \mathit{Term}(t_1) \rightarrow \mathit{ist}(t_2, \underline{\forall}(\underline{x}, t_1)) \rightarrow \forall x.\ \mathit{ist}(t_2, \mathit{Sub}(t_1, \underline{x}, \mathbb{Q}(x))) \tag{A11$^{\text{fin}}$}$$

Range: $x \in V$

7.4 Correctness

Definition 11. *We can now formally define the finite axiomatization of Qiana on H as the set H_C^{fin}:*

$$H_C^{fin} := H \cup H_{ist}^{fin} \cup H_{tools}^{fin} \cup H_{\mathbb{T}}^{fin} \cup A11^{fin}$$

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

Theorem 1. *H_C is coherent if and only if H_C^{fin} 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 $H_C^{fin} \not\models \bot$. In the supplementary material, we prove both directions of this equivalence in Propositions 10 and 11. □

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

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

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 $H_{ist}^{fin} \cup H_{tools}^{fin} \cup H_{\mathbb{T}}^{fin}$ is in $\mathcal{O}(|S|^2)$, where $|S|$ is the total number of symbols in S, excluding $V_\infty \setminus V$.*

We introduced a finite number of quoted variables to make this finite axiomatization process possible. 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_C^n be the Qiana closure of H where V has size $n \in \mathbb{N}$, and let H_C^∞ be the Qiana closure of H obtained by allowing the set V to be infinite. Let φ be any well-formed closed formula. Then if $H_C^\infty \models \varphi$ then there is some $n \in \mathbb{N}$ such that $H_C^n \models \varphi$.*

Proof. Any proof derivation of φ from H_C^∞ uses finitely many formulas, which are all in H_C^n 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, we do not lose any deductive power.

7.5 Using Qiana in Theorem Provers

Our finite axiomatization allows us to transform the Qiana closure of any finite theory into an equi-satisfiable finite first-order logic theory, which can then be fed into an Automated Theorem Prover (ATP). 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 & 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.

8. Temporality in Qiana

In this section, we extend Qiana for temporal reasoning. In order to take advantage of the Qiana approach, we rely on event calculus which is also based on first-order logic. There are several modal logics to deal with time that are modal logic: linear-temporal logic (LTL), computational tree logic (CTL), etc. However, contrary to event calculus, time instants are not explicit in LTL/CTL, which makes them less powerful.

8.1 Overview of Event Calculus

Event calculus is a popular family of formalisms to represent actions and their effects on systems through time. In this article, we follow the definitions of Shanahan (2000), which we chose for its clarity and concision. Event calculus is based on the following concepts. *Fluents* are properties of the system that can change over time; there is typically a finite set of fluents under consideration. *Actions* (also called *Events*) occur at points in time or during time intervals and can change the value of fluents.

Here are a few example sentences written in classical event calculus and toying with the Romeo and Juliet story:

- The construction $\mathit{HoldsAt}(\mathit{Alive\_Romeo}, t)$, meaning that Romeo is alive at instant t. Here, the fluent is *Alive_Romeo*, and the time is t.
- The construction $\mathit{Happens}(\mathit{Drink\_Potion\_Juliet}, t_1, t_2)$ means that Juliet drinks a potion between time t_1 and t_2. Here the action is again *Drink_Potion_Juliet*.

- The construction *Happens*(*Drink_Potion_Juliet*, t) means that Juliet drinks a potion at time t. Here the action is *Drink_Potion_Juliet*. Note the operator overload on *Happens* with the previous statement. In fact, *Happens*(a, t) is simply syntactic sugar for *Happens*(a, t, t).

- The construction *Initiates*(*Appear_Dead_Juliet*, *Drink_Potion_Juliet*, t) means that drinking a potion makes Juliet start appearing dead at time t. Here the fluent is *Appear_Dead_Juliet* and the action is *Drink_Potion_Juliet*.

It should be noted that fluents are atomic and cannot be connected to form more complex fluents (there are no equivalents of $\neg$ or $\wedge$ on fluents). Also, this formalism features a notion of inertia. When a fluent becomes true it remains so unless it is "clipped", which is represented by a dedicated predicate *Clipped*. This behavior is formaly defined by axioms EC1, EC2, and EC3 in Subsection 8.2, which we present with Qiana notations. Table 2 lists and describes the main operators of event calculus.

Event Calculus operator	Description
0	The first time instant
$i_1 < i_2$	Instant i_1 occurs before instant i_2
Initially$(\underline{\varphi})$	Nontemporal quoted formula $\underline{\varphi}$ holds at the beginning
HoldsAt$(\underline{\varphi}, i_1)$	Nontemporal quoted formula $\underline{\varphi}$ holds at time i_1
Happens(a, i_1, i_2)	Action a happens between times i_1 and i_2
Initiates$(a, \underline{\varphi}, i_1)$	If action occurs at time i_1 it initiates $\underline{\varphi}$ at that time
Terminates$(a, \underline{\varphi}, i_1)$	If action occurs at time i_1 it terminates $\underline{\varphi}$ at that time
Releases$(a, \underline{\varphi}, i_1)$	$\underline{\varphi}$ is not subject to inertia after action a at time i_1
Clipped$(i_1, \underline{\varphi}, i_2)$	$\underline{\varphi}$ is terminated between times i_1 and i_2
Declipped$(i_1, \underline{\varphi}, i_2)$	$\underline{\varphi}$ is initiated between times i_1 and i_2

Table 2: Matching Event Calculus operators to their descriptions

This concludes our presentation of event calculus. In the next subsection, we describe how to adapt event calculus to Qiana.

8.2 Event calculus in Qiana

Because the full event calculus of Shanahan (2000) is based on first-order logic, its adaptation to Qiana will be relatively straightforward. We allow using any quoted formula as a fluent and write the axioms of event calculus in Qiana. This requires the introduction of a few new symbols to Qiana to match the operators of event calculus.

The event calculus presented in Shanahan (2000) also contains an operator *Initially*$_N$ to say that some fluent is initially false. Thanks to the quote symbol $\underline{\neg}$, we write *Initially*$(\underline{\neg}\ f)$ instead of *Initially*$_N(f)$.

Here are the axioms of event calculus in Qiana:

$$\forall f, t.\ \mathit{HoldsAt}(f,t) \leftarrow \mathit{Initially}_P(f) \wedge \neg \mathit{Clipped}(0,f,t) \tag{EC1}$$

$$\forall f, t_3, a, t_1, t_2.\ \mathit{HoldsAt}(f,t_3) \leftarrow \mathit{Happens}(a,t_1,t_2) \wedge \mathit{Initiates}(a,f,t_1) \wedge \neg \mathit{Clipped}(t_1,f,t_3) \wedge t_2 < t_3 \tag{EC2}$$

$$\forall t_1, f, t_4.\ \mathit{Clipped}(t_1,f,t_4) \leftrightarrow \exists a, t_2, t_3.\ \mathit{Happens}(a,t_2,t_3) \wedge (\mathit{Terminates}(a,f,t_2) \vee \mathit{Releases}(a,f,t_2)) \wedge t_1 < t_3 \wedge t_2 < t_4 \tag{EC3}$$

$$\forall f, t.\ \neg \mathit{HoldsAt}(f,t) \leftarrow \mathit{Initially}(\underline{\neg} f) \wedge \neg \mathit{Declipped}(0,f,t) \tag{EC4}$$

$$\forall f, t_3, a, t_1, t_2.\ \neg \mathit{HoldsAt}(f,t_3) \leftarrow \mathit{Happens}(a,t_1,t_2) \wedge \mathit{Terminates}(a,f,t_1) \wedge \neg \mathit{Declipped}(t_1,f,t_3) \wedge t_2 < t_3 \tag{EC5}$$

$$\forall t_1, f, t_4.\ \mathit{Declipped}(t_1,f,t_4) \leftrightarrow \exists a, t_2, t_3.\ \mathit{Happens}(a,t_2,t_3) \wedge (\mathit{Initiates}(a,f,t_2) \vee \mathit{Releases}(a,f,t_2)) \wedge t_1 < t_3 \wedge t_2 < t_4 \tag{EC6}$$

$$\forall a, t_1, t_2.\ \mathit{Happens}(a,t_1,t_2) \rightarrow t_1 \leq t_2 \tag{EC7}$$

Remark 1. *We do nothing to prevent the use of the quotation of temporal statements as fluent. For example, nothing explicitly prevents the use of* $\underline{\mathit{Happens}(\mathit{drink}(P,J),t)}$ *as a fluent. This is a choice we make to simplify the formalism, but we also make no special effort to give them a special meaning. Hence, they can be considered as any other meaningless quotation we could, in theory, pass to a temporal operator. Because no axiom allows their introduction in a temporal context, this creates no problem.*

8.3 Example

We will now adapt our running example of Romeo and Juliet (Section 6.1) to our temporal framework. We will tell the same story as before, but we will now account for time: At first Romeo is alive, then he sees Juliet, then he dies. We will use the following axioms:

$$\forall \varphi.\ \mathit{ist}(\mathit{says}(L),\varphi) \rightarrow \mathbb{T}(\varphi) \tag{18}$$

$$\mathit{ist}(\mathit{says}(L), \forall t.\ \mathit{Happens}(\mathit{drink}(P,J),t) \rightarrow \mathit{Initiate}(\mathit{drink}(J,P), \underline{\mathit{LookDead}}(J),t)) \tag{19}$$

$$t_{\mathit{drink}} < t_{\mathit{see}} \tag{20}$$

$$\mathit{Happens}(\mathit{drink}(P,J), t_{\mathit{drink}}) \tag{21}$$

$$\mathit{Happens}(\mathit{see}(R,J), t_{\mathit{see}}) \tag{22}$$

$$\neg \mathit{Clipped}(\mathit{LookDead}(J), t_{\mathit{drink}}, t_{\mathit{see}}) \tag{23}$$

$$\forall t_1, p.\ \mathit{HoldsAt}(\underline{\mathit{LookDead}}(\mathbb{Q}(p)), t_1) \rightarrow \mathit{Initiate}(\mathit{See}(R,p), \underline{\mathit{ist}(\mathit{bel}(R), \underline{\mathit{dead}(p)})}, t_1) \tag{24}$$

$$\forall t_1.\ \mathit{HoldsAt}(\mathit{ist}(\mathit{bel}(R), \underline{\mathit{dead}(J)}), t_1) \rightarrow \mathit{Happens}(\mathit{die}(R), t_1) \tag{25}$$

Note the different kinds of elements. The terms $\mathit{see}(R,J)$ and $\mathit{die}(R)$ are actions. $\mathit{LookDead}(J)$ is a formula (like $\mathit{dead}(J)$), its quotation $\underline{\mathit{LookDead}(J)}$ is a fluent. Two points in time are important to the story: the time t_{drink} when Juliet drinks the potion and the time t_{see} when Romeo sees Juliet. Formulas 18 and 19 behave similarly to their counterparts Formula 1 and Formula 5 in Section 6.1, except that Laurence now says Juliet will "start looking dead", rather than simply "look dead". Formula 21 states that Juliet drinks the potion, and formula 23 states that Juliet does not wake up before t_{see}. Hence Juliet looks dead at time t_{see}. Together with formulas 22 and 24 this tells us that Romeo starts believing Juliet is dead, which leads to his own death (Formula 25).

8.4 The frame problem and differences between Qiana and Event Calculus

The frame problem is a classic issue in formalisms that model change over time, and it is the reason for a variation in the semantics between the event calculus of Shanahan (2000) and Qiana.

To explain this difference, we begin by explaining the frame problem. The frame problem arises because things generally remain unchanged unless something happens to them to make them change. This is referred to as inertia—the natural tendency of systems to stay in their current state. The challenge is how to formalize this idea without having to explicitly list every possible situation in which something might change.

The solution used by event calculus is circumscription. Entailment in event calculus relies on axioms, normal first-order logic (FOL) entailment, and a function called *circumscription*. The idea is to use $\mathit{Circum}(\Gamma)$ which represents an extansion of Γ with reasonable assumptions. The full definition of $\mathit{Circum}(\Gamma)$ is omitted here and we refer the reader to Shanahan (2000) for a more complete explanation. The two following facts are important:

$$\mathit{Circum}(\Gamma) \models \Gamma$$

$$\Gamma \models_{EC} \varphi \Leftrightarrow \mathit{Circum}(\Gamma) \cup \mathit{EC} \models \varphi$$

where $\models_{EC}$ is the entailment of event calculus and *EC* the set of axioms of event calculus.

In a sense, the entailment of event calculus (with circumscription) is weaker than normal FOL entailment. Valid FOL entailment still holds under it, but the entailment of event calculus also makes additional assumptions.

We do not port this entailment to Qiana; instead, we only bring the axioms of event calculus to Qiana, which we use with normal FOL entailment. Therefore, our form of entailment is stronger than the one of event calculus, meaning it is sound but not complete with respect to it. We prove this with Proposition 8.

Proposition 8. *Let Γ be a set of formulas valid under event calculus. Let H_{TQ} be the set of axioms of temporal Qiana, and let φ be a temporal formula valid under event calculus. Then:*

$$\Gamma \cup H_{TQ} \models \varphi \implies \Gamma \models_{EC} \varphi$$

Proof. Suppose $\Gamma \cup H_{TQ} \models \varphi$. Then, we have:

$$\mathit{Circum}(\Gamma_{ECQ}) \cup H_{TQ} \models \varphi$$

Since the temporal axioms of Qiana follow the axioms of event calculus, we obtain:

$$\mathit{Circum}(\Gamma_{ECQ}) \cup H_{TQ} \models \varphi \implies \mathit{Circum}(\Gamma_{ECQ}) \cup \mathit{EC} \models \varphi$$

By definition, this is equivalent to:

$$\Gamma \models_{EC} \varphi$$

□

Considering that temporality and contexts are largely orthogonal in Qiana, nothing stops us from defining a circumscription operator in Qiana and using it to define another notion of entailment for

our logic. However, the fact entailment in Qiana is just FOL entailment with certain axioms is an important feature of the logic. Therefore, we consider the simple application of the axioms of event calculus to be the most appropriate way to model temporal reasoning in Qiana. The temporal version of Qiana follows event calculus but does not include its additional assumptions regarding inertia.

9. Typing Qiana

A Qiana theory uses different types of objects: formulas, quoted formulas, terms, and (together with event calculus) events, actions, and fluents. The boundary between these types is sometimes porous: One way to make the distinction clearer is to resort to typed first-order logic (also known as many-sorted FOL). In what follows, we define a typed version of Qiana that distinguishes different types of objects more clearly. This typed version of Qiana can be considered more intuitive, and the ability to distinguish natively between different types could be useful in concrete applications with complex rules mixing different types of objects. However, types make formulas longer and create multiple technical issues that have to be handled throughout the entire process of typing Qiana. In particular, the finite axiomatization process of Section 7 becomes more complex and much longer (see Subsection 9.4). Hence, we define the typed version of Qiana here merely as a theoretical exercise.

9.1 A quick summary of typed first-order logic

Typed FOL (first-order logic) is sometimes also called many-sorted FOL. We will give only a quick summary of the topic here and redirect the unfamiliar reader to the literature (Manzano, 1996). Many variations on typed FOL have been proposed, but we will use the basic one simply called "many sorted FOL", which is what we present below.

In many-sorted FOL, the signature includes a finite set U of types. All function symbols and predicate symbols have a signature, indicating the types of their arguments and the type of the output in the case of functions. Variable symbols also have an associated type and we assume there are countably many symbols of each type. The type of a term is the type of its top-level symbol (a function symbol or a variable). A term can be used as an argument to a predicate or function only if it has the correct type, as indicated by the signature of the predicate or function. Models of many-sorted FOL are similar to models of unsorted FOL, but the domain of the model is partitioned into disjoint sets, one for each type.

Remark 2. *Ideally, we would have preferred to use a flavor of typed FOL that allows non-disjoint types, such as order-sorted FOL. However, as far as we know, all ATPs (Automated Theorem Provers) that support TPTP input (and, in fact, all ATPs we know of) support only disjoint types. In order to maintain the compatibility of Qiana with existing ATPs, we will have to use disjoint types.*

9.2 Types for Qiana

Table 3 introduces the types used to produce typed-Qiana, along with their short descriptions.

We extend the FOL signature (F, P, V_∞, δ) to $(F, P, V_\infty, \delta, U)$ by adding a finite set of types U, and adapting δ and V_∞. We have $U = \{o, q, c, \tau, a\}$.

Type	Name	Description
o	*Objects*	Non-Qiana objects (people, places, ...)
q	*Quotations*	Type of all quotations (quoted formulas, quoted terms, ...)
c	*Contexts*	Contexts
τ	*Time instant*	Event calculus points in time.
a	*Actions*	Event calculus actions

Table 3: Types and their descriptions in Qiana

Whereas δ gave the arity of symbols so far, now δ takes the types into account. More precisely, δ gives a signature to each predicate and function symbol, and a type to each variable symbol.

$$\begin{aligned}&\text{for each } p \in P,\ \delta(p) \in U^\star\\&\text{for each } f \in F,\ \delta(f) \in U^\star \times U\\&\text{for each } v \in V_\infty,\ \delta(v) \in U\end{aligned}$$

where $\delta(f) \in U^\star \times U$ means that f has a signature of the form $\gamma_1 \times \cdots \times \gamma_n \to \gamma_o$, with n the arity of f and $\gamma_1, \ldots, \gamma_n, \gamma_o \in U$. For example $\delta(\mathit{ist}) = c \times q$.

Also, we assume that there is an infinity of variable symbols of each type:

$$\text{for each } \gamma \in U,\ |\{\, v \in V_\infty \mid \delta(v) = \gamma \,\}| = +\infty$$

Remark 3. *If needed in practical applications, we can always split the type o into multiple subtypes. Here, we present everything with a single type for non-Qiana-specific objects; this is without loss of generality, and everything can be straightforwardly adapted to the case where o is split into many subtypes.*

We introduce multiple notations to indicate the types of symbols we use. We will alternate between the notations depending on which are most convenient for the formulas we write.

First, we can directly indicate the type of one or multiple symbols in a separate line: The following line indicates that f takes as arguments an object and a time instant, and returns a context, that p takes a context, and that p_2 has no argument (i.e., it is an atomic predicate).

$$f : o \times \tau \to c; p : c; p_2 : ()$$

We can also indicate the type of a symbol by writing the type as an exponent. In the following formula, t_1 and t_2 are variables of type o, t_3 is a variable of type c, and p can implicitly be deduced to be a predicate of signature $o^2 \times c$.

$$\forall t_1^o, t_2^o, t_3^c.\ p(t_1^o, t_2^o, t_3^c)$$

We can indicate the type of variables during quantification. The following formula is equivalent to the previous one:

$$\forall^o t_1, t_2.\ \forall^c t_3.\ p(t_1, t_2, t_3)$$

Lastly, in some contexts, the type can be clearly inferred from the context, and therefore, no additional type annotations are necessary.

In this section, we will first always include a type indication. This is because the first axioms we present will also serve as examples of our typing notations. However, we will gradually omit type annotations as they would only make our formulas less readable without any gain in clarity.

Example 10. *In the following, we explicitly type each symbol as an example. In total, this formula contains terms of type q, c, τ, and a.*

$$\textit{between} : \tau \times \tau \to c$$
$$\varphi : () \to q$$
$$a : () \to a$$

$$\forall t_1^\tau, t_2^\tau.\ \textit{ist}(\textit{between}(t_1, t_2), \textit{Initiates}(a, \varphi)) \to \textit{ist}(\textit{between}(t_1, t_2), \varphi)$$

9.3 Typing the general Qiana axioms

We now present a typed version of the general Qiana axioms presented in Section 5, using the types introduced in Table 3.

$$\forall x_1^{\gamma_1}, \ldots, x_n^{\gamma_n}.\ \mathbb{T}(\underline{p}(t_1, \ldots, t_m)) \leftrightarrow p(\mu^{\vec{1}}(t_1), \ldots, \mu^{\vec{1}}(t_m)) \tag{A35}$$

Range: $p \in P, t_1, .., t_n \in \underline{\mathcal{T}_v}$ with $\forall i.\ \delta(\mu^{\vec{1}}(t_i)) = \delta(p)_i, x_1^{\gamma_1}, .., x_n^{\gamma_n}$ the free variables in $t_1, .., t_m$

$$\forall x_1^{\gamma_1} \ldots, x_n^{\gamma_n}\ \mathbb{T}(A \underline{\wedge} B) \leftrightarrow (\mathbb{T}(A) \wedge \mathbb{T}(B)) \tag{A36}$$
$$\forall x_1^{\gamma_1} \ldots, x_n^{\gamma_n}\ \mathbb{T}(\underline{\neg} A) \leftrightarrow (\neg \mathbb{T}(A)) \tag{A37}$$
$$\forall x_1^{\gamma_1} \ldots, x_n^{\gamma_n}\ \mathbb{T}(\underline{\forall}(\underline{x}, A)) \leftrightarrow (\forall^\gamma x.\ \mathbb{T}(A[\underline{x} \leftarrow \mathbb{Q}_\gamma(x)]_q)) \tag{A38}$$

Range: $A, B \in \mathcal{Q}_v, x_1^{\gamma_1} \ldots, x_n^{\gamma_n}$ the free variables in $A, B, \underline{x}$ a quoted variable of type γ

$$\forall^c x_c, \forall^q x_1, x_2.\ \textit{ist}(x_c, x_1 \underline{\wedge} x_2) \to \textit{ist}(x_c, x_1) \tag{A39}$$
$$\forall^c x_c, \forall^q x_1, x_2.\ \textit{ist}(x_c, x_1 \underline{\wedge} x_2) \leftrightarrow \textit{ist}(x_c, x_2 \underline{\wedge} x_1) \tag{A40}$$
$$\forall^c x_c, \forall^q x_1.\ \textit{ist}(x_c, \underline{\neg}\underline{\neg} x_1) \leftrightarrow \textit{ist}(x_c, x_1) \tag{A41}$$
$$\forall^c x_c, \forall^q x_1, x_2, x_3.\ \textit{ist}(x_c, (x_1 \underline{\wedge} x_2) \underline{\wedge} x_3) \leftrightarrow \textit{ist}(x_c, x_1 \underline{\wedge} (x_2 \underline{\wedge} x_3)) \tag{A42}$$
$$\forall^c x_c, \forall^q x_1, x_2, x_3.\ \textit{ist}(x_c, (x_1 \underline{\wedge} x_2) \underline{\vee} x_3) \leftrightarrow \textit{ist}(x_c, (x_1 \underline{\vee} x_3) \underline{\wedge} (x_2 \underline{\vee} x_3)) \tag{A43}$$
$$\forall^c x_c, \forall^q x_1, x_2.\ \textit{ist}(x_c, x_1 \underline{\vee} x_2) \wedge \textit{ist}(x_c, \underline{\neg} x_1) \to \textit{ist}(x_c, x_2) \tag{A44}$$
$$\forall^c x_c.\ \textit{ist}(x_c, \underline{\forall}(\underline{x}, \underline{\varphi})) \to \forall^\gamma x.\ \textit{ist}(x_c, \underline{\varphi}[\underline{x} \leftarrow \mathbb{Q}_\gamma(x)]_q) \tag{A45}$$

Range: $\underline{x}$ a quoted variable of type $\gamma, \underline{\forall}(\underline{x}, \underline{\varphi}) \in \underline{\mathcal{L}}$

9.4 Typing the finite axiomatization

We now present a typed version of the finite axiomatization presented in Section 7, using the types introduced in Table 3. Most are straightforward adaptations of the ones from Section 7. There is a single type q for all quotations, but the core idea of the finite axiomatization process is to go

over the quoted formula and interpret them recursively within the top-level domain of discourse. In particular, the function E matches a quoted term to the actual term it is a quotation of. Because terms can have different types, we need to create multiple versions of E with different output types. We will write E_γ for the version of E that matches the quotation of a term of type γ to said term.

Also, we need to rework the axiomatization of *Wft* so that it also checks that the quotation is well-typed. To do so, we introduce one instance of *Wft* per type γ, which we write Wft_γ.

Likewise, we introduce multiple versions of the equality predicate = and the reachability predicate *Term*.

$$\forall^\gamma x.\ x =_\gamma x \tag{A46}$$

$$\forall^\gamma x, y.\ x =_\gamma y \to y =_\gamma x \tag{A47}$$

$$\forall^\gamma x, y, z.\ x =_\gamma y \wedge y =_\gamma z \to x =_\gamma z \tag{A48}$$

$$\forall x_1^{\gamma_1}, .., x_n^{\gamma_n}, y_1^{\gamma_1}, .., y_n^{\gamma_n}.\ x_1^{\gamma_1} =_{\gamma_1} y_1^{\gamma_1} \wedge .. \wedge x_n^{\gamma_n} = y_n^{\gamma_n} \to f(x_1^{\gamma_1}, .., x_n^{\gamma_n}) =_{\gamma_o} f(y_1^{\gamma_1}, .., y_n^{\gamma_n}) \tag{A49}$$

$$\forall x_1^{\gamma_1}, .., x_n^{\gamma_n}, y_1^{\gamma_1}, .., y_n^{\gamma_n}.\ x_1^{\gamma_1} =_{\gamma_1} y_1^{\gamma_1} \wedge .. \wedge x_n^{\gamma_n} = y_n^{\gamma_n} \to p(x_1^{\gamma_1}, .., x_n^{\gamma_n}) \leftrightarrow p(y_1^{\gamma_1}, .., y_n^{\gamma_n}) \tag{A50}$$

Range: $f \in F, p \in P, \gamma, \gamma_o, \gamma_1, \ldots, \gamma_n \in U, \delta(f) = \gamma_1 \times \cdots \times \gamma_n \to \gamma_o, \delta(p)\gamma_1 \times \cdots \times \gamma_n$

$$\forall^\gamma x.\ \mathit{Term}_q(\mathbb{Q}_\gamma(x)) \tag{A51}$$

$$\forall t_1, \ldots, t_n.\ (\mathit{Term}_{\gamma_1}(t_1) \wedge \cdots \wedge \mathit{Term}_{\gamma_n}(t_n)) \to \mathit{Term}_{\gamma_o}(f(t_1, \ldots, t_n)) \tag{A52}$$

Range: $f \in F, \gamma, \gamma_o, \gamma_1, ..., \gamma_n \in U$

$$\forall y^\gamma.\ \mathit{Wft}_\gamma(\mathbb{Q}_\gamma(y^\gamma)) \tag{A53}$$

$$\mathit{Wft}_\gamma(\underline{x}) \tag{A54}$$

$$\forall^q t_1, \ldots, t_n.\ (\mathit{Wft}_{\gamma_1}(t_1) \wedge \cdots \wedge \mathit{Wft}_{\gamma_n}(t_n)) \to \mathit{Wft}_{\gamma_o}(\underline{f}(t_1, \ldots, t_n)) \tag{A55}$$

Range: $\underline{x} \in \underline{V}$ a quoted variable of type γ, $\underline{f} \in \underline{F}$ such that $\delta(f) = \gamma_1 \times ... \times \gamma_n \to \gamma_o, \gamma \in U$

$$\forall^\gamma t.\ \mathit{Term}_\gamma(t) \to E_\gamma(\mathbb{Q}_\gamma(t)) = t \tag{A56}$$

$$\forall t_1, .., t_n.\ \mathit{Term}(t_1) \wedge .. \wedge \mathit{Term}(t_n) \to E(\underline{f}(t_1, .., t_n)) = f(E(t_1), .., E(t_n)) \tag{A57}$$

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

$$\forall t.\ \mathit{Term}_q(t) \to \mathit{Sub}(\underline{x}, \underline{x}, t) = t \tag{A58}$$

$$\forall t.\ \mathit{Term}_q(t) \to \mathit{Sub}(\underline{x}, \underline{y}, t) = \underline{x} \tag{A59}$$

$$\forall t, t_1, \ldots, t_n.\ (\mathit{Term}_q(t_1) \wedge \cdots \wedge \mathit{Term}_q(t_n)) \to \mathit{Sub}(\underline{f}(t_1, \ldots, t_n), \underline{x}, t) = \underline{f}(\mathit{Sub}(t_1, \underline{x}, t), \ldots, \mathit{Sub}(t_n, \underline{x}, t)) \tag{A60}$$

$$\forall t_1, t_2.\ (\mathit{Term}_q(t_1) \wedge \mathit{Term}_q(t_2)) \to \mathit{Sub}(\underline{\forall}(\underline{x}, t_1), \underline{x}, t_2) = \underline{\forall}(\underline{x}, t_1) \tag{A61}$$

$$\forall t_1, t_2.\ (\mathit{Term}_q(t_1) \wedge \mathit{Term}_q(t_2)) \to \mathit{Sub}(\underline{\forall}(\underline{y}, t_1), \underline{x}, t_2) = \underline{\forall}(\underline{y}, \mathit{Sub}(t_1, \underline{x}, t_2)) \tag{A62}$$

$$\forall t_1^\gamma, t_2^q.\ (\mathit{Term}_\gamma(t_1) \wedge \mathit{Term}_q(t_2)) \to \mathit{Sub}(\mathbb{Q}_\gamma(t_1), \underline{x}, t_2) = \mathbb{Q}_\gamma(t_1) \tag{A63}$$

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

Definition 12. $H_{\mathbb{T}}^{fin} = A1^{fin}\text{-}A4^{fin}$

$$\forall t_1, .., t_n.\ (\mathit{Wft}_{\gamma_1}(t_1) \wedge .. \wedge \mathit{Wft}_{\gamma_n}(t_n)) \rightarrow \mathbb{T}(\underline{p}(t_1, .., t_n)) \leftrightarrow p(E_{\gamma_1}(t_1), .., E_{\gamma_n}(t_n)) \quad (\text{A1}^{\text{fin}})$$
$$\forall t_1, t_2.\ (\mathit{Term}_q(t_1) \wedge \mathit{Term}_q(t_2)) \rightarrow \mathbb{T}(t_1 \underline{\wedge} t_2) \leftrightarrow (\mathbb{T}(t_1) \wedge \mathbb{T}(t_2)) \quad (\text{A2}^{\text{fin}})$$
$$\forall t_1.\ \mathit{Term}_q(t_1) \rightarrow \mathbb{T}(\underline{\neg} t_1) \leftrightarrow (\neg \mathbb{T}(t_1)) \quad (\text{A3}^{\text{fin}})$$
$$\forall t_1.\ \mathit{Term}_q(t_1) \rightarrow \mathbb{T}(\underline{\forall}(\underline{x}, t_1)) \leftrightarrow (\forall x^{\gamma}.\ \mathbb{T}(\mathit{Sub}(t_1, \underline{x}, quote_{\gamma}(x)))) \quad (\text{A4}^{\text{fin}})$$

Range: $p \in P \setminus \{\mathbb{T}\}, \gamma, \gamma_1, ..., \gamma_n \in U, \underline{x} \in \underline{V}$ of type γ

$$\forall t_1, t_2^c.\ \mathit{Term}(t_1) \rightarrow \mathit{ist}(t_2^c, \underline{\forall}(\underline{x}, t_1)) \rightarrow \forall x^{\gamma}.\ \mathit{ist}(t_2^c, \mathit{Sub}(t_1, \underline{x}, \mathbb{Q}_{\gamma}(x))) \quad (\text{A11}^{\text{fin}})$$

Range: $\gamma \in U, \underline{x} \in \underline{V}$ of type γ

9.5 Typing the temporal axioms

Lastly, we type the axioms of temporal-Qiana (Section 8) with the types introduced in Table 3.

$$\forall f^q, t^{\tau}.\ \mathit{HoldsAt}(f^q, t^{\tau}) \leftarrow \mathit{Initially}_P(f^q) \wedge \neg \mathit{Clipped}(0, f^q, t^{\tau}) \quad (\text{EC1})$$
$$\forall f^q, t_3^{\tau}, a^a, t_1^{\tau}, t_2^{\tau}.\ \mathit{HoldsAt}(f^q, t_3^{\tau}) \leftarrow \mathit{Happens}(a^a, t_1^{\tau}, t_2^{\tau}) \wedge \mathit{Initiates}(a^a, f^q, t_1^{\tau}) \wedge \neg \mathit{Clipped}(t_1^{\tau}, f^q, t_3^{\tau}) \wedge t_2^{\tau} < t_3^{\tau} \quad (\text{EC2})$$
$$\forall t_1^{\tau}, f^q, t_4^{\tau}.\ \mathit{Clipped}(t_1^{\tau}, f^q, t_4^{\tau}) \leftrightarrow \exists a^a, t_2^{\tau}, t_3^{\tau}.\ \mathit{Happens}(a^a, t_2^{\tau}, t_3^{\tau}) \wedge (\mathit{Terminates}(a^a, f^q, t_2^{\tau}) \vee \mathit{Releases}(a^a, f^q, t_2^{\tau})) \wedge t_1^{\tau} < t_3^{\tau} \wedge t_2^{\tau} < t_4^{\tau} \quad (\text{EC3})$$
$$\forall f^q, t^{\tau}.\ \neg \mathit{HoldsAt}(f^q, t^{\tau}) \leftarrow \mathit{Initially}(\underline{\neg} f^q) \wedge \neg \mathit{Declipped}(0, f^q, t^{\tau}) \quad (\text{EC4})$$
$$\forall f^q, t_3^{\tau}, a^a, t_1^{\tau}, t_2^{\tau}.\ \neg \mathit{HoldsAt}(f^q, t_3^{\tau}) \leftarrow \mathit{Happens}(a^a, t_1^{\tau}, t_2^{\tau}) \wedge \mathit{Terminates}(a^a, f^q, t_1^{\tau}) \wedge \neg \mathit{Declipped}(t_1^{\tau}, f^q, t_3^{\tau}) \wedge t_2^{\tau} < t_3^{\tau} \quad (\text{EC5})$$
$$\forall t_1^{\tau}, f^q, t_4^{\tau}.\ \mathit{Declipped}(t_1^{\tau}, f^q, t_4^{\tau}) \leftrightarrow \exists a^a, t_2^{\tau}, t_3^{\tau}. \quad (\text{EC6})$$
$$\mathit{Happens}(a^a, t_2^{\tau}, t_3^{\tau}) \wedge (\mathit{Initiates}(a^a, f^q, t_2^{\tau}) \vee \mathit{Releases}(a^a, f^q, t_2^{\tau})) \wedge t_1^{\tau} < t_3^{\tau} \wedge t_2^{\tau} < t_4^{\tau}$$
$$\forall a^a, t_1^{\tau}, t_2^{\tau}.\ \mathit{Happens}(a^a, t_1^{\tau}, t_2^{\tau}) \rightarrow t_1^{\tau} \leq t_2^{\tau} \quad (\text{EC7})$$

10. Qiana as a modal logic

We now show that modal logic can be mirrored in Qiana.

10.1 Background on modal logic

For our purposes, the set of modal formulas is defined by adding the arity one "necessity" operator $\Box$ to the inductive definition of propositional formulas. The counterpart "possibility" operator $\Diamond$ is then defined by $\Diamond\varphi \Leftrightarrow \neg\Box\neg\varphi$. We recall the usual axioms of formal logics and the definitions

of the associated systems (K, T, $S4$, $S5$, D) (Chellas, 1980; Blackburn, Rijke, & Venema, 2001; Garson, 2024). These are the axioms:

$$\begin{aligned} K &: \Box(p \to q) \to (\Box p \to \Box q) \\ T &: \Box p \to p \\ 4 &: \Box p \to \Box\Box p \\ 5 &: \Diamond p \to \Box\Diamond p \\ D &: \Box p \to \Diamond p \end{aligned}$$

There is also a rule called *necessitation*, which states that if a formula is a tautology of the system **K** (see below), then it can be inferred from the axioms:

$$N : \frac{\mathbf{K} \models p}{\Box p}$$

These are the systems:

$$\begin{aligned} \mathbf{K} &= K + N \\ \mathbf{T} &= \mathbf{K} + T \\ \mathbf{S4} &= \mathbf{T} + 4 \\ \mathbf{S5} &= \mathbf{T} + 5 \\ \mathbf{D} &= \mathbf{K} + D \end{aligned}$$

10.2 Translation into Qiana

We can translate these usual propositional modal logics to Qiana. For the purpose of this section, we assume an augmented signature with no functions or predicates except for arity 0 predicates – which we take as our propositional variables – and the various predicates and functions necessary for the definition of Qiana or introduced in this section.

We introduce a special context $\Box$ for modal logic. We introduce the following notations:

$$\Box\underline{\varphi} \Leftrightarrow \mathit{ist}(\Box, \underline{\varphi})$$

$$\Diamond\underline{\varphi} \Leftrightarrow \neg\mathit{ist}(\Box, \underline{\neg\varphi})$$

Thanks to these notations, we can interpret a modal formula as a Qiana formula. To adapt the axioms K, T, 4, 5, and D to Qiana, we need to explicitly quantify on quoted formulas and to replace any logical connective with its quotation. Because $\Box$ and $\Diamond$ are contexts, we add a level of quotation per level of nesting.

$$\begin{aligned} QK &: \forall p, q.\, \Box(p\underline{\to}q) \to (\Box p \to \Box q) \\ QT &: \forall p.\, \Box p \to \mathbb{T}(p) \\ Q4 &: \forall p.\, \Box p \to \Box\underline{\Box p} \\ Q5 &: \forall p.\, \Diamond p \to \Box\underline{\Diamond p} \\ QD &: \forall p.\, \Box p \to \Diamond p \end{aligned}$$

To adapt the necessitation rule N to Qiana, we will need to add a new predicate *Tauto*. Where *Tauto*($\underline{\varphi}$) holds if and only if *Qiana* $\models \varphi$, where *Qiana* is the set of Qiana axioms on the signature at hand. Using this predicate, we can write the necessitation rule as follows:

$$QN: \ \forall p.\ \textit{Tauto}(p) \to \Box p$$

To define *Tauto* we introduce the predicate Wff to represent well-formed formulas:

$$\text{Wff}(\underline{p}()) \tag{T1}$$
$$\forall A.\ \text{Wff}(A) \to \text{Wff}(\underline{\neg} A) \tag{T2}$$
$$\forall A, B.\ \text{Wff}(A) \wedge \text{Wff}(B) \to \text{Wff}(A \underline{\wedge} B) \tag{T3}$$
$$\forall A.\ \text{Wff}(A) \to \text{Wff}(\underline{ist}(\underline{\Box}, \mathbb{Q}(A))) \tag{T4}$$

Range: $p \in P$ with $|\delta(p)| = 0$

We adapt a simple Hilbert-style proof system of propositional logic (Mendelson, 2009) to Qiana.

$$A \to (B \to A)$$
$$(A \to (B \to C)) \to ((A \to B) \to (A \to C))$$
$$(\neg B \to \neg A) \to ((\neg B \to A) \to B)$$

This can be done with the following axioms:

$$\forall A, B.\ \text{Wff}(A) \wedge \text{Wff}(B) \to \textit{Tauto}(A \underline{\to} (B \underline{\to} A)) \tag{T5}$$
$$\forall A, B, C.\ \text{Wff}(A) \wedge \text{Wff}(B) \wedge \text{Wff}(C) \to \textit{Tauto}((A \underline{\to} (B \underline{\to} C)) \underline{\to} ((A \underline{\to} B) \underline{\to} (A \underline{\to} C))) \tag{T6}$$
$$\forall A, B.\ \text{Wff}(A) \wedge \text{Wff}(B) \to \textit{Tauto}((\neg B \underline{\to} \neg A) \underline{\to} ((\neg B \underline{\to} A) \underline{\to} B)) \tag{T7}$$

We also need to apply the *modus ponens* rule to *Tauto*:

$$\forall A, B.\ \textit{Tauto}(A \underline{\to} B) \wedge \textit{Tauto}(A) \to \textit{Tauto}(B) \tag{T8}$$

Lastly, there is a more complex issue to consider. The necessitation rule N reacts to what is tautological within the system **K** and not only what is generally tautological in classic propositional logic. Hence, we need to include rules to extend *Tauto* to tautologies of **K** rather than only tautologies of classical propositional logic.

$$\forall A.\ \textit{Tauto}(A) \to \textit{Tauto}(\underline{ist}(\underline{\Box}, \mathbb{Q}(A))) \tag{T9}$$
$$\forall A, B.\ \textit{Tauto}(\underline{ist}(\underline{\Box}, A \underline{\to} B)) \to \textit{Tauto}(\underline{ist}(\underline{\Box}, A) \underline{\to} \underline{ist}(\underline{\Box}, B)) \tag{T10}$$

Together, these axioms define the behavior of *Tauto*.

Definition 13. *We define $H_{tauto} = T1 - T10$, the set of axioms that define the behavior of Tauto.*

Lemma 1 (Definition of *Tauto*)**.** *If modal formula φ is a tautology under* **K***, then $H_{tauto} \models$ Tauto(φ).*

Proof. Any tautology of **K** can be proven by a finite number of applications of the axioms of **K** and the rules *modus ponens* and necessitation. These admit counterparts under H_{tauto}, which means each step of such a proof is a valid inference under H_{tauto}. □

We now define sets of Qiana axioms corresponding to the various systems of modal logic: Let H_{Q} be the Qiana axioms on the signature at hand. Then, we can define the following systems:

$$\begin{aligned} H_{\mathbf{QK}} &= H_{\mathrm{Q}} + H_{\text{tauto}} + QK + QN \\ H_{\mathbf{QT}} &= H_{\mathbf{QK}} + QT \\ H_{\mathbf{Q4}} &= H_{\mathbf{QT}} + Q4 \\ H_{\mathbf{Q5}} &= H_{\mathbf{QT}} + Q5 \\ H_{\mathbf{QD}} &= H_{\mathbf{QK}} + QD \end{aligned}$$

We prove in Proposition 9 that these systems are equivalent to the usual systems of modal logic.

Proposition 9. *Let φ be a formula propositional modal formula. Then :*

$$\begin{aligned} H_{\boldsymbol{QK}} \models \varphi \textit{ if and only if } \boldsymbol{K} \models \varphi \\ H_{\boldsymbol{QT}} \models \varphi \textit{ if and only if } \boldsymbol{T} \models \varphi \\ H_{\boldsymbol{Q4}} \models \varphi \textit{ if and only if } \boldsymbol{S4} \models \varphi \\ H_{\boldsymbol{Q5}} \models \varphi \textit{ if and only if } \boldsymbol{S5} \models \varphi \\ H_{\boldsymbol{QD}} \models \varphi \textit{ if and only if } \boldsymbol{D} \models \varphi \end{aligned}$$

Proof. We describe the proof that $H_{\mathbf{QK}} \models \varphi$ if and only if $\mathbf{K} \models \varphi$. The other cases are similar.

We first prove that if $\mathbf{K} \cup \Gamma \models \bot$ for some set Γ of modal formulas then $H_{\mathbf{QK}} \cup \Gamma \models \bot$.

If $\mathbf{K} \cup \Gamma \models \bot$, then there is some proof of $\bot$ from Γ using only the axiom K, the axioms of classical propositional logic listed above, and the rules *modus ponens* and necessitation. Each of these axioms and rules has a counterpart in the formulas of $H_{\mathbf{QK}}$. The only nontrivial case is the necessitation rule N which relies on *Tauto*, the behavior of which was shown in Lemma 1. Hence, each step of the proof is a valid inference under $H_{\mathbf{QK}}$. Therefore, $H_{\mathbf{QK}} \cup \Gamma \models \bot$.

Now we prove the other direction; if $\mathbf{K} \cup \Gamma \not\models \bot$ then $H_{\mathbf{QK}} \cup \Gamma \not\models \bot$. Assume $\mathbf{K} \cup \Gamma \not\models \bot$. Then there is a model M of $\mathbf{K} \cup \Gamma$. We can adapt M to a model M_Q of $H_{\mathbf{QK}} \cup \Gamma$. We briefly describe M_Q:

- The truth value of every arity 0 predicate is the same as in M.
- For every formula φ of modal logic, we have $M_Q \models \Box\varphi$ if and only if $M \models \Box\varphi$. This is coherent with all axioms of $H_{\mathbf{QK}}$.
- Wff is true on every quotation of modal formula and false otherwise. This is coherent with the axioms of $H_{\mathbf{QK}}$.
- *Tauto* is true on every tautology of $\mathbf{K}$ and false otherwise. This is coherent with the axioms of $H_{\mathbf{QK}}$.

The model M_Q behaves like M on all modal formulas and is therefore a model of Γ. Because we can check it is a model of $H_{\mathbf{QK}}$, we have $H_{\mathbf{QK}} \cup \Gamma \not\models \bot$.

We have shown that $H_{\mathbf{QK}} \models \varphi$ if and only if $\mathbf{K} \models \varphi$ by proving both directions of the equivalence. □

10.3 Example

We illustrate with an example of a simple line of reasoning in the system **D** of modal logic and its translation to Qiana. In natural language, we take the following premises:

- "Necessarily, either Juliet faked her death or she killed herself."
- "Necessarily, if Juliet faked her death, she was unhappy with her family's decision."
- "Necessarily, if Juliet killed herself, she was unhappy with her family's decision."

We want to prove that it is possible that Juliet was unhappy with her family's decision.

First, we write this in modal logic:

A : Juliet faked her death
B : Juliet killed herself
C : Juliet was unhappy with her family's decision

The premises can then be formalized as:

$$\Box(A \vee B)$$
$$\Box(A \to C)$$
$$\Box(B \to C)$$

First, we notice the following propositional tautology:

$$\models (A \to C) \to ((B \to C) \to ((A \vee B) \to C))$$

By necessitation (N), this gives:

$$\Box(A \to C) \to ((B \to C) \to ((A \vee B) \to C))$$

By applying K multiple times along with the *modus ponens* rule, we derive:

$$\Box C$$

Then, axiom D gives us:

$$\Diamond C$$

We now translate this to Qiana. We define arity-0 predicates A, B, and C to represent the propositions. The necessity operator ($\Box$) is treated as a context. The translation of the premises is:

$$\textit{ist}(\Box, \underline{A \vee B})$$
$$\textit{ist}(\Box, \underline{A \to C})$$
$$\textit{ist}(\Box, \underline{B \to C})$$

Which can be written more concisely as:

$$\Box\underline{(A \vee B)}$$
$$\Box\underline{A \to C}$$
$$\Box\underline{B \to C}$$

We derive *Tauto*$(\underline{(A \to C) \to ((B \to C) \to ((A \vee B) \to C))})$ by Lemma 1. Then, by the rule *Modus Ponens* and Axiom QN, we obtain:

$$\Box\underline{(A \to C) \to ((B \to C) \to ((A \vee B) \to C))}$$

By repeatedly applying QK and the rule *Modus Ponens*, we derive:

$$\Box C$$

Finally, using axiom QD and the rule *Modus Ponens*, we conclude:

$$\Diamond C$$

As we can see, the structure of the proof is very similar to that of pure modal logic. The only steps that do not mirror the ones from the modal logic reasoning are those where we use the *Tauto* predicate, but its behavior is guaranteed by Lemma 1.

11. Discussion

11.1 About the definition of the set V of quotable variables

In Section 4 we defined V as a finite subset of V_∞, 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_∞. But all the bound variables can be replaced with elements of V_∞ without issue. The only aspect of V that matters is that it has the same size as $\underline{V}$. The notion of "quotable variable" amounts to a limitation on the number of distinct variables in a quotable formula, along with giving us a convenient way to state our axioms.

11.2 Axioms for disambiguation

The axioms on the behavior of contexts introduced in Section 5 are minimal. In particular, we do not enforce the correct interpretation of terms in a context. For example, the formula *ist*$(c, \underline{P}(\underline{2}))$ does not imply *ist*$(c, \underline{P}(\underline{1+1}))$; even where 1+1=2 is taken for granted. This nonenforcement of the correct interpretation of terms in contexts extends to the function $\mathbb{Q}$. The recursive definition of $\mathbb{T}$ includes a mechanism to unwrap $\mathbb{Q}$ symbols, but this is not enforced for arbitrary contexts.

In applications where the correct interpretation of terms in contexts is important, the following axioms can be included to enforce this behavior:

$$\forall c, t. \mathit{Wft}(t) \rightarrow \mathit{ist}(c, t \equiv \mathbb{Q}(E(t)))$$

$$\forall c, t_1, t_2, t.\ \mathit{ist}(c, t_1 \equiv t_2) \rightarrow \mathit{ist}(c, t \underline{\leftrightarrow} sub(t, t_1, t_2))$$

where we recall sub is substitution $t[t_1 \leftarrow t_2]_q = sub(t, t_1, t_2)$; and E is the evaluation symbol, matching a quoted term to its value (the opposite of $\mathbb{Q}$).

12. 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.

Furthermore, we have extended Qiana to reason about temporality with event calculus. We have also presented an alternative many-sorted version of Qiana that includes time. Finally, we have shown how the usual systems of modal logic can be written within Qiana.

We expect Qiana to be usable for and adaptable to various types of contextual reasoning cases, including reasoning on hypothetical scenarios, fake news, legal reasoning, and different points of view. In future work we intend to produce the tools necessary to translate natural language knowledge to Qiana and to develop a Qiana-based system for reasoning on contexts and beliefs. Once this is done, the ability of Qiana to quantify over both formulas and contexts while remaining compatible with automated theorem provers will make it a powerful tool for interpretability.

References

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

Blackburn, P., Rijke, M. d., & Venema, Y. (2001). *Modal Logic*. Cambridge Tracts in Theoretical Computer Science. Cambridge University Press.

Brewka, G., Eiter, T., Fink, M., & Weinzierl, A. (2011). Managed multi-context systems. In *IJCAI*.

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

Buvac, S., Buvac, V., & Mason, I. A. (1994). The semantics of propositional contexts. In *ISMIS*.

Buvac, S., & Mason, I. A. (1993). Propositional logic of context. In *AAAI*.

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

Chellas, B. F. (1980). Modal logic: An introduction..

Coumes, S., Paris, P.-H., Schwarzentruber, F., & Suchanek, F. M. (2024). Qiana: A First-Order Formalism to Quantify over Contexts and Formulas. In *Proceedings of the 21st International Conference on Principles of Knowledge Representation and Reasoning*, pp. 295–305.

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

Garson, J. (2024). Modal Logic. In Zalta, E. N., & Nodelman, U. (Eds.), *The Stanford Encyclopedia of Philosophy* (Spring 2024 edition). Metaphysics Research Lab, Stanford University.

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

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

Giunchiglia, F. (1997). Contextual reasoning. *Epistemologia*, *16*.

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

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., & Fikes, R. (2004). Contexts for the semantic web. In *ISWC*.

Halpern, J. Y., & 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., & Seaborne, A. (2023). Rdf-star and sparql-star..

Hintikka, J. (1962). Back matter. *The Philosophical Review*, *71*(3).

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

Manzano, M. (1996). *Extensions of First Order Logic*. Cambridge University Press, New York.

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

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

Mendelson, E. (2009). *Introduction to mathematical logic*. Chapman and Hall/CRC.

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

Moore, R. C. (1980). Reasoning about knowledge and action. Tech. rep. ADA126244, Massachusetts institute of Technology.

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

Mossakowski, T., Codescu, M., Kutz, O., Lange, C., & 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., & Campbell, R. H. (2003). An infrastructure for context-awareness based on first order logic. *Pers. Ubiquitous Comput.*, *7*(6), 353–364.

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

Shanahan, M. (2000). The event calculus explained. *In Artificial Intelligence LNAI*, *1600*.

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., & 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.

Appendix A. Proof of truth definition

To prove Property 4 we will instead prove Lemma 2, which is stronger.

Lemma 2. *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 varialbes of A. Then*

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

We prove Lemma 2 by induction on A.

Base case Let M be a model of $A1$-4. Let us prove that $M \models \forall x_1, \ldots, x_n.\ \mathbb{T}(\underline{p}(t_1, \ldots, t_m)) \leftrightarrow \mu^{\dot{-}1}(\underline{p}(t_1, \ldots, t_m))$. For all assignments σ of variables $x_1, \ldots, x_n$, we have:

$$\begin{aligned}
&M, \sigma \models \mathbb{T}(\underline{p}(t_1, \ldots, t_m)) \\
&\text{iff } M, \sigma \models p(\mu^{\dot{-}1}(t_1), \ldots, \mu^{\dot{-}1}(t_m)) && \text{as } M \models \text{A1} \\
&\text{iff } M, \sigma \models \mu^{\dot{-}1}(\underline{p}(t_1, \ldots, t_m)) && \text{by definition of } \mu^{\dot{-}1}
\end{aligned}$$

Negation Let M be a model of $A1$-4. Let us prove that $M \models \forall x_1, \ldots, x_n.\ \mathbb{T}(\underline{\neg} A) \leftrightarrow \mu^{\dot{-}1}(\underline{\neg} A)$. For all assignments σ of variables $x_1, \ldots, x_n$, we have:

$$\begin{aligned}
&M, \sigma \models \mathbb{T}(\underline{\neg} A) \\
&\text{iff } M, \sigma \not\models \mathbb{T}(A) && \text{as } M \models \text{A3} \\
&\text{iff } M, \sigma \not\models \mu^{\dot{-}1}(A) && \text{by IH} \\
&\text{iff } M, \sigma \models \mu^{\dot{-}1}(\underline{\neg} A) && \text{by definition of } \mu^{\dot{-}1}
\end{aligned}$$

$\underline{\forall}$ Let M be a model of $A1$-4. Let us prove that $M \models \forall x_1, \ldots, x_n.\ \mathbb{T}(\underline{\forall}(\underline{x}, A)) \leftrightarrow \mu^{\dot{-}1}(\underline{\forall}(\underline{x}, A))$. For all assignments σ of variables $x_1, \ldots, x_n$, we have:

$$\begin{aligned}
&M, \sigma \models \mathbb{T}(\underline{\forall}(\underline{x}, A)) \\
&\text{iff } M, \sigma \models \forall x.\ \mathbb{T}(A[\underline{x} \leftarrow quote(x)]_q) && \text{as } M \models \text{A4} \\
&\text{iff } M, \sigma \models \forall x. \mu^{\dot{-}1}(A[\underline{x} \leftarrow quote(x)]_q) && \text{by IH} \\
&\text{iff } M, \sigma \models \mu^{\dot{-}1}(\underline{\forall}(\underline{x}, \underline{\neg} A)) && \text{by definition of } \mu^{\dot{-}1}
\end{aligned}$$

Appendix B. Proof of finite axiomatization

We will now prove Theorem 1. To simplify the proof, we will omit the existence of schema A11 and its finite counterpart schema A11^{fin}. The reason is that they are vastly orthogonal to the other difficulties of the proof and can be handled in the same way as we deal with the other axioms in this proof, except they form a more straightforward case. Hence, they would only bloat the proof with unnecessary tedium largely redundant in spirit with the rest of the reasoning.

Therefore, for the purpose of this proof, we assume:

$$\begin{aligned}
H_C &:= H \cup H^{\mathit{fin}}_{\mathit{ist}} \cup H_{\mathbb{T}} \\
H^{\mathit{fin}}_C &:= H \cup H^{\mathit{fin}}_{\mathit{ist}} \cup H^{\mathit{fin}}_{\mathbb{T}} \cup H^{\mathit{fin}}_{\mathit{tools}}
\end{aligned}$$

We now prove that H_C is coherent if and only if H^{fin}_C is coherent through Proposition 10 and Proposition 11.

Proposition 10.

$$H_C^{\mathit{fin}} \not\models \bot \rightarrow H_C \not\models \bot$$

Proof. We want to prove that if $H_C^{\mathit{fin}} = H \cup H_{\mathit{ist}}^{\mathit{fin}} \cup H_{\mathit{tools}}^{\mathit{fin}} \cup H_{\mathbb{T}}^{\mathit{fin}}$ has a model, then $H_C = H \cup H_{\mathit{ist}}^{\mathit{fin}} \cup H_{\mathbb{T}}$ also has one. We will now prove $H_C^{\mathit{fin}} \models H_{\mathbb{T}}$, which is sufficient.

Lemma 3. *Let $t_1, t_2 \in \mathcal{Q}_v$ with free variables $x_1, \ldots, x_m$, all within quotes. Then for all $\underline{x}$, $H_C^{\mathit{fin}} \models \forall x_1, ..., x_m.\ \mathit{Sub}(t_1, \underline{x}, t_2) = t_1[\underline{x} \leftarrow t_2]_q$.*

Proof. Proven by direct recursion on t_1. □

Lemma 4. *Let $t \in \mathcal{Q}_v$ with free variables $x_1, \ldots, x_m$, all within quotes. Then $H_C^{\mathit{fin}} \models \forall x_1, \ldots, x_m.\ \mathit{Term}(t)$.*

Proof. Proven by direct recursion on $\mathbb{T}$. □

Lemma 5. *Let $t \in \underline{\mathcal{T}_v}$ with free variables $x_1, \ldots, x_m$, all within quotes. Then*

$$H_C^{\mathit{fin}} \models \forall x_1, \ldots, x_n.\ E(t) = \dot{\mu^{-1}}(t)$$

Proof. Proven by direct recursion on $\mathbb{T}$, as E is by construction built on the same recursion as $\dot{\mu^{-1}}$. □

Armed with these lemmas, we can prove all schemas of $H_{\mathbb{T}}$ with their direct counterpart from $H_{\mathbb{T}}^{\mathit{fin}}$.

We provide a sketch for the more complicated case of schema A1, highlighting the most important elements of the proof.
Let $t_1, \ldots, t_k \in \underline{\mathcal{T}_v}$ with free variables $x_1, \ldots, x_n$.
By construction of *Wft* we have $\forall x_1, \ldots, \forall x_n.\ \mathit{Wft}(t_i)$ for all i. Hence by schema A1$^{\text{fin}}$ we have $\forall x_1, \ldots, \forall x_n.\ T(\underline{p}(t_1, \ldots, t_k)) \leftrightarrow p(E(t_1), \ldots, E(t_k))$. Lemma 5 allows us to conclude. □

We now prove the other direction of the equivalence.

Proposition 11.

$$H_C \not\models \bot \rightarrow H_C^{\mathit{fin}} \not\models \bot$$

Proof. Let M be a model of H_C. We will define a model M_f and prove that it is a model of H_C^{fin}. We define M_f as follows:

Let D be the domain of M. Recall that $\mathcal{T}$ is the set of all terms under S. Without loss of generality, we assume $D \cap \mathcal{T} = \emptyset$. We define D_f as the set obtained by adding D and removing variables to and from the recursive definition of $\mathcal{T}$. We define the M-interpretation of elements of $t \in D_f$ as the value in D that we obtain by recursively evaluating $\mathbb{T}$ under M.

- Under M_f, the interpretation of any function symbol from S is to recursively build the element of D_f. Intuitively, we "only" store the terms as we evaluate them without performing any other operation.

- Under M_f, the interpretation of any predicate symbol is to turn all arguments in D_f to their M-interpretation and then interpret the predicate as in M.
- $=$ is true equality on D_f.
- Wft is true only on elements of D_f recursively built with elements of $\underline{V}$, $\underline{F}$, and $quote(y)$ for $y \in D$. Remark that *Wft* is the minimal predicate that satisfies schemas A19 to A21 and is built with schemas that follow the recursive construction of $\underline{\mathcal{T}_v}$.
- *Term* is likewise the minimal predicate to satisfy its definition schemas (schemas A17 to A18). It is only true on elements recursively built as terms.
- *Sub*$(t_2, \underline{x}, t_1)$ is recursive and localy behaves as $t_2[\underline{x} \leftarrow t_1]_q$ if the top symbol of t_2 is coherent with t_2 being in $\mathcal{Q}_v$. Otherwise sub simply returns t_2. If the first argument of sub is not in $\underline{V}$, then sub returns t_2. The intuition is that sub is a recursive function defined as "behaves like $t_2[\underline{x} \leftarrow t_1]_q$ if it makes sense to do so (locally). Otherwise, return t_2.".
- E likewise behaves like $\mu^{\text{-}1}$ where it locally makes sense to do so and otherwise is identity. Remark that $E(t) = \mu^{\text{-}1}(t)$ for $t \in \underline{\mathcal{T}_v}$.

Lemma 6. *Let φ be a formula well-defined on S. Then $M \models \varphi$ iff $M_f \models \varphi$*

Proof. (Sketch) At each step of term evaluation, the M-interpretation of the result is equal to the same operation applied to the M interpretations of the arguments. Since every term will have to be interpreted through a predicate symbol from S, and therefore sent to its M-interpretation before evaluation, everything happens as though only the M-interpretation of the values was considered. This is exactly the application of M itself to the formulas. This proves Lemma 6. □

We recall that $M \models H_C$, which means $M \models H \cup H^{fin}_{ist} \cup H_{\mathbb{T}} \cup H_{\mathbb{T}}$. Hence and by Lemma 6, $M_f \models H$ and $M_f \models H^{fin}_{ist}$. By directly applying the definitions, we see that $M_f \models H^{fin}_{tools}$. Since $H^{fin}_C = H \cup H^{fin}_{ist} \cup H^{fin}_{tools} \cup H^{fin}_{\mathbb{T}}$, we now need to prove only that $M_f \models H^{fin}_{\mathbb{T}}$. We prove this by checking that M_f accepts schemas A1$^{\text{fin}}$ to A4$^{\text{fin}}$. As they are quite similar to one another, we only provide explanations for the more complicated case of schema A1$^{\text{fin}}$.

Let p be a predicate symbol of arity n. Further, let $v_1, \ldots, v_n \in D_f$ such that $M_f \models$ *Wft*(v_i) for all i. By definition of *Wft*, there is some selection of variables $x_1, \ldots, x_m$, a valuation $\sigma : x_1, \ldots, x_m \to D_f$, and terms $t_1, \ldots, t_n \in \underline{\mathcal{T}_v}$ such that: $\forall i \in [1, n], (M_f, \sigma)(t_i) = v_i$. By definition of $H_{\mathbb{T}}$ and Lemma 6, we have: $M_f \models \forall x_1, \ldots, x_m.\ T(\underline{p}(t_1, \ldots, t_n)) \leftrightarrow p(\mu^{\text{-}1}(t_1), ..., \mu^{\text{-}1}(t_n))$. Hence $M_f, \sigma \models T(\underline{p}(t_1, \ldots, t_n)) \leftrightarrow p(\mu^{\text{-}1}(t_1), ..., \mu^{\text{-}1}(t_n))$. E equals $\mu^{\text{-}1}$ on $\underline{\mathcal{T}_v}$, hence this gives $M_f, \sigma \models T(\underline{p}(t_1, \ldots, t_n)) \leftrightarrow p(E(t_1), ..., E(t_n))$. Finally, this gives $M_f \models \overline{T(p}(v_1, \ldots, v_n)) \leftrightarrow p(E(v_1), \ldots, \overline{E(v_n))}$, which is what we wanted to prove.

Remark that we can handle the case of schema A4$^{\text{fin}}$ in a vastly similar fashion, relying on the similarity of Sub to $\_[\_ \leftarrow \_]_q$ instead of to similarity of E to $\mu^{\text{-}1}$. □